$\varepsilon-$MC nets: A Compact Representation Scheme for Large Cooperative Game Settings

Errikos Streviniotis[1], Athina Georgara[2], and Georgios Chalkiadakis[1]

[1] Technical University of Crete, Chania, Greece
[2] Artificial Intelligence Research Institute, CSIC, Barcelona, Spain
{estreviniotis, gehalk}@intelligence.tuc.gr, ageorg@iiia.csic.es

Abstract. In this paper we put forward ε-MC nets, a novel succinct rule-based representation scheme for large cooperative games. First, we provide a polynomial algorithm that reaches the proposed representation by exploiting the agents' estimates over marginal contributions, along with their acceptable information loss, ε, regarding these estimates. Then we introduce the notion of *equivalence classes* of agents, and exploit it to *(i)* obtain an even more compact representation; and *(ii)* derive *new*, previously unheld, beliefs over the value of unobserved agent collaboration patterns. Moreover, we present theoretical and empirical results on the information loss arising from this "representational compression", and on the degree of succinctness achieved. Notably, we show that an arbitrary number of merges to reach the compressed representation, exhibits an information loss that does not exceed ε. Finally, we provide theoretical guarantees for the coalitional relative error and the Shapley value in the ε-MC net with respect to the initial representation.

Keywords: Knowledge representation · Large Coalitional Games · MC nets · Rule-based representation · Equivalent agents

1 Introduction

Coalitional games [2] capture settings where individuals need to form coalitions in order to fulfil some complicated task, which they would not be able to accomplish on their own or to achieve better outcomes. As the number of individuals scales up, the number of different possible coalitions one may participate in rises exponentially. Thus, it is essential to find schemes for representing large coalitional games in an efficient way. Moreover, in large open multiagent systems, we may have hundreds or thousands of agents which form coalitions in order to perform complex tasks. In such large settings, it is unrealistic to assume that we can have complete knowledge over every possible collaboration pattern between the agents. As such, it is natural to assume that we have *estimates* over the value of potential collaboration patterns. Fully representing such multiagent systems can be extremely inefficient as the number of agents rises.

In this light, we provide a novel representation that encodes the prior information of the agents over the value of some observed collaboration patterns in a succinct way. Specifically, we build on the celebrated MC-nets representation [8,7,3,6], and enhance it with the ability to exploit similarities among agent collaboration patterns. To do so, we equip our scheme with an $\varepsilon \in \mathbb{R}^+$ signifying how far away from our perceived value of a collaboration pattern we are willing to deviate in order to compress an original rule. Acknowledging that the environment is not fully observable (i.e., the values of the rules of the initial representation may be different than the true ones), allowing a deviation of at most ε in order to compress the representation is a reasonable trade-off. The $\varepsilon-$MC nets representation captures collaboration patterns with similar values among similar agents; encodes them into compact rules; and retains the highly attractive *full expressiveness* and *conciseness* properties of the MC net representation.[3]

As such, our contributions in this paper are as follows. First, we propose a novel succinct representation scheme for (large) cooperative games. Then, we provide an algorithm that compresses the original game to reach the $\varepsilon-$MC nets representation; study its complexity; and provide theoretical results regarding: *(i)* the information loss of the perceived value of agent collaboration patterns; *(ii)* the relative error on the coalitional values; and *(iii)* the (estimated) *Shapley value* [11] of the game after the representation's compression, showing that they are bounded by ε, and a value proportional to ε, respectively. We extend our algorithm so that it exploits "equivalence classes" of agents in order to produce an even more compact representation of the game, inspired by the original work of [8]. This variant can also *produce* new, previously unknown, collaboration patterns among agents. Finally, we conduct a systematic evaluation of our algorithm, studying its behaviour in various realistic settings, and reporting on the degree of succinctness achieved and other measures of interest. Our experimental results confirm the effectiveness of our approach.

2 Preliminaries

Let $N = \{1, \ldots, n\}$ be a finite non-empty set of agents,with $|N| = n$. A coalitional game with transferable utility, also referred to as *characteristic function game (CFG)*, is given by a pair $\langle N, v\rangle$, where N is a set of agents, and $v : 2^N \rightarrow \mathbb{R}$ a characteristic function that maps each coalition $S \subseteq N$ to a real number [2]. The *Marginal Contribution Networks* representation [8] for a CFG $G = \langle N, v\rangle$ is given by a set of rules of the form: $Pattern \rightarrow value$. A pattern consists of positive and negative literals, where each literal corresponds to some agent. The "positiveness" or "negativeness" of the literal indicates that agent's presence or absence in the pattern, respectively. A rule $r : p_1 \wedge p_2 \wedge \cdots \wedge p_x \wedge \neg n_1 \wedge \neg n_2 \wedge \cdots \wedge \neg n_y \rightarrow val_r$ applies on a coalition $S \subseteq N$, denoted by $S \models r$, *iff* each positive literal p_i exists in S, i.e, $p_i \in S$ for $i = 1, \ldots, x$, and no negative literal n_j exists in S, i.e, $n_j \notin S$ for $j = 1, \ldots, y$. Given a coalition S, we can compute its utility by summing up the values of all the rules that apply to S: $v(S) \equiv \sum_{S\models r} val_r$. [8] shows that *(i)* any CFG can be

[3] This work is an improved version of our earlier work presented in [13].

represented by a set of such rules [8], and *(ii)* computing the Shapley values [11] in an MC-net representation is easy.

