Pre-processing Techniques For Anytime Coalition Structure Generation Algorithms

Tomasz Michalak, Andrew Dowell, Peter McBurney and Michael Wooldridge

Department of Computer Science,
The University of Liverpool, UK
{tomasz, adowell, mcburney, mjw}@ liv.ac.uk

Abstract. [1] This paper is concerned with optimal coalition structure generation in multi-agent systems. For characteristic function game representations, we propose a pre-processing technique, presented in the form of filter rules, that reduces the intractability of the coalition structure generation problem by identifying coalitions which cannot belong to any optimal structure. These filter rules can be incorporated into many potential anytime coalition structure generation algorithms but we test the effectiveness of these filter rules in the sequential application of the distributed coalition value calculation algorithm (DCVC) [1] and the anytime coalition structure generation algorithm of Rahwan *et al.* (RCSG) [2]. The distributed DCVC algorithm provides an input to the centralised RCSG algorithm and we show that for both normal and uniform distributions of coalition values, the proposed filter rules reduce the size of this input by a considerable amount. For example, in a system of 20 agents, fewer than 5% of coalition values have to be input, compared to more than 90% when filter rules are not employed. Furthermore, for a normal distribution of coalition values, the running time of the RCSG algorithm exponentially accelerates as a consequence of the significantly reduced input size. This pre-processing technique bridges the gap between the distributed DCVC and centralised RCSG algorithms and is a natural benchmark to develop a distributed CSG algorithm.

1 Background and Problem Definition

In multi-agent systems (MAS), coalition formation occurs when distinct autonomous agents group together to achieve something more efficiently than they could accomplish individually. One of the main challenges in co-operative MAS is to determine which exhaustive division of agents into disjoint coalitions (referred to as a *coalition structure* (CS) from now on) maximizes the total payoff to the system. This complex research issue is referred to as the *coalition structure generation* (CSG) problem. To this end, coalition formation is often studied using *characteristic function game* (CFG) representations which consist of a set of agents A and a characteristic function v, which assigns a numerical value to every feasible coalition $C \subseteq A$, reflecting the effectiveness of the co-operation of the agents within each coalition. In this convenient but simplified representation, it is assumed that the performance of any one coalition is independent

[1] The authors are grateful for financial support received from the UK EPSRC through the project *Market-Based Control of Complex Computational Systems* (GR/T10657/01).

from other co-existing coalitions in the system. In other words, every coalition C has the same value in every structure CS to which it belongs. Evidently, the CFG representation is a special case of the more general *partition function game* (PFG) representation in which the value of any coalition depends on other co-operational arrangements between the agents in the whole system [3].

Since the number of structures increases exponentially as the number of agents increases linearly (for example, there are $190,899,322$ possible structures for 14 agents compared to 1.3 billion structures for 15 agents) then the problem of computing an optimal CS becomes time consuming, even for a moderate number of agents. Consequently, much of the research into the CSG problem has focused on generating an optimal CS by evaluating as few as possible. This line of research can be divided into three broad categories: (i) limiting the size of coalitions that can be formed [4], (ii) reducing the number of structures that need to be searched at the expense of accuracy [5, 6] and (iii) proposing algorithms which take the advantage of the CFG representation to vastly reduce the time taken to generate an optimal CS. Since we are interested in generating optimal solutions in an unconstrained domain, in this paper, we will focus exclusively on the third approach.

There are two general classes of CSG algorithms. Yen [7] proposed an algorithm based on dynamic programming (DP) methods. The advantage of this approach is that it outputs an optimal without comparing every possible structure. However, one disadvantage is that it only outputs an optimal after it has completed its entire execution meaning such methods are not appropriate when the time required to return an optimal solution is longer than the time available to the agents. To circumvent these problems, Rahwan *et al.* [2] proposed an anytime CSG algorithm which divides the space of all structures (denoted Π from now on) into sub-spaces consisting of coalition structures which are identical w.r.t. the sizes of the coalitions involved. Using this representation, and taking, as input, all feasible coalition values, this algorithm uses statistical information, computed from the coalition values input, to determine which of the sub-spaces are 'promising', *i.e.* which of them may contain an optimal structure. The algorithm then searches these 'promising' subspaces, once again, using the statistical information to avoid generating structures which cannot be optimal. This methodology exploits the fact that in CFGs, every coalition C has the same value in every structure CS to which it belongs. Since it is possible to utilize statistical data computed from coalition values to reason about the values of coalition structures, in this paper, we propose a number of pre-processing techniques. These techniques are represented in the form of filter rules which have syntax: *condition* $\rightarrow$ *action*, with the interpretation being that all coalition values which meet the requirements of the condition cannot belong to an optimal structure and so an appropriate action is performed.

Typically, such actions involve filtering coalition values from the input, or filtering all structures containing these coalitions from the search-space. Filtering coalition values from the input is important for two reasons. Firstly, it reduces the number of coalition values an individual agent needs to transfer if the coalition value calculation process is distributed as in the DCVC algorithm of Rahwan and Jennings [1]. Secondly, it automatically reduces the search space as fewer coalition structures can be created from the input. To test the effectiveness of our approach, we compare the sequential

operation of the DCVC and Rahwan *et al.* anytime algorithm both with and without the filter rules. Following the MAS literature, we focus on normal and uniform distributions of coalition values [2] and show that our filter rules:

- always significantly reduce the size of input (from 90% to 5% and 3% for normal and uniform distributions, respectively); and,
- exponentially reduce the time needed to search promising subspaces for a normal distribution whereas they do not affect the performance of the anytime CSG algorithm for a uniform distribution.

2 A pre-processing approach to solve the CSG problem

Let $A = \{1, \ldots, n\}$ be the set of all agents in the system. Since more than one structure can maximize the value of a system, the output of the CSG process may consist of a set of optimal coalition structures, denoted by $\{CS^*\}$. As of now, the CSG literature for CFG representations has exclusively focused on finding a single optimal coalition structure, denoted by $CS^* \subseteq \{CS^*\}$ [7, 2]. Usually, the choice of $CS^* \subseteq \{CS^*\}$ for $|\{CS^*\}| \geq 2$ is made in an *ad hoc* manner, *e.g.* the optimal coalition structure output is the first CS with maximal value which the algorithm encounters. However, there may be other factors which, although not displayed in the characteristic function, can be used to determine if one structure is better than the other and so we consider the $CSG(\{CS^*\})$ problem from now onward. We will denote the set of all feasible coalitions that can be created in the system by F and the corresponding set of all coalition values by $V(F)$.