3 The ε-MC Net Representation

Here we describe the $\varepsilon-$MC nets representation scheme. An ε-MC net constitutes a compact set of rules based on an initial MC-net representation. The compactness is achieved by merging patterns and regulating the rule-values accordingly. Let $N = \{a_1, \cdots, a_n\}$ be a set of agents ($|N| = n$) and $L = \{i, \neg i \ \forall a_i \in N\}$ be the set of literals corresponding to agents in N ($|L| = 2 \cdot n$). Formally,

Definition 1 ***($\varepsilon-$MC net Rule).*** An $\varepsilon-$MC net rule is of the form $i \wedge \mathrm{CG} \rightarrow val$. Here $i \in L$ is called the *common literal* and CG, the i's *collaborations group*, is of the form $\left\{\{\bigwedge_{j\in L, j\neq i} j\}, \{\bigwedge_{k\in L, k\neq i} k\}, \cdots\right\}$, and represents a set of distinct collaboration patterns among agents in $N \setminus \{a_i\}$. Each pattern $p = \left\{\bigwedge_{j\in L, j\neq i} j\right\}$ is a conjunction among a subset of literals in L; while an agent's positive and negative literals cannot both appear in the same p. $val \in \mathbb{R}$ expresses the estimated value of the collaboration pattern between literal i and any pattern of literals $p \in \mathrm{CG}$. $\varepsilon \in \mathbb{R}^+$ is a parameter denoting how far from the rule's value we are willing to depart in order to compress the representation.

Intuitively, the $\varepsilon-$MC net rule denotes that a collaboration between i and *any* pattern of literals p in CG, has an expected value val. (Thus, in reality CG is a disjunction $\vee$ of patterns.) Then, ε represents the margin of information loss we are willing to accept in order to compress the representation; specifically, it denotes the acceptable information loss for merging an MC-net rule with some other rule (be that an MC-net or an ε-MC net one; this will be clarified in what follows). Naturally, the larger the ε, the wider these margins are, and the more compact the representation we obtain. Notice that any MC-net rule can be trivially written as an ε-MC net rule (with one pattern $p \in \mathrm{CG}$ and an arbitrary i as the common literal); while any ε-MC net rule can be rewritten as a collection of MC-net rules (see e.g., Section 3.2 below).

The process of compressing an initial MC-net set of rules to a final set of $\varepsilon-$MC nets rules works by progressively building the collaborations group around some common literal, via merging rules. We distinguish two types of merging: *(a)* the full-merge, and *(b)* the half-merge. The *full-merge* describes the merge of two MC-net rules that produces a new $\varepsilon-$MC net rule. A full-merge can occur if the rules share a *common literal* (indicating the presence or absence of mutual agent) between the rules, and if the values of the two rules differ by at most ε, where ε is the margin of information loss that we are willing to accept. Formally, two MC-net rules $r_1 : Pattern_1 \rightarrow val_1$ and $r_2 : Pattern_2 \rightarrow val_2$, where $Pattern_1$ and $Pattern_2$ are a conjunction of literals,[4] can be full-merged ($\textsc{CanFullMerge}(r_1, r_2)$) *iff*:

[4] Note that a pattern may consist of only one literal, representing singletons, thus we can assume that $i \rightarrow val \equiv i \wedge \bot \rightarrow val$, where $\bot$ is the empty clause.

Algorithm 1 Merging MC-net Rules

1: $R \leftarrow$ initial set of MC-net rules of size m
2: $R' \leftarrow \emptyset$
3: **for** $r \in R$ **do**
4: $\quad$ CG $\leftarrow \emptyset$
5: $\quad V_{\text{CG}}, min, max, avg \leftarrow val_r$
6: $\quad$ Remove r from R
7: $\quad$ **for** $r' \in R$ **do**
8: $\quad\quad$ **if** CanFullMerge(r, r') **OR** CanHalfMerge(r, r') **then**
9: $\quad\quad\quad$ Insert $val_{r'}$ in V_{CG}
10: $\quad\quad\quad$ Insert non common literals in CG
11: $\quad\quad\quad$ Update min, max, avg variables
12: $\quad\quad\quad$ Rule r becomes: $\{common \quad literal\} \wedge \text{CG} \rightarrow avg\{V_{CG}\}$
13: $\quad\quad\quad$ Remove r' from R
14: $\quad\quad$ **end if**
15: $\quad$ **end for**
16: $\quad$ Insert r in R'
17: **end for**
18: **return** R'

(I) $i \equiv j$, where $i \in Pattern_1$ and $j \in Pattern_2$. The two literals are identical if they refer to the same agent, and they are both positive or both negative;

(II) $|val_1 - val_2| \leq \varepsilon$, i.e., the values of the two rules are at most ε away.