The most important property of CFG representation is that, as opposed to the general PFG representation, the value of any coalition is independent from the formation of other distinct coalitions in the system. In other words, in a system of $n = |A|$ agents, for any coalition $C \subseteq A$, the value of this coalition $v(C)$ is the same in every possible structure CS where $C \in CS$. Thus, if it can be shown that the value $v(C)$ is too small for C to be in any optimal structure of the system then, clearly, any structure containing C cannot be in the optimal set and can be disregarded. Similarly, if it is proven that the combined value of a group of disjoint coalitions it too small for this group to be in any optimal structure then any structure simultaneously containing every single one of these coalitions can be disregarded. The above discussion can be summarized in the following lemma which holds under the CFG representation:

Lemma 1. *For any non-trivial*[2] *coalition C that can be divided into k disjoint sub-coalitions $C_1, \ldots, C_k$ where $C_1 \cup C_2 \ldots \cup C_k = C$; if $v(C_1) + \ldots v(C_k) > v(C)$ then $C \notin CS^*$ and so $\forall CS : C \in CS, CS \notin \{CS^*\}$.*

Proof. Consider any coalition structure CS containing coalition C. If this structure is the optimal structure (belongs to $\{CS^*\}$) then no other structure can have a value greater then this. However, if C can be divided into k sub-coalitions $C_1, \ldots, C_k$ where

[2] All coalitions of more than two agents will sometimes be referred to as *non-trivial coalitions*. In contrast, singletons, *i.e.* agents acting on their own, will sometimes be referred to as *trivial coalitions*.

$C_1 \cup C_2 \ldots \cup C_k = C$ and $|C_i| \geq 1 \; \forall i = 1, \ldots, k$ such that $v(C_1) + \ldots v(C_k) > v(C)$ then clearly the structure $CS' = CS \setminus \{C\} \cup \{C_1 \cup \ldots \cup C_k\}$ has a value greater than CS. Therefore CS, where CS is any structure containing coalition C, cannot be an optimal structure and $\forall CS : C \in CS, CS$ does not belong to $\{CS^*\}$.

For example, in a system of 5 agents $A = \{a_1, a_2, a_3, a_4, a_5\}$ if the value of the coalition $\{a_1, a_2\}$ is less than the sum of the values of the individual coalitions $\{a_1\}$ and $\{a_2\}$ then it is clear that any structure $CS_a = \{a_1, a_2\} \cup CS'$, where CS' is a specific structure of the agents $\{a_3, a_4, a_5\}$, must have value less than the structure $CS_b = \{a_1\}, \{a_2\} \cup CS'$. Consequently, any structure CS which contains coalition $\{a_1, a_2\}$ cannot be optimal and all such structures can be disregarded.

As mentioned earlier, most existing CSG algorithms take advantage of this characteristic. For example, in the anytime CSG algorithm of Rahwan *et al.* described in Section 1 some subspaces of coalition structures are pruned away from the search space before the search process has begun. Using statistical information obtained from the input of coalition values, it is *a priori* decided if certain types of coalition structures cannot be optimal. Similarly, in the process of searching promising subspaces, certain search directions are *a priori* identified as those that cannot lead to an improved outcome. Examples include dynamic programming (DP) CSG algorithms [7] in which it is evaluated whether a decomposition a coalition C into exactly two smaller coalitions of all the agents in C would be profitable. In this spirit, the improved dynamic programming (IDP) algorithm presented in [8] considers more dissociations than just the dissociation of a coalition into two disjoint coalitions. This approach, which is yet another application of Lemma 1, turns out to be more efficient in terms of time and memory cost than the conventional DP algorithms.

Consider a system of 5 agents where a value of a non-trivial coalition $C := \{a_1, a_2, a_3, a_4\}$ is 7. In order to prove that C cannot be in an optimal CS, it is not always necessary to show that any partition of this coalition has a combined value greater than 7. It may also be sufficient to show that the values of a strict subset of k disjoint coalitions as in Lemma 1 are greater than the value of $v(C)$, for instance, it may be the case that $v(\{a_1\}) + v(\{a_3, a_4\}) \geq v(\{a_1, a_2, a_3, a_4\})$. Following this intuition, we can relax some of the assumptions in Lemma 1 and propose Lemma 2:

Lemma 2. *For any non-trivial coalition C that can be divided into k disjoint sub-coalitions $C_1, \ldots, C_k$ where $C_1 \cup C_2 \ldots \cup C_k = C$; if $\sum_{i=1}^{j} v(C_i) > v(C)$ where $j < k$ then $C \notin CS^*$ and so $\forall CS : C \in CS$, CS does not belong to $\{CS^*\}$.*

Theoretically, every feasible non-trivial coalition could be decomposed into all possible combinations of sub-coalitions, however, this is a completely inefficient approach. After all, such a decomposition of the grand coalition yields Π. In the remainder of this paper, we will show that an appropriate application of both Lemmas 1 and 2 may still considerably speed up the CSG process in the state-of-the-art CSG algorithm.

Firstly, we will extend Lemma 1 so that it can be applied not only to a particular coalition but to collections of coalitions which have been grouped together w.r.t. some criteria. One natural criterion to group coalitions is size. Let all coalitions of the same size be grouped together in $|A|$ sets $\mathcal{C}_1, \ldots, \mathcal{C}_i, \ldots, \mathcal{C}_{|A|}$ where $\mathcal{C}_i$ denotes the set of all coalitions of size i. For example, for $A = \{1, \ldots, 5\}$ there will be $|A| = 5$ sets

$\mathcal{C}_1, \ldots, \mathcal{C}_5$ where $\mathcal{C}_1$ contains all coalitions of size 1, $\mathcal{C}_2$ all coalitions of size 2, *etc*. Additionally, in this example, suppose that the coalitions with smallest values in $\mathcal{C}_1$ and $\mathcal{C}_2$ are $\{a_1\}$ and $\{a_2, a_3\}$, respectively. This means that any decomposition of coalition of size 3 will not have a smaller value than $v(\{a_1\}) + v(\{a_2, a_3\})$. Consequently, if any coalition from $\mathcal{C}_3$ has a value smaller than $v(\{1\}) + v(\{2, 3\})$ then we can disregard this coalition as, following Lemma 1, it cannot be in an optimal structure. We extend Lemma 1 as follows:

Lemma 3. *Let $\mathcal{C}_i$ denote the (complete) set of all coalitions of size i. For any set $Z_i \subseteq \mathcal{C}_i$ of coalitions of size i and for a particular integer partition p of i, $i_1, \ldots, i_k$ such that $i_1 + \ldots + i_k = i$, if the sum of the lowest coalition values in sets $\mathcal{C}_{i_1}, \ldots, \mathcal{C}_{i_k}$ (denoted $d_i(p)$) is strictly greater than the maximum value in set Z_i then no coalition from set Z_i can be in an optimal structure. More formally, if $d_i(p) := \sum_{i=1}^{k} \min \mathcal{C}_{i_k} > \max Z_i$ then $\forall CS : C \in CS$ and $C \in Z_i$ it holds that $CS \notin \{CS^*\}$.*

Proof. Suppose that coalition C_i is the coalition with the biggest value in set $Z_i \subseteq \mathcal{C}_i$ and coalitions $C_{i_l}, \ldots, C_{i_k}$ are the coalitions with the smallest value in lists $i_1, \ldots, i_k$ respectively. Now consider any coalition structure CS which contains coalition C_i. If CS is an optimal structure no other structure can have value greater than this. Now, consider structure $CS' = CS \setminus \{C_i\} \cup \{C_{i_1} \cup \ldots \cup C_{i_k}\}$ where $C_{i_1}, \ldots, C_{i_k}$ are all disjoint and $C_{i_1} \cup \ldots \cup C_{i_k} = C_i$. If the sum of the smallest values in sets $\mathcal{C}_{i_l}, \ldots, \mathcal{C}_{i_k}$ is greater than the biggest value in set $Z_i \subseteq \mathcal{C}_i$ then clearly for all coalitions $C_{i_l}, \ldots, C_{i_k}$ in $\mathcal{C}_{i_l}, \ldots, \mathcal{C}_{i_k}$ respectively, and any $C_i \in Z_i \subseteq \mathcal{C}_i$, $v(C_{i_1}) + \ldots + v(C_{i_k}) > v(C_i)$ and so $v(CS') > v(CS)$. Therefore, following Theorem 1, for any partition of i, $i_1, \ldots, i_k$ such that $i_1 + \ldots + i_k = i$, if the sum of the minimum values in sets $i_1, \ldots, i_k$ is greater than the maximum value in set $Z_i \subseteq \mathcal{C}_i$ then no coalition in $Z_i \subseteq \mathcal{C}_i$ can be in an optimal coalition structure and so no CS which contains a coalition in $Z_i \subseteq \mathcal{C}_i$ can belong to $\{CS^*\}$.

For any set containing non-trivial coalitions of size i, it is possible to compute the set of all integer partitions of value i. Such a set is denoted $P(i)$. Furthermore, for each $p' \in P(i)$ it is possible to compute a value $d_i(p')$ as in Lemma 3. Now, following the same lemma, we can compare every coalition C of size i with $d_i(p')$ and immediately disregard those for which $v(C) < d_i(p')$. In fact, there is no need to apply Lemma 3 to all partitions in $P(i)$ but only to the partition $p'' \in P(i)$ such that $d_i(p'')$ is maximal in $P(i)$. Clearly, for a coalition C of size i, if it holds that $v(C) < d_i(p')$ then it also holds that $v(C) < d_i(p') \leq d_i(p'')$. Such a maximal value will be referred to as the *domination value* of coalitions of size i. More formally:

Definition 1. *For any set containing coalitions of size i and every partition $p \in P(i)$,* domination value $\tilde{d}_i$ *is the highest value of $d_i(p)$ for all $p \in P$, or $\tilde{d}_i = \max_{p \in P} d_i(p)$.*

Following Lemma 1, if the value of any coalition C of size i is less than the domination value $\tilde{d}_i$ then there exists a dissociation of C with a greater value than $v(C)$. In such a case, C cannot be in any optimal structure. This property is exploited by the filter rules presented in the next section.

3 Filter Rules

In this section, we propose filter rules that can be applied both while calculating the values of all the coalitions in $\mathcal{F}$ (thus, generating $V(\mathcal{F})$) and while searching through the (sub-)spaces of coalition structures. We will refer to coalitions which cannot belong to any optimal structure as *not-promising*. All the other coalitions will be called *promising*. We denote both of these disjoint sets by $\mathcal{F}_{np} \subseteq \mathcal{F}$ and $\mathcal{F}_p \subseteq \mathcal{F}$, respectively. Initially, before the filter rules are applied, it is assumed that all coalitions are promising, *i.e.* $\mathcal{F}_p = \mathcal{F}$. Furthermore, we will refer to a subset of coalition values $Z \subseteq V(\mathcal{F}_p)$ ($Z \subseteq V(\mathcal{F}_{np})$) as promising (not promising).

3.1 Filter rules for input calculation

While calculating all coalition values in the input, Lemma 2 can be applied to highlight those coalitions which are not promising. However, dissociating every coalition into all potential sub-coalition combinations is usually inefficient for coalitions of greater size. It should be left to a system designer to decide into how many sub-coalitions a currently computed coalition should be disaggregated. The natural and possibly most efficient choice is to consider the singleton partition of a coalition, *i.e.* the decomposition of the coalition into all of the individual agents who participate in this coalition. Clearly, in the process of computing the value of a given coalition, all agents who co-operate in this coalition must be known and such a disaggregation can be easily performed. Therefore, following Lemma 2 we propose the first filter rule:

FR1 For any non-trivial coalition C that can be divided into k disjoint singleton sub-coalitions $C_1, \ldots, C_k$ where $C_1 \cup C_2 \ldots \cup C_k = C$ and $\forall i = 1, \ldots, k\ |C_i| = 1$; if it is the case that either (i) $v(C_1) + \ldots v(C_k) > v(C)$ or (ii) $\sum_{i=1}^{j} v(C_i) > v(C)$ where $j < k$ then all such coalitions $C \in \mathcal{F}_{np}$.

Naturally, we can relax the constraint that $|C_i| = 1$ to $|C_i| \leq s$ where $1 < s < |A|$ depending on into how many partitions the system designer wishes to divide coalition C. The computational cost of applying **FR1** should be balanced against potential gains.

Example 1. Consider a 5 agent system $A = \{a_1, a_2, a_3, a_4, a_5\}$ with the following coalition values: $v(C) = 14$ for $C = A$; $\forall |C| = 1\ v(C) = 3$; $\forall |C| = 2\ v(C) = 5$; $\forall |C| = 3\ v(C) = 10$;, and $\forall |C| = 4\ v(C) = 12$. Observe that the sum of any two coalitions of size 1 is strictly greater than the value of any coalition of size 2. Furthermore, the value of the grand coalition is smaller than the value of a non-cooperative coalition structure. Thus, **FR1** filters out all coalitions of size two as well as the grand coalition. However, following Lemma 1 the system designer may consider all dissociations of a coalition into exactly two disjoint sub-coalitions in the same way as in the DP algorithm presented in [7]. Since the value of any coalition of size 4 is smaller than the value of any coalition of size 1 added to the value of a disjoint coalition of size 3, then coalitions of size 4 are not promising.