The resulting $\varepsilon-$MC net rule is $r_{merged} : l_{common} \wedge \text{CG} \rightarrow \frac{val_1+val_2}{2}$, where l_{common} is the common literal, and $\text{CG} = \{Pattern_1', Pattern_2'\}$, where $Pattern_1' = Pattern_1 \backslash \{i\}$ and $Pattern_2' = Pattern_2 \backslash \{j\}$. Similarly, given an ε-MC net rule $r : i \wedge \text{CG} \rightarrow avg\{V_{CG}\}$, where CG is a set of patterns and V_{CG} is the set containing all the values of the rules merged so far to produce r, and their average value $avg\{V_{CG}\}$ is the value of r; and an MC-net rule $r_3 : Pattern_3 \rightarrow val_3$, we say that r_3 can be half-merged with r (CanHalfMerge(r, r_3)) *iff*:

(III) $i \equiv j$, where $j \in Pattern_3$. The two literals are identical if they refer to the same agent, and they are both positive or both negative; and

(IV) $avg\{V_{\text{CG}}\} - \varepsilon \leq val_3 \leq avg\{V_{\text{CG}}\} + \varepsilon$; and

(V) $\max_{v \in V_{\text{CG}}} v - \varepsilon \leq avg\big\{V_{\text{CG}} \cup \{val_3\}\big\} \leq \min_{v \in V_{\text{CG}}} v + \varepsilon$.

The resulting rule after the half-merge is: $r_{merged} : i \wedge \text{CG}' \rightarrow avg\big\{V_{\text{CG}} \cup \{val_3\}\big\}$, where $\text{CG}' \equiv \text{CG} \cup \{Pattern_3'\}$, and $Pattern_3' = Pattern_3 \backslash \{j\}$. (We discern two sets of merging conditions, since we use different kind of rules in each case.)

Algorithm 1 illustrates a quadratic (in the number of rules) algorithm that performs a series of full- and half-merges, compressing the initial set of MC-net rules R to a succinct representation captured via R' of $\varepsilon-$MC nets rules, with $|R'| \leq |R| = m$. Going through Algorithm 1, we see that the outer loop in line 3 needs exactly m iterations, where m is the size of the initial set of MC-net rules. The inner loop in lines 7-15 needs at most m iterations; while the condition in line 8 is trivial. As such the complexity of the algorithm is $\mathcal{O}(m^2)$.

In a nutshell, the ε-MC net representation has a number of desired properties. First of all, it retains the fully expressiveness and the conciseness of the classic MC nets representation, but is also able to form even more compact rules by performing a series of merges between rules with similar values that contain common literals. Moreover, note that the proposed representation applies to settings with uncertainty by simply "paying" an additional information loss, which in any case never exceeds ε, as we show in the next section. Another beneficial property of the ε-MC net representation is that it allows us to exploit the common literal to prune the initial rule space while computing the coalitional values by disregarding rules where their common literal is not part of the coalition of interest. Finally, our scheme easily adopts a notion of equivalent classes of agents to not only further compress the representation, but also to *learn* new rules from previously unheld information that emerges through the compression process.

3.1 A bound on the values of $\varepsilon-$MC nets rules

Here we provide a bound that our representation guarantees on the maximum information loss of any $\varepsilon-$MC nets rule with respect to its initial MC-net rule. We show that due to the necessary conditions for full- and half-merge, the information loss on the initial values is at most ε. Obviously this bound is tight.

Lemma 1. *For any full-merge between two MC-net rules r_x and r_y producing a new $\varepsilon-$MC net rule r_z, it holds that: $|val_z - val| \leq \frac{\varepsilon}{2}$, where $val = \{val_x, val_y\}$.*

Proof. It is straightforward following condition (II).

Theorem 1. *For any MC-net rule $\tilde{r} : i \wedge Pattern_j \rightarrow val$ that is merged (either full- or half-merged) in the process of reaching an ε-MC net rule $r_{merged} : i \wedge CG_{merged} \rightarrow v_{merged}$, with $CG_{merged} = \{Pattern_j, Pattern_k, \cdots, Pattern_x\}$ and $v_{merged} = avg\{V_{CG_{merged}}\}$, it holds that: $|v_{merged} - val| \leq \varepsilon$.*

Proof. All proofs can be found in `https://rb.gy/aexgbt`.

Note that since the coalitional utility is derived by *summing* the rules applying in the coalition at hand, a direct consequence of Theorem 1 is that the utility of a coalition S is computed via the ε-MC nets rules, exhibits an information loss *equal at most to* $\sum_{S \models r} \varepsilon$, where r is any ε-MC net rule that applies to S.

3.2 Decompression of the ε-MC net representation

Given an initial MC-net representation described by a set of rules R, we achieve a compressed $\varepsilon-$MC net representation described by a set of rules R' following the merging rules detailed in the previous section. We can always decompress R' into a set $\hat{R}$ of classic MC-net rules (with $|\hat{R}| \geq |R'|$ and $|\hat{R}| = |R|$): $R \overset{\text{compress}}{\longrightarrow} R' \overset{\text{decompress}}{\longrightarrow} \hat{R}$.