Now, consider the domination value. Lemma 3 immediately yields the following filter rule:

FR2 For any subset $Z_s \subseteq \mathcal{C}_s$ of coalitions of size s, if $\tilde{d}_s > \max Z_s$ then all coalition from Z_s are not promising. More formally: $\forall Z_s \subseteq \mathcal{C}_s$, if $\tilde{d}_s > \max Z_s$ then $Z_s \in \digamma_{np}$ and $Z_s \notin \digamma_{p.}$

FR2 can be applied by a system designer to every coalition C of size i immediately after $v(C)$ has been calculated (*i.e.* in the same manner as **FR1**). Alternatively, if $|Z_i| \geq 2$ then the value of $\max Z_i$ can be recorded while calculating coalition values in this subset and **FR2** can be applied after this process has been finished. We illustrate the functioning of **FR2** with the following example:

Example 2. Suppose the coalition values for four agents $A = \{a_1, a_2, a_3, a_4\}$ are as follows: $\forall |C| = 1, v(C) \in \langle 4, 7\rangle, \forall |C| = 2, v(C) \in \langle 5, 7\rangle, \forall |C| = 3, v(C) \in \langle 7, 11.5\rangle$ and $v(A) = 11$.[3] The domination values for the coalitions of size 2 to 4 are computed as follows: $\tilde{d}_2 = \min S_1 + \min S_1 = 4 + 4 = 8$, $\tilde{d}_3, = \max\{3 \times \min S_1, \min S_1 + \min S_2\} = 12$, $\tilde{d}_4, = \max\{\ 4 \times \min S_1, 2 \times \min S_1 + \min S_3, 2 \times \min S_2\ \} = 16$, Observe that both $\tilde{d}_2 > \max S_2$ and $\tilde{d}_4 > \max S_2 = v(A)$. Intuitively, this means that for every coalition of size 2 and 4, there exists a dissociation of this coalition into sub-coalitions which have value greater than the value of the coalition and so following previous reasoning, no coalition of size 2 or 4 can be in an optimal structure and no structure containing these coalitions can be in an optimal set.

3.2 Filter rules for search of coalition structure space

As mentioned in the Introduction, the anytime algorithm of Rahwan *et al*. divides Π into sub-spaces containing structures which are identical w.r.t. size of the coalitions involved. In a system of $|A|$ agents let $S^* := \{m_1, \ldots, m_k\}$ denote the (currently) most promising subspace, where $m_1, \ldots, m_k$ represent sizes of the coalition involved ($\sum_{i=1}^{k} m_i = |A|$ and $k \geq 2$). To search S^*, the values of all the coalition structures belonging to this sub-space should be computed unless it is proven beforehand that they cannot belong to $\{CS^*\}$.[4] The general form of a coalition structure in S^* is $\{C_1, \ldots, C_k\}$ such that all of $C_1, \ldots, C_k$ are disjoint and $\forall i = 1, \ldots, k\ C_i \in \mathcal{C}_{m_i}$. Let CS^*_N denote the coalition structure with the highest value found thus far. Rahwan *et al.* propose a filter rule that, based on the statistical information gathered about the coalition value input, avoids those structures which cannot be optimal. For $k \geq 3$ and $k - 1 \geq l \geq 1$ it holds that:

B&B If $\sum_{i=1}^{l} v(C_i) + \sum_{i=l+1}^{k} \max \mathcal{C}_{m_i} \leq v(CS^*_N)$ then no structures in S^* can be optimal, to which simultaneously belong all of $C_1, \ldots, C_l$.

This filter rule ensures that, for a particular structure under consideration $\{C_1, \ldots, C_k\}$, if the combined value of first $1 \leq l \leq n-1$ coalitions ($\sum_{i=1}^{l} v(C_i)$) plus the value of the sum of maximum values of coalitions in the remaining sets $\mathcal{C}_{m_{l+1}}, ..., \mathcal{C}_{m_k}$ is less

[3] For example, the notation $v(C) \in \langle 4, 7\rangle$ means that $v(C)$ can be any real value higher than or equal to 4 and lower than or equal to 7 and that the minimal value of all such coalitions is 4 and the maximal is 7.

[4] Or unless it is proven that an optimal coalition structure in this sub-space has been found.

than the current optimum value $v(CS^*_N)$ then no structures to which all of $C_1, \ldots, C_l$ simultaneously belong will be considered in the optimal CSG process.[5]

Example 3. For $|A| = 9$ agents let $S^* := \{1, 2, 2, 2, 2\}$ be the (currently) most promising subspace, $CS^*_N = 28$, $v(\{a_1\}) + v(\{a_2, a_3\}) = 9$, $v(\{a_4, a_5\}) = 4$, $v(\{a_4, a_6\}) = 6$ and $\max \mathcal{C}_2 = 7$. Following **B&B**, since $v(\{a_1\}) + v(\{a_2, a_3\}) + v(\{a_4, a_5\}) + \max \mathcal{C}_2 + \max \mathcal{C}_2 < CS^*_N$ then no structures in S^* can be optimal, to which simultaneously belong all of $\{1\}, \{2, 3\}$ and $\{4, 5\}$. These structures are: $\{\{a_1\}, \{a_2, a_3\}, \{a_4, a_5\}, \{a_6, a_7\}, \{a_8, a_9\}\}$, $\{\{a_1\}, \{a_2, a_3\}, \{a_4, a_5\}, \{a_6, a_8\}, \{a_7, a_9\}\}$, and $\{\{a_1\}, \{a_2, a_3\}, \{a_4, a_5\}, \{a_6, a_9\}, \{a_7, a_8\}\}$. Thus, in this example branch-and-bound rule saves on calculation time by avoiding calculations that lead to three structures which cannot be optimal. Considering $\{a_4, a_6\}$, since $v(\{a_1\}) + v(\{a_2, a_3\}) + v(\{a_4, a_6\}) + \max \mathcal{C}_2 + \max \mathcal{C}_2 > CS^*_N$ then condition in **B&B** does not hold.