We present the process in order to be comprehensive and provide clarity. Let an ε-MC net rule $r' : i \wedge CG \rightarrow v_{r'}$ where $CG = \{\text{pattern}_1, \text{pattern}_2, \cdots, \text{pattern}_k\}$, which has been reached by the merges (both full- and half-) over the rules in R. Any such $\varepsilon-$MC net rule r' can then always be substituted by the rules in $\hat{R}$:

$$R = \left.\begin{matrix} r_1 : i \wedge \text{pattern}_1 \rightarrow val_1 \\ r_2 : i \wedge \text{pattern}_2 \rightarrow val_2 \\ \vdots \\ r_k : i \wedge \text{pattern}_k \rightarrow val_k \end{matrix}\right\} k \quad ; \quad \hat{R} = \left.\begin{matrix} \hat{r}_1 : i \wedge \text{pattern}_1 \rightarrow v_{r'} \\ \hat{r}_2 : i \wedge \text{pattern}_2 \rightarrow v_{r'} \\ \vdots \\ \hat{r}_k : i \wedge \text{pattern}_k \rightarrow v_{r'} \end{matrix}\right\} k$$

Hence $\hat{R}$ is a decompressed set of rules such that $|R| = |\hat{R}|$. Note that there is exact correspondence between R and $\hat{R}$, i.e., for each rule in R there is exactly one rule in $\hat{R}$ describing the same collaboration. We remind the reader that according to Theorem 1 for any initial rule $r_j : i \wedge \text{pattern}_j \rightarrow val_j$ in R and its corresponding decompressed rule $\hat{r}_j : i \wedge \text{pattern}_j \rightarrow v$ in $\hat{R}$ it holds that $|v - val_j| \leq \varepsilon$. Finally, notice that the decompressed representation is a classic MC-net representation. A similar procedure can be applied in the variant with equivalent classes, however, in this case we have that $|\hat{R}| \geq |R|$.

3.3 Relative Error Guarantees

Earlier we presented a theoretical bound on the *absolute* error, ε, between the merged value and the value of an initial MC-net rule. However, this suggests an *a priori knowledge* over the magnitude of the rule values, which may not be always the case, especially in large cooperative environments. Instead, one may find more beneficial to express a *relative* error, $\tilde{\varepsilon}$, on the coalitional values. Thus, we now explore the relation between relative and absolute errors in order to exploit the useful results previously obtained.

Assume we want to compress the initial MC-net representation into an ε-MC net one, while the relative error of the coalitional values does not exceed $\tilde{\varepsilon}$. The value of some S in the initial MC-net R is given by $v(S) = \sum_{i=1}^{\mu_S} val_{r_i}$ where μ_S is the number of rules that apply on S; similarly, in a decompressed ε-MC net R' the value of S is $v'(S) = \sum_{i=1}^{\mu_S} v_{r'_i}$.Thus, for any S we demand that $\left|\frac{v(S)-v'(S)}{v(S)}\right| \leq \tilde{\varepsilon}$. Following Theorem 1, the absolute error between the two coalitional values is: $|v(S) - v'(S)| \leq \mu_S \cdot \varepsilon$, and thus for the relative error it holds that $\left|\frac{v(S)-v'(S)}{v(S)}\right| \leq \frac{\mu_S}{v(S)} \cdot \varepsilon$, which does not exceed $\tilde{\varepsilon}$ if and only if $\varepsilon \leq \frac{v(S)}{\mu_S} \cdot \tilde{\varepsilon}$. Thus, we need to find an acceptable information loss ε such that for any coalition S it holds $\varepsilon \leq \frac{v(S)}{\mu_S} \cdot \tilde{\varepsilon}$; in other words any absolute error $\varepsilon \leq \tilde{\varepsilon} \cdot \min_{S \subseteq N} \frac{v(S)}{\mu_S}$.

3.4 Equivalence Classes of Agents

Here we discuss a variant of our representation that exploits not only (the presence or absence of) *mutual* agents (indicated via common literals), but also *equivalence classes* of agents: agents in the same class may have similar behaviour, preferences or properties. Such a variant can be very useful in partially observable environments, where we are aware of a subset of collaboration patterns, or in settings where new agents arrive over time. Considering equivalences among agents we manage to: *(a) compress even more* the representation compared to the initial version; *(b) extract underlying collaboration patterns that are "new"*, as they could not have been observed in the initial set of MC-net rules.

Definition 2 (Equivalent Agents). *Given a set of agents N, and a similarity metric $s : N \times N \to [0,1]$, two agents i and j are equivalent iff $s(i,j) \geq T$, with T a threshold in $[0,1]$.*[5]

In this version, the rules take the form $\Omega_{\text{equiv}} \wedge \text{CG} \to val$, where Ω_{equiv} is a set of equivalent agents (all as positive or all as negative literals) and it substitutes the *common literal* of the initial representation. In words, a rule $\Omega_{\text{equiv}} \wedge \text{CG} \to val$ is interpreted as: *Our estimate of the collaboration between any literal $i \in \Omega_{equiv}$ with any pattern $p \in \text{CG}$ is equal to val*. We can easily obtain the variant with the equivalence classes by slightly changing Algorithm 1. That is, we simply need to check whether the rules r and r' have agents in the same *equivalence class* instead of just mutual agents (i.e., common literals). In other words, we need to check if there exists a literal i in Ω_{equiv} on rule r such that any of the literals in rule r' belongs in the same equivalence class. Such a modification results to an increase in the computational complexity to $\mathcal{O}(n^2 \cdot m^2)$, where n is the number of agents, and m is the size of the initial set of rules.[6]

Intuitively, in this version agents belonging in the same class are expected to have similar behaviour, preferences or properties—for example, in a search & rescue mission all firefighters comprise one equivalence class, while all nurses another. We are thus able to obtain an estimate over the utility of a previously unseen collaborative pattern based on our expectations that equivalent agents behave similarly. Of course, there is a trade-off: to extract new patterns, we drop our guarantees provided by our theoretical results.

Now, this variation may result in *ambiguous* $\varepsilon-$MC nets rules: depending on the way agents' equivalence is determined, we may end up producing overlapping rules, i.e., multiple $\varepsilon-$MC nets rules may apply to the very same collaborative pair. To overcome this ambiguity we set the *post-merge estimate* for a collaboration pattern p to the average value of the rules that apply to p.

4 Shapley values for ε-MC nets

In this section we explore the concept of *Shapley value* in the ε-MC net representation, and provide theoretical guarantees on the error incurred. The Shapley value [11] is a celebrated solution concept designed to capture the notion of fairness in CFGs. Intuitively, it grants each player i a payment ϕ_i that is proportional to her expected marginal contribution in the game. Given an MC-net representation, we can compute the Shapley values of the agents.

Proposition 1 ([8]). *The Shapley value of an agent in a marginal contribution network is equal to the sum of the Shapley values of that agent over each rule.*

[5] The threshold denotes the minimum similarity degree for two agents in order for them to be equivalent, and depends on the problem at hand. In our experimental evaluation we demonstrate how to employ specific correlation metrics to this purpose.

[6] To ensure the practicality of the algorithm for large n and m, we note that our implementation *samples* an agent in Ω_{equiv}, and checks the conditions for mutual *or* equivalent agents. Thus the complexity of our *implementation*, is $\mathcal{O}(n \cdot m^2)$.

Proposition 2. *Proposition 1 holds for the ε-MC nets as well.*

Next we compute the *Shapley values* of the agents over the rules in the ε-MC net representation. Following [8], we distinguish two cases, considering (a) *positive literals* and (b) *mixed literals* (both positive and negative).

Only Positive Literals The Shapley value of any MC-net rule $r : i \wedge Pattern \rightarrow val_r$ that contains only positive literals, is equal to $\frac{val}{m}$, where val is the value of the rule and m is the number of literals in the pattern [8]. In a ε-MC net rule r', the Shapley value of an agent i depends on whether the agents is the common literal in the rule. That is, in case i is the common literal in r' then: $\phi_{\varepsilon,i,r'} = \sum_{c \in CG_{r'}} \frac{val_{r'}}{|c|+1}$, while if i is in a pattern c within the collaborations group, then: $\phi_{\varepsilon,i,r'} = \frac{val_{r'}}{|c|+1}$.

Theorem 2. *Given an ε-MC nets representation R' with only positive literals, for any agent i, we obtain an estimate $\phi_{\varepsilon,i} = \sum_{r \in R'_i} \phi_{\varepsilon,i,r}$ of the actual Shapley value ϕ_i s.t.: $|\phi_i - \phi_{\varepsilon,i}| \leq \sum_{r \in R_i} \frac{\varepsilon}{m_r}$, where R_i and R'_i are the subsets of rules regarding agent i in the initial and the ε-MC net representation, respectively.*

Mixed Literals Inspired by [8], for rules that have mixed literals, we can consider the positive and the negative literals separately. According to [8] in a classic MC-net representation, if i is a positive literal, a rule r will apply *iff* i occurs in a given permutation after the rest of the positive literals but before any of the negative literals. Formally, let $\phi_{i,r}$ denote the Shapley value of i, p_r denote the cardinality of the positive set, and n_r denote the cardinality of the negative set, then: $\phi_{i,r} = \frac{(p_r-1)!n_r!}{(p_r+n_r)!} \cdot val_r = \frac{val_r}{p_r \cdot \binom{p_r+n_r}{n_r}}$.

Similarly to the case with only positive literals, the Shapley value of an agent i as positive literal in an ε-MC net rule depends on whether i is the common literal. That is, if i is the common literal in ε-MC net rule r' then $\phi_{\varepsilon,i,r'} = \sum_{c \in CG_{r'}} \frac{val_{r'}}{p_c \cdot \binom{p_c+n_c}{n_c}}$ where p_c denote the cardinality of the positive set in pattern $c \in CG_{r'}$, and n_c denote the cardinality of the negative set in $c \in CG_{r'}$.[7] Now if i is in pattern c within the collaborations group, then $\phi_{\varepsilon,i,r'} = \frac{val_{r'}}{p_c \cdot \binom{p_c+n_c}{n_c}}$.

For a given negative literal $\neg i$, the appearance of i in some pattern will be responsible for cancelling the application of the rule if all positive literals come before the negative literals in the ordering, and $\neg i$ is the first among the negative literals. That is, let rule $r : a \wedge b \wedge \neg i \wedge \neg j \rightarrow val$, i is responsible for canceling the application of r in any permutation of some pattern where literals a and b proceed the appearance of i, while j either appears after i or not at all. Therefore: $\phi_{\neg i,r} = \frac{p_r!(n_r-1)!}{(p_r+n_r)!} \cdot (-val_r) = \frac{-val_r}{n_r \cdot \binom{p_r+n_r}{p_r}}$.