The above branch-and-bound technique is based on basic statistical information collected about $\mathcal{C}_m$, namely the maximum value of coalitions in this set ($\max \mathcal{C}_m$). Assuming that this information is known about some subsets $Z_m \subseteq \mathcal{C}_m$ for $m = m_1, \ldots, m_k$ then the filter rule above can be generalized as follows:

FR3 If $\sum_{i=1}^{k} \max Z_{m_i} < v(CS^*_N)$, where $k \geq 2$, then no structures in S^* can be optimal, in which simultaneously $C_1 \in Z_{m_1}, \ldots$ and $C_k \in Z_{m_k}$.[6]

Example 4. In the system of 9 agents let $S^* := \{1, 2, 2, 4\}$ be the (currently) most promising subspace, $CS^*_N = 25$, $\max\{Z_1 := \{\{a_1\}, \{a_2\}, \{a_3\}\}\} = 4$, $\max\{Z_2 := \{\{a_2, a_3\}, \{a_4, a_5\}, \{a_6, a_7\}, \{a_8, a_9\}, \{a_1, a_3\}, \{a_1, a_2\}\}\} = 6$ and $\max\{Z_4 := \mathcal{C}_4\} = 7$. Following **FR3**, since $\max Z_1 + 2 \times \max Z_2 + \max Z_4 < CS^*_N$ then no structures in S^* containing all of C_1, C_2, C_3, C_4 such that $C_1 \in Z_1$, $C_2 \in Z_2$, $C_3 \in Z_2$, and $C_4 \in Z_4$ can be optimal. In this example **FR3** saves on calculation time by avoiding $\{\{a_1\}, \{a_2, a_3\}, \{a_4, a_5\}, \{a_6, a_7, a_8, a_9\}\}$, $\{\{a_2\}, \{a_1, a_3\}, \{a_4, a_5\}, \{a_6, a_7, a_8, a_9\}\}$, and $\{\{a_1, a_2\}, \{a_3\}, \{a_4, a_5\}, \{a_6, a_7, a_8, a_9\}\}$.Note that there are certain combinations of $C_1 \in Z_1, C_2 \in Z_2, C_3 \in Z_2, C_4 \in Z_4$ for which $\{C_1, C_2, C_3, C_4\}$ is not a coalition structure. For instance, any combination $\{\{a_1\}, \{a_2, a_3\}, \{a_4, a_5\}, \{a_1, a_2, a_3, a_4\}\}$ is neither exhaustive nor disjoint. Although, we are not interested in such combinations, it can be observed that the inequality in **FR3** holds for them as well.

4 Application

To test the effectiveness of the filter rules we employ them in the state-of-the-art distributed coalition value calculation algorithm of Rahwan and Jennings [1] (DCVC from

[5] Note that Rahwan *et al.* focused on CSG(CS^*) problem so that in their branch-and-bound filter rule there is weak inequality sign '$\leq$'. To use this filter rule in the solution of CSG($\{CS^*\}$) problem the sign '$<$' should be assumed. This holds for the other inequality conditions in this paper.

[6] Note that certain combination of $C_1 \in Z_{m_1}, \ldots, C_k \in Z_{m_k}$ are neither disjoint nor exhaustive, *i.e.* they are not proper coalition structures as assumed in this paper. We leave them aside as not being relevant to our analysis. See Example 4.

now on) and in the state-of-the-art anytime CSG algorithm of Rahwan *et al.* [2] (RCSG from now on).

Apart from the decentralised design, the key advantage of DCVC is that calculated coalition values in $V(F)$ are ordered in a unique way ensuring that only $V(F)$ and not F must be kept in memory. Coalition values structured in the same way as in DCVC are used as an input to RCSG. However, it is not trivial to connect both algorithms as the former one is distributed whereas the latter one is centralised. This means that, at the moment when calculations in DCVC are complete, every agent knows only a fraction of $V(F)$ needed in RCSG. Thus, a resource-consuming data transfer has to take place.

An application of any filter rules always requires to balance computational costs with potential gains. We will show that filter rules **FR1**, **FR2** and **FR3** can be applied at a relatively low computational cost, and using them results in:

1. A substantial decrease the number of coalitions to be transferred; and
2. An increase the efficiency of the promising structure sub-space search in RCSG.

4.1 Application of FR1, FR2 and FR3 in DCVC

In DCVC, the space of all coalitions is represented as a set of lists $L_1, \ldots, L_{|A|}$ where list L_i contains all coalitions of size i. Within each list, the coalitions are ordered w.r.t. the agents they are made of. This ordering ensures that the agent composition of every coalition in each list can be derived from the place in the list it occupies. In terms of the notation introduced in Section 2 we may write $L_i = \overrightarrow{\mathcal{C}_i}$, *i.e.* every list is an ordered set of the values in $\mathcal{C}_i$. Such lists for a system of 6 agents $a_1, a_2, ..., a_6$, all with equal computational capabilities, are presented in Figure 1.[7]

All lists are divided into disjoint segments proportional to the agents' computational capabilities. Usually every agent is assigned two segments, one in the upper and one in the lower part of the list. We will denote both segments assigned to agent a_i in list L_m by $\mathcal{L}^U_m(a_i)$ and $\mathcal{L}^L_m(a_i)$, respectively. As not all the list are exactly divisible by $|A|$, variable α, depicted in Figure 1, is used to distribute 'left over' coalitions as equally as possible between agents. 'Left over' coalition values assigned to agent a_i will be denoted by $\mathcal{L}^O_m(a_i)$. Overall, the above methods to distribute $V(F)$ have the advantage that all agent computations are finished (almost) simultaneously, even when some coalitions require more arithmetic operations than others and when agents have different processing capabilities. The allocation process is described in detail in [1].

After the allocation of segments has been performed, agents sequentially compute values in lists starting from L_1. Since there are $|A|$ coalitions in L_1, each agents is assigned exactly one value from this list to calculate. Let us assume that every agent transmits this value to all the other agents in the system. By doing so, agents are ready to apply **FR1** while computing coalition values in L_2. Additionally, they are able to efficiently compute the value of a structure consisting of every promising coalition C and singletons, *i.e.* $\{C, \{j\} : j \in A \backslash C\}$.

[7] Note that in Figure 1 numbers in lists are a shorthand notation representing agents. It should be emphasized that every list contains coalition values from $V(F)$ and not coalitions from F themselves. Thus, the representation of lists $L_1, ..., L_{|A|}$ in Figure 1 should be read not as coalitions of agents from F but as their numerical values from $V(F)$.

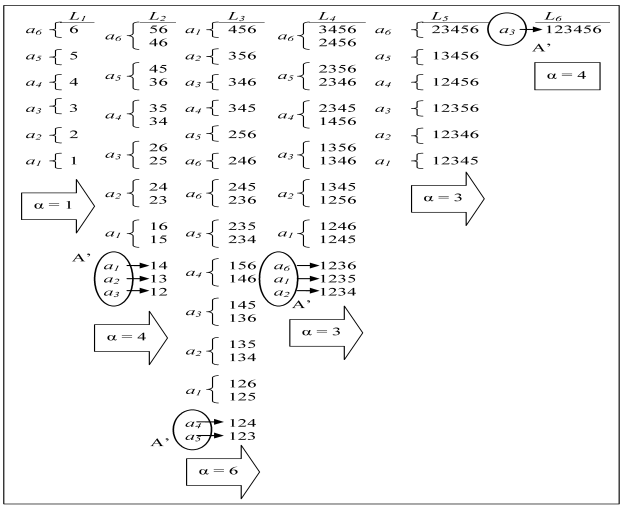

Figure 1: Division of $V(F)$ in a system of 6 agents

For example, in Figure 1, while computing the values of coalitions $\{a_4, a_5\}$ and $\{a_3, a_6\}$ in $\mathcal{L}_2^U(a_5)$, agent a_5 applies **FR1** and compares $v(\{a_4\}) + v(\{a_5\})$ with $v(\{a_4, a_5\})$ and $v(\{a_3\})+v(\{a_6\})$ with $v(\{a_4, a_6\})$. Furthermore, this agent calculates the values of $\{\{a_1\}, \{a_2\}, \{a_3\}, \{a_4, a_5\}, \{a_6\}\}$ and $\{\{a_1\}, \{a_2\}, \{a_3, a_6\}, \{a_4\}, \{a_5\}\}$. Let $CS_N^*(a_i)$ denote a coalition structure with the highest value that agent a_i found in its assigned segments.