Again, the Shapley value of an agent i as negative literal in an ε-MC net rule depends on whether i is the common literal. That is, if i is the common literal

[7] The cardinality of positive/negative sets in pattern c also considers the common literal.

in ε-MC net rule r' then $\phi_{\varepsilon,\neg i,r'} = \sum_{c\in CG_{r'}} \frac{-val_{r'}}{n_c\cdot\binom{p_c+n_c}{p_c}}$; while if i is in pattern c within the collaborations group, then $\phi_{\varepsilon,\neg i,r'} = \frac{-val_{r'}}{n_c\cdot\binom{p_c+n_c}{p_c}}$.

Theorem 3. *Given an ε-MC nets representation R' (with mixed literals), for any agent i (appearing as a positive literal i, a negative literal $\neg i$, or both) we can provide an estimate $\phi_{\varepsilon,i} = \sum_{r\in R_i'^{+}} \phi_{\varepsilon,i,r} + \sum_{r\in R_i'^{-}} \phi_{\varepsilon,\neg i,r}$ of the actual Shapley value ϕ_i such that:*

$$|\phi_i - \phi_{\varepsilon,i}| \leq \varepsilon \cdot \Big(\sum_{r\in R_i^+} \frac{1}{p_r \cdot \binom{p_r+n_r}{n_r}} + \sum_{r\in R_i^-} \frac{1}{n_r \cdot \binom{p_r+n_r}{p_r}} \Big)$$

where $R_i'^{+} \subseteq R'$ and $R_i'^{-} \subseteq R'$ are subsets of rules regarding agent i as positive or negative literal in the ε-MC net representation R'. Respectively, $R_i^+ \subseteq R$ and $R_i^- \subseteq R$ are subsets of rules regarding agent i as positive or negative literal in the initial MC net representation R.

5 Experimental Evaluation

Here we evaluate the performance of our algorithms via simulations. All experiments ran on a PC with i5@2.2GHz and 8GB of RAM. The framework was coded in Python 3.8. We used synthetic data, and the presented results are the average values over 5 sets of experiments on settings with same properties *wrt.* ε, and number of agents and rules used, as we explain immediately below.

5.1 ε-MC nets with Mutual Agents

First we present experiments performed to evaluate our approach with mutual agents (i.e., common literals), using synthetic data. We generated synthetic data with varying number of agents $n = \{100, 200, 300\}$ and rules $m = \{\frac{n\cdot(n-1)}{2}, \frac{n\cdot(n-1)}{3}, \frac{n\cdot(n-1)}{4}\}$. In each dataset, every rule consists of a pair of agents, (either as positive or negative literals, i.e we have in total $2\cdot n$ literals) randomly selected out of $\binom{n}{2}$ possible unordered pairs; and the rule's value is drawn from $\mathcal{U}(1, 200)$. For each $\langle n, m\rangle$ we generated 5 datasets, to a total of 45 datasets.

We ran our algorithm for each setting using different values of ε. We use the *reduction percentage (RP)* to measure the compactness achieved in the ε−MC nets representation by computing the number of rules comprising the new representation compared to the initial MC-net one. Formally,

$$RP = \Big(1 - \frac{\#\varepsilon\text{-MC nets rules} + \#\text{un-merged rules}}{\#\text{initial MC-net-like rules}}\Big) \cdot 100\%$$

Fig (1.a) illustrates the results of applying Alg. 1 on this set of experiments. We can see that for ε fixed across different settings, the RP achieved by our algorithm increases as the number of rules increases. Such a result is expected, since when we have more rules it is more likely to find MC-net rules that satisfy the conditions for merging, and thus the algorithm produces more compact representations. Also, for the same number of MC-net rules, as ε increases, we observe that the achieved reduction increases as well. This is due to the fact that for greater values of ε, the conditions for merging are more relaxed, and thus easier to be met. Indicatively, in settings with $n = 300$, $m = 44850$ and $\varepsilon = 2$ we get $RP = 66.5\%$, while for $n = 300, m = 44850, \varepsilon = 8$ we achieve $RP = 85.1\%$. Note that for m= $\frac{n\cdot(n-1)}{2}$ our algorithm always achieves RP greater than 46%.

5.2 Mutual vs Equivalent Agents

We now compare the performance of our approach using only mutual agents against its variant that considers equivalent agents, in terms of RP. Here we generated 75 synthetic datasets, 5 for each $\langle n, m\rangle$ combination, following the process described in the previous section; now $n = 50, 100, 200, 300$ and 400, while $m = \frac{n\cdot(n-1)}{2}, \frac{n\cdot(n-1)}{3}$, and $\frac{n\cdot(n-1)}{4}$; and again rule values are drawn from $\mathcal{U}(1, 200)$. In order to determine equivalence among agents we adopted the following scenario: agents participate in a *ridesharing setting* as drivers or commuters. First, to determine the agents' payoffs, we ran the "PK Algorithm" from [1], which computes *kernel-stable* [2,12] payments for such scenarios. Specifically, for each dataset we run the PK algorithm for a number of partitions depending on the number of agents in the dataset. Each such partition consists of a randomly sampled coalition S containing one driver (20% of agents are drivers) and 1 to 4 commuters, along with a set of singletons corresponding to the remaining agents.