In DCVC, when the computational capabilities of all the agents are symmetric, each of them receives an (almost) equal fraction of F to calculate $V(F)$ (as in Figure 1). The total number of values assigned to every agent is equal to either $\lfloor (2^{|A|} - 1) / |A| \rfloor$ or $\lfloor (2^{|A|} - 1) / |A| \rfloor + 1$, depending on the distribution of 'left-over' coalitions. For example, for 20 agents this amounts to either 52428 or 52429 and for 30 agents to either 53687091 or 53687092. Clearly, transferring such numbers of coalition values from DCVC to RCSG is a potential bottleneck in the system performance. The application of **FR1** helps to reduce the transfer load but its effectiveness depends on the values of the singletons. In some systems, these values might be lower than a value of any coalition. In such a case, following Lemma 2, **FR1** can be extended to consider other partitions of a coalition C whose value is being computed. However, dissociations other than into singletons may considerably increase computational cost and should be weighted against potential gains. Consequently, we propose another approach that can significantly reduce the transfer load. Since RCSG searches only through promising subspaces of Π, then only some lists of coalition values have to be transferred. This means that the procedure to evaluate whether a given subspace is promising should be removed from the RCSG and incorporated into DCVC.

In RCSG a subspace S is considered to be promising if its upper bound $UB_S := \sum_{\forall m_i \in S} maxL_{m_i}$ is no smaller than the lower bound of the entire system $LB := \max\{v(CS_N^*), \max\{Avg_S\}\}$, where $Avg_S = \sum_{\forall m_i \in S} avgL_{m_i}$ is proven to be the average value of every subspace. In DCVC, where calculations are distributed among all

of the agents, statistics such as maximum, minimum and average coalition values cannot be computed without an exchange of information among agents. However, if agents record maximum, minimum and average values in every segment they are assigned to calculate then after all calculations are (almost simultaneously) finished every agent a_i can broadcast this statistical data along with $CS^*_N(a_i)$. Using this knowledge, the agents can calculate, for every list L_m, the values of $maxL_m$, $avgL_m$, and $minL_m$. Both the maximum and average values are then used, by the agents, to determine which sub-spaces are promising. Consequently, by exchanging some basic information about segments the agents can prune the search space at the end of DCVC in the same way as the centre would do it in RCSG.

The subspace with the maximum upper bound is the most promising, *i.e.* $S^* = \arg\max_S UB_S$, and is searched first. It might happen that the unique CS^* (*i.e.* when $|\{CS^*\}| = 1$) is found in this sub-space. Thus, in order to reduce potential transfer load, we assume that only lists needed to construct coalition structures in S^* are initially passed on from DCVC to RCSG.

In the process of transmitting the lists which are required to generate all structures in S^*, both **FR2** and a version of **FR3** are applied. Both filter rules can reduce the transfer load even further. Before the transmission begins, agents calculate the domination value $\tilde{d}_m$ for every list $L_m : m \in S^*$ using the relevant statistics.[8] Segments assigned to agent a_i which meet the condition $\max \mathcal{L}_m^{U/L/O}(a_i) < \tilde{d}_m$ are filtered out as not promising. Furthermore, for all segments in which $\max \mathcal{L}_m^{U/L/O}(a_i) \geq \tilde{d}_m$, it is determined if individual coalition values within these segments, not filtered by **FR1** are not smaller than $\tilde{d}_m$. Individual coalitions with such values become not promising. The version of **FR2** that concerns segments will be denoted as **FR2a**, whereas, the version that concerns individual coalitions as **FR2b.**

Although **FR3** has been constructed to increase efficiency of searching Π, a version of this filter rule can also be applied before data transfer. Let the most promising subspace be $S^* := \{m_1, \ldots, m_k\}$. Since every agent a_1 knows $\max \mathcal{L}_m^{U/L/O}(a_i)$ and $\max L_m$ for all $m = m_1, ..., m_k$ the following version of **FR3** (denoted as **FR3a**) can be applied to decide whether either $\mathcal{L}^U_{m_j}(a_i)$, $\mathcal{L}^L_{m_j}(a_i)$ or $\mathcal{L}^O_{m_j}(a_i)$, where $m_j \in S$, can be filtered out as not promising:

$$\sum_{m \in S \setminus \{m_j\}} \max L_m + \max \mathcal{L}_{m_j}^{U/L/O}(a_i) < v(CS^*_N). \tag{1}$$

If the segment cannot be filtered out as not promising then the above filter rule can be applied to every individual coalition C within this segment not filtered out by **FR1** or **FR2a/b**. In such a situation, $\max \mathcal{L}_{m_j}^{U/L/O}(a_i)$ in formula (1) can be replaced with $v(C)$, where $v(C) \in \mathcal{L}_{m_j}^{U/L/O}(a_i)$. and this version of **FR3** is denoted **FR3b**. Finally,

[8] We leave it to the discretion of the system designer as to the exact architecture of domination value calculations. At the end of DCVC, agents can either calculate all needed domination values individually or distribute calculations among themselves in such a way that the most efficient agents calculate domination values for the lists of coalitions with the highest cardinality and *vice versa*.

after all the filter rules have been applied, the agents transfer only those values which are promising.

To summarize, in order to reduce the transfer load from DCVC to RCSG we propose the following extension of the former algorithm:

Step 1: Agents exchange among themselves values of singleton coalitions in L_1. While calculating value of any non-trivial coalition C they: (1.1) apply **FR1**; (1.2) compute $CS^*_N(a_i)$ and (1.3) Update and store the statistics about their segments $\mathcal{L}_m^{U/L/O}$;

Step 2: After calculating the values in all of their segments, agents exchange among themselves the segments' statistics and $CS^*_N(a_i)$. Using this information, they determine the most promising sub-space S^* and domination values $\tilde{d}_m$ for every list $L_m : m \in S^*$ that needs to be transmitted;

Step 3: Just before transmission takes place each agent applies both **FR2a** and **FR3a** to segments $\mathcal{L}_m^{U/L/O}$, where $m \in S^*$. If a segment is not filtered out then **FR2b** and **FR3b** are applied again to individual coalitions within this segment that have not been filtered out by **FR1**. Only the values of promising coalitions are transferred.[9]

Finally, it should be observed that even if the coalition value calculation algorithm was not distributed, then **FR1**, **FR2b** and **FR3b** can be still be applied to determine the not promising coalitions. In the simulations we will show that it considerably reduces the running time of RCSG.

4.2 Application of FR3 in RCSG

The strength of RCSG in searching the promising sub-spaces is that it avoids creating both invalid and repeated coalition structures as described in [9]. Suppose that $S^* := \{m_1, \ldots, m_k\}$ is the most promising subspace in a system of $|A|$ agents and that the value of the particular partial structure $\{C_{m_1}, ..., C_{m_l}\}$, where $l < k$, has already been calculated. Let $\overrightarrow{A'}$ be the ordered set of agents which do not belong to any coalition in this partial structure, *i.e.* $\overrightarrow{A'} := \overrightarrow{A} \backslash \{C_{m_1}, ..., C_{m_l}\}$. The method proposed by Rahwan *et al.* in [9] cycles through all $m_{l+1}-$element combination from $\overrightarrow{A'}$ to construct all feasible coalitions $C_{m_{l+1}}$. The cycle is designed in such a way that all $m_{l+1}-$combinations which must contain the agent in $\overrightarrow{A'}$ with the lowest index are considered first and all $m_{l+1}-$combinations which must contain the second agent in $\overrightarrow{A'}$ but not the first one are considered in the next step and so on. As explained in Subsection 3.2, after constructing every $C_{m_{l+1}}$ RCSG applies **B&B** filter rule to decide whether any partial structure $\{C_{m_1}, ..., C_{m_l}, C_{m_{l+1}}\}$ is promising. A version of **FR3** can be used to identify groups rather than only individual partial structures. To this end, agents in DCVC have to gather additional information about some particular segments in lists $L_1, ..., L_{|A|}$. Let $\overrightarrow{Z}_m(i)$ denote a segment of list L_m such that a_i is the first agent in every coalition in $\overrightarrow{Z}_m(i) \subseteq L_m$. For example, referring to Figure

[9] To indicate the position of each promising coalition value and maintain the list structure we propose to transmit a characteristic bit vector alongside the stream of values. In such a vector 1 indicates value of a promising coalition and 0 a not promising one.

1, $\overrightarrow{Z}_3(2)$ is $\{\{2,5,6\},\{2,4,6\},\{2,4,5\},\{2,3,6\},\{2,3,5\},\{2,3,4\}\}$. Assume that, while calculating coalition values in DCVC, the agents record $\max \overrightarrow{Z}_m(i)$ for every combination of $m = 1, ..., |A|$ and $a_i \in A$. In such a case, before cycling through all $m_{l+1}-$combinations which contain the agent in $\overrightarrow{A'}$ with the lowest index, the following version of **FR3** (denoted as **FR3c**) can be applied, by a centre in RCSG. If

$$\sum_{i=1}^{l} v(C_i) + \max \overrightarrow{Z}_{l+1}(i) + \sum_{i=l+2}^{k} \max L_{m_i} < v(CS_N^*),$$

then any partial structure $\{C_{m_1}, ..., C_{m_l}, C_{m_{l+1}}\}$ such that $C_{m_{l+1}} \in \overrightarrow{Z}_{l+1}$ is not promising. **FR3c** is a generalization of RCSG **B&B** as it is applicable to groups rather than individual coalitions.

4.3 Numerical Simulations

In the numerical simulations we compare the sequential execution of DCVC and RCSG with and without filter rules. The following assumptions are imposed: (i) the calculation of any particular coalition values in DCVC takes no time; and (ii) any data transfers are instantaneous. We focus on the percentage of $V(F)$ which does not have to be transmitted from DCVC to RCSG due to applied filter rules. Additionally, we demonstrate that, for a normal distribution of coalition values, filter rules considerably improve the performance of the CSG process in RCSG.

Following [2, 5], we evaluate the algorithms under two different assumptions as to the probability distributions of coalition values:[10] (ND) Normal: $v(\mathbf{C}) = max(0, |C| \times p)$, where $p \in N(\mu = 1, \sigma = 0.1)$; and ($UD$) Uniform: $v(\mathbf{C}) = max(0, |C| \times p)$, where $p \in U(a, b)$, where $a = 0$ and $b = 1$. For a system of $|A| = 11 \ldots 20$ agents, we ran both versions of the algorithms 25 times and reported the results within a 95% confidence interval. The algorithms were implemented in MATLAB.

The results are presented in Table 1. Column 2 shows the number of all coalition values $|F|$ and Column 3 percentage of $V(F)$ that needs to be transmitted from DCVC to RCSG without filter rules. Column 4 shows the actual percentage of $V(F)$ transferred when filter rules are applied. Column 5 contains the size of the characteristic bit vector (as a percentage of $V(F)$) that must be transferred simultaneously with the values in Column 4 (see Footnote 8). The next five columns present the individual contribution of filter rules **FR1**, **FR2a**, **FR2b**, **FR3a** and **FR3b** to the reduction of the overall transfer load between Columns 3 and 4. The last column shows the running time of RCSG based on the unfiltered input and without **FR3c** divided by the running time of this algorithm based on filtered input and with **FR3c**.

[10] We omit the sub- and super-additive cases as their solution is straightforward.