In order to determine equivalent classes of agents, as soon as we have the agents payoffs received in different sampled partitions, for every i, j pair we build two ranking lists M_i and M_j respectively, as follows: *For the k^{th} sampled partition π (with $S \in \pi$): (1) if $\{i, j\} \subseteq S$, then add i's payoff according π in the k^{th} position of M_i and add j's payoff according π in the k^{th} position of M_j; otherwise (2) $i \in S$ and $\exists\ \pi'$ such that $j \in S'$ and $S \setminus \{i\} \equiv S' \setminus \{j\}$ with $S' \in \pi'$, then add i's payoff according π in the k^{th} position of M_i and add j's payoff according π' in the k^{th} position of M_j*. We then use the lists above by applying the Kendall's τ distance [9] and the the Pearson Correlation Coefficient (PCC) [4], and we consider agents i and j to be equivalent[8] if it holds that $K(M_i, M_j) \geq 0.97$ *and* $r_{M_i,M_j} \geq 0.97$. Fig (1.b) shows the results (average over 5 datasets with the same combination $\langle n, m\rangle$). We see that the algorithm that takes advantage of equivalences consistently achieves *manyfold greater reduction* than the algorithm with the mutual agents. Indicatively, for $n = 300$, $m = 44850$ and $\varepsilon = 2$ we achieve an $RP = 81.45\%$ compared to $\sim 66.5\%$ for solely mutual agents. Finally, the variant with equivalence classes achieves an RP up to 87.5% for $n = 400$, $m = 79800$, $\varepsilon = 3$. This improvement is expected, since equivalences allows us to exploit information not considered with mutual agents only.

Finally, our experiments confirm that the extra information on equivalences among agents allows us to not only produce more succinct representations, but to also learn new collaboration patterns. We show this through the NCP ratio:

$$NCP = \frac{\text{New collaboration patterns}}{\text{Total number of collaboration patterns}} \cdot 100\%,$$

[8] We consider equivalences only on positive literals.

		ε = 2	ε = 4	ε = 6	ε = 8
n = 100	m = 2475	33.35%	44.60%	51.88%	56.90%
	m = 3300	38.32%	49.60%	57.10%	62.27%
	m = 4950	46.50%	57.75%	64.74%	69.49%
n = 200	m = 9950	46.62%	57.87%	64.77%	69.46%
	m = 13267	51.97%	63.18%	69.70%	74.15%
	m = 19900	59.50%	70.26%	76.26%	80.04%
n = 300	m = 22425	54.12%	65.40%	71.85%	76.11%
	m = 29900	59.56%	70.31%	76.24%	80.01%
	m = 44850	66.56%	76.57%	81.82%	85.10%

(a)

			ε = 1	ε = 2	ε = 3
n = 50	m = 613	ε-MCMut	14.70%	21.98%	27.20%
		ε-MCEq	23.77%	32.13%	37.64%
	m = 817	ε-MCMut	18.74%	26.87%	32.45%
		ε-MCEq	28.29%	37.59%	43.47%
	m = 1225	ε-MCMut	23.08%	32.81%	39.04%
		ε-MCEq	33.09%	43.18%	49.43%
n = 100	m = 2475	ε-MCMut	24.06%	33.35%	39.65%
		ε-MCEq	43.77%	53.25%	58.77%
	m = 3300	ε-MCMut	28.99%	38.70%	45.40%
		ε-MCEq	48.41%	57.44%	62.97%
	m = 4950	ε-MCMut	35.85%	46.41%	52.88%
		ε-MCEq	54.99%	63.93%	68.79%
n = 200	m = 9950	ε-MCMut	36.09%	46.58%	52.92%
		ε-MCEq	60.28%	67.72%	72.49%
	m = 13267	ε-MCMut	41.39%	51.93%	58.27%
		ε-MCEq	63.84%	71.33%	75.58%
	m = 19900	ε-MCMut	48.78%	59.51%	65.68%
		ε-MCEq	69.20%	76.19%	80.07%
n = 300	m = 22425	ε-MCMut	43.60%	54.18%	60.46%
		ε-MCEq	67.98%	74.59%	78.19%
	m = 29900	ε-MCMut	48.74%	59.51%	65.68%
		ε-MCEq	71.29%	77.59%	80.93%
	m = 44850	ε-MCMut	55.98%	66.48%	72.46%
		ε-MCEq	75.50%	81.45%	84.52%
n = 400	m = 39900	ε-MCMut	48.89%	59.41%	65.68%
		ε-MCEq	72.53%	78.54%	81.81%
	m = 53200	ε-MCMut	54.01%	65.70%	70.65%
		ε-MCEq	75.60%	81.36%	84.40%
	m = 79800	ε-MCMut	60.92%	71.29%	76.88%
		ε-MCEq	79.49%	84.73%	87.50%

(b)