Normal Distribution										
1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.
|A|	|V(F)|	without FRs	with FRs	c. bit vector	FR1	FR2a	FR2b	FR3a	FR3b	Relative Time
11	2047	91±4.5	8.68±1.69	5.61	53.12	0	0.01	21.61	25.26	2.75±0.81
12	4095	91±2.4	8.43±1.51	5.55	55.71	0	0.01	18.13	26.15	4.17±1.55
13	8191	90±5.5	5.29±1.37	5.5	54.91	0	0	16.69	28.4	6.61±2.03
14	16383	91±5.3	6.05±1.61	5.7	54.99	0	0	16.7	28.31	16.51±5.0
15	32767	93±8.4	5.04±1.21	5.9	53.11	0	0	14.7	32.19	17.88±6.04
16	65535	91±5.1	4.48±1.14	4.48±1.14	49.79	0	0	14.04	36.17	23.21±7.4
17	131071	92±4.5	4.29±1.01	4.29±1.01	53	0	0	11.19	35.81	112.2±44
18	262143	93±6.3	3.69±0.89	3.69±0.89	52.47	0	0	9.01	38.52	169±54
19	524287	94±6.1	3.12±0.81	3.12±0.81	51.48	0	0	7.8	40.72	197±67
20	1048575	91±5.3	2.49±0.67	2.49±0.67	50.03	0	0	5.99	43.98	380±88
Uniform Distribution										
|A|	|V(F)|	without FRs	with FRs	c. bit vector	FR1	FR2a	FR2b	FR3a	FR3b	Relative Time
11	2047	51.31±16.51	3.32±1.79	3.32±1.79	49.24	0.07	1.17	35.21	14.31	1.12±0.14
12	4095	39.99±18.01	2.89±1.41	2.89±1.41	51.3	0.06	1	28.01	19.63	1.22±0.09
13	8191	49.68±17.52	2.71±1.02	2.71±1.02	49.43	0.12	0.68	27.83	21.94	1.34±0.18
14	16383	41.13±18.80	2.42±0.68	2.42±0.68	50.32	0.04	0.32	23.15	26.17	1.14±0.12
15	32767	48.52±21.00	2.35±0.48	2.35±0.48	51.12	0	0.11	18.51	30.26	1.23±0.16
16	65535	52.98±19.40	1.26±0.33	1.26±0.33	50.51	0	0.04	16.54	32.91	1.14±0.09
17	131071	50.41±18.79	1.15±0.29	1.15±0.29	51.81	0	0.02	11.19	36.98	1.01±0.03
18	262143	46.41±19.70	1.13±0.28	1.13±0.28	50.18	0	0	10.91	38.91	1.03±0.02
19	524287	41.71±18.42	1.05±0.13	1.05±0.13	50.02	0	0	8.01	41.97	1.04±0.04
20	1048575	32.80±19.70	1.04±0.1	1.04±0.1	50.25	0	0	7.46	42.29	1.19±0.11

Table 1: Simulation results (all values, except for columns 1,2, and 11 are expressed in $\%$)

Consider ND first. Without filter rules, about 90% of $V(F)$ needs to be transmitted from DCVC to RCSG (Column 3). Filter rules reduce this number to about $9\% + 5.5\%$ for $n = 11$ and to around $2.5\%+2.5\%$ for $n = 20$ (Columns 4 and 5).[11] **FR1** is the most successful in filtering coalitions accounting for about 50% of them. Consequently, both **FR2a** and **FR2b**, based on the same assumption, cannot offer any significant improvement upon this result. However, if the singleton coalitions have relatively low values, then **FR1** would not perform so well, and **FR2a/b** would become more profitable. A decreasing percentage of coalition values are ruled out by rule **FR3a**. The intuition behind the decreasing effectiveness of this filter rule is as follows. The higher the value of $|A|$, the longer the segments become. Consequently, there is a greater probability that the (randomly-drawn) extreme values in these segments are similar to each other. This reduces the effectiveness of filter rules based on segments' maxima. In contrast, **FR3b**, which focuses on the particular coalition values, has increasing success. We expect that for $n > 25$ only **FR1** and **FR3b** should be used. In conclusion, the combination of the filtered input and the application of **FR3c** in RCSG results in an exponentially faster performance of this algorithm.

For the UD case, only about 50% of $V(F)$ needs to be transmitted from DCVC to RCSG (Column 3) but filter rules are able to reduce this number: from about $2 \times$

[11] All simulations are performed in MATLAB. In the future, the same simulations will be reprogrammed in JAVA, which is a more popular platform. JAVA is also faster than MATLAB but it should not influence the results presented in this paper which are based on relative comparison. Note that for $n > 15$, it is more efficient to transfer the exact location of every coalition (in the form of an integer) that does not satisfy the filter rules rather than transfer the characteristic bit function since the former method requires less data.

$(3.32\% \pm 1.79\%)$ for $n = 11$ to around $2 \times (1.04\% + 0.1\%)$ for $n = 20$. Analysis of the effectiveness of individual filter rules is similar to the ND case. However, in contrast to ND, the combination of the filtered input and the application of **FR3c** does not significantly outperform the standard RCSG. This is caused by the very nature of the uniform-distribution. Intuitively, since coalition values in any lists are dispersed neither reduced input nor **FR3c** perform much better than **B&B** in standard RCSG. However, our filter rules still prove its effectiveness by achieving large reduction in the transfer load.

5 Conclusions

In this paper we have discussed a number of techniques designed to increase the efficiency of CSG algorithms. In particular, we developed filter rules which can, for CFG representations, eliminate a significant proportion of not promising coalitions from the space of all coalition values. Such a (structured) space of coalition values acts as input to, among others, the state-of-the-art CSG algorithm of Rahwan *et al*. Although this algorithm has been demonstrated to be very effective, its particular drawback is its centralised nature. This is especially challenging as there exist already a very efficient algorithm for distributed coalition value calculation; this means that after the DCVC algorithm has been employed, all of the agents have to transmit all the values they have computed to a single entity. In this paper, we demonstrated that our proposed filter rules are extremely effective in reducing the size of this coalition value input to the CSG algorithm. Consequently, we showed how to efficiently bridge the gap between the decentralised DCVC and the centralised RCSG. A natural follow up to our work, which we plan to explore, is to develop a distributed CSG algorithm that can be relatively easily constructed from the analysis in this paper.

References

1. Rahwan, T., Jennings, N.: An algorithm for distributing coalitional value calculations among cooperating agents. Artificial Inteligence **(8-9)**(171) (2007) 535–567
2. Rahwan, T., Ramchurn, S., Dang, V., Giovannucci, A., Jennings, N.: Anytime optimal coalition structure generation. In: Proceedings of AAAI, Vancouver, Canada (2007) 1184–1190
3. Michalak, T., Dowell, A., McBurney, P., Wooldridge, M.: Optimal coalition structure generation in partition function games. In: Proceedings of ECAI 2008, Patras, Greece (2008)
4. Shehory, O., Kraus, S.: Methods for task allocation via agent coalition formation. Artificial Intelligence **1**(101) (1998) 165–200
5. Sandholm, T., Larson, K., Andersson, M., Shehory, O., Tohme, F.: Coalition structure generation with worst case guarantees. Artificial Intelligence **1-2**(111) (1999) 209–238
6. Dang, V., Jennings, N.: Generating coalition structures with finite bound from the optimal guarantees. In: Proceedings of AAMAS, New York, USA (2004)
7. Yeh, D.Y.: A dynamic programming approach to the complete. BIT **4**(26) (1986) 467–474
8. Rahwan, T., Jennings, N.: An improved dynamic programming algorithm for coalition structure generation. In: Proceedings of AAMAS,, Estoril, Portugal, (2008)
9. Rahwan, T., Ramchurn, S., Dang, V., Giovannucci, A., Jennings, N.: Near-optimal anytime coalition structure generation. In: Proceedings of IJCAI, Hyderabad, India (2007)