		ε = 1	ε = 2	ε = 3
n = 50	m = 613	18.95%	25.00%	28.20%
	m = 817	19.92%	27.23%	30.61%
	m = 1225	21.52%	27.73%	30.37%
n = 100	m = 2475	47.20%	56.65%	60.76%
	m = 3300	48.12%	56.24%	60.05%
	m = 4950	47.95%	54.29%	56.51%
n = 200	m = 9950	67.91%	73.57%	75.63%
	m = 13267	66.86%	71.01%	72.38%
	m = 19900	61.68%	63.41%	64.26%
n = 300	m = 22425	76.47%	79.00%	79.77%
	m = 29900	73.35%	74.75%	75.13%
	m = 44850	64.75%	65.16%	65.28%
n = 400	m = 39900	79.58%	80.66%	80.95%
	m = 53200	75.08%	75.61%	75.71%
	m = 79800	65.48%	65.68%	65.76%

(c)

Fig. 1: (a) *RP* for Alg. 1 with only Mutual Agents. (b) *RP* per setting of "Mutual vs Equivalent Agents". (c) *NCP* per setting of "Mutual vs Equivalent Agents".

where the denominator corresponds to the number of initial MC-net rules plus the new collaborative pairs of agents that our algorithm produced, exploiting equivalences among agents.[9] Fig (1.c) shows the *NCP*s for every setting when we employ the algorithm using equivalence classes of agents (averages are over the 5 different datasets for each combination $\langle n, m \rangle$). As the results show, for a given n, the *NCP* is rising as ε rises. Intuitively, since for larger ε our algorithm achieves more merges, and new collaboration patterns are discovered. Indicatively, in settings with n=400, m=39900 and ε=3, we achieve an NCP ∼80.95%.

6 Conclusions and Future Work

In this work we introduced a novel succinct representation for cooperative games. This extends the work of [8] to have rules that include sets of agents, instead of just individuals. We formally defined the $\varepsilon-$MC nets rules, merging conditions, and we proposed a polynomial algorithm for constructing such a representation. Moreover, we provided theoretical bounds regarding the Shapley value, and the absolute and relative error of the compressed representation *wrt* the initial one. Then, as envisaged by [8], we considered equivalence classes of agents, and put forward a variant of our algorithm which takes these into account, and which can generate values for collaboration patterns that were initially unknown.

[9] In case of ambiguities, we count the collaboration pattern once.

Future work will extend our algorithm to perform a *backtracking* technique. That is, merges rejected at some point may become feasible due to equivalent agents. One could also devise techniques to exploit the initial order of rules, in the spirit of heuristics used in constraint satisfaction problems; explore *machine learning* to extract the equivalent classes of agents, in terms of agents' behaviour [10] or preferences [5].

Acknowledgements. E. Streviniotis has been supported by the Onassis Foundation - Scholarship ID: G ZR 012-1/2021-2022. A. Georgara has been supported by TAILOR (H2020-952215), 2019DI17, Crowd4SDG (H2020-872944), CI-SUSTAIN (PID2019-104156GB-I00) and Enzyme Advising Group.

References

1. Bistaffa, F., Farinelli, A., Chalkiadakis, G., Ramchurn, S.: A cooperative game-theoretic approach to the social ridesharing problem. Artificial Intelligence **246**, 86–117 (2017)
2. Chalkiadakis, G., Elkind, E., Wooldridge, M.: Computational Aspects of Cooperative Game Theory. Morgan & Claypool Publishers (2011)
3. Elkind, E., Goldberg, L.A., Goldberg, P.W., Wooldridge, M.: A tractable and expressive class of marginal contribution nets and its applications. Mathematical Logic Quarterly **55**(4), 362–376 (2009)
4. Freedman, D., Pisani, R., Purves, R.: Statistics (international student edition). Pisani, R. Purves, 4th edn. WW Norton & Company, New York (2007)
5. Georgara, A., Troullinos, D., Chalkiadakis, G.: Extracting hidden preferences over partitions in hedonic cooperative games. In: 12th International Conference, KSEM 2019, Athens, Greece, August 28–30, 2019. p. 829–841. Springer-Verlag (2019)
6. Greco, G., Malizia, E., Palopoli, L., Scarcello, F.: On the complexity of core, kernel, and bargaining set. Artificial Intelligence **175**(12), 1877–1910 (2011)
7. Hirayama, K., Hanada, K., Ueda, S., Yokoo, M., Iwasaki, A.: Computing a payoff division in the least core for mc-nets coalitional games. In: Principles and Practice of Multi-Agent Systems (PRIMA'14). pp. 319–332 (2014)
8. Ieong, S., Shoham, Y.: Marginal contribution nets: A compact representation scheme for coalitional games. In: Proc. of the 6th ACM Conference on Electronic Commerce (EC'05). pp. 193–202 (01 2005)
9. Kendall, M.: Rank correlation methods. Griffin, London (1948)
10. Ogston, E., Overeinder, B., van Steen, M., Brazier, F.: A method for decentralized clustering in large multi-agent systems. In: Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems. p. 789–796. AAMAS '03, Association for Computing Machinery, New York, NY, USA (2003)
11. Shapley, L.: A Value for n-Person Games. In: Contributions to the Theory of Games II, pp. 307–317. Princeton University Press, Princeton (1953)
12. Stearns, R.E.: Convergent transfer schemes for n-person games. Transactions of the American Mathematical Society **134**(3), 449–459 (1968)
13. Streviniotis, E., Georgara, A., Chalkiadakis, G.: A succinct representation scheme for cooperative games under uncertainty. In: Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems. p. 1661–1663. AAMAS '21, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2021)