Generating coalition structures with finite bound from the optimal guarantees[*]

Viet Dung Dang and Nicholas R. Jennings
School of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ, UK.
{vdd00r,nrj}@ecs.soton.ac.uk

Abstract

The coalition formation process, in which a number of independent, autonomous agents come together to act as a collective, is an important form of interaction in multi-agent systems. When effective, such coalitions can improve the performance of the individual agents and/or of the system as a whole. However, one of the main problems that hinders the wide spread adoption of coalition formation technologies is the computational complexity of coalition structure generation. That is, once a group of agents has been identified, how can it be partitioned in order to maximise the social payoff? This problem has been shown to be NP-hard and even finding a sub-optimal solution requires searching an exponential number of solutions. Against this background, this paper reports on a novel anytime algorithm for coalition structure generation that produces solutions that are within a finite bound from the optimal. Our algorithm is benchmarked against Sandholm et al.'s algorithm [8] (the only other known algorithm for this task that can also establish a worst-case bound from the optimal) and is shown to be up to 10^{379} *times faster (for systems containing* 1000 *agents) when small bounds from the optimal are desirable.*

1. INTRODUCTION

The coming together of a number of distinct, autonomous agents in order to act as a coherent grouping is an important form of interaction in multi-agent systems. It has been advocated in e-commerce (where buyers may pool their requirements in order to obtain bigger group discounts [11]), in grid computing (where multi-institution virtual organisations are viewed as being central to coordinated resource sharing and problem solving [2]), and in e-business (where agile groupings of agents need to be formed in order to satisfy particular market niches [5]). In all of these cases, the formation of coalitions aims to increase the agents' abilities to satisfy goals and to maximise their personal or the system's outcomes.

In this context, the coalition formation process can be viewed as being composed of three main activities [8]:

1. *Coalition structure generation*: forming coalitions of agents such that those within a coalition coordinate their activities, but those in different coalitions do not. This primarily involves partitioning the set of all agents in the system into exhaustive and disjoint coalitions.[1] Such a partition is called a **coalition structure**. For example, in a multi-agent system composed of three agents $\{a_1, a_2, a_3\}$, there exist seven possible coalitions: $\{a_1\}$, $\{a_2\}$, $\{a_3\}$, $\{a_1, a_2\}$, $\{a_1, a_3\}$, $\{a_2, a_3\}$, $\{a_1, a_2, a_3\}$
   and five possible coalition structures: $\{\{a_1, a_2, a_3\}\}$, $\{\{a_1\}, \{a_2, a_3\}\}$, $\{\{a_2\}, \{a_3, a_1\}\}$, $\{\{a_3\}, \{a_1, a_2\}\}$, $\{\{a_1\}, \{a_2\}, \{a_3\}\}$.
2. *Optimising the value of each coalition*: pooling the resources and tasks of the agents in a given coalition to maximise the coalition value. For example, given the coalition structure $\{\{a_1\}, \{a_2, a_3\}\}$, each of the two coalitions $\{a_1\}$ and $\{a_2, a_3\}$ will try to optimise its value.
3. *Payoff distribution*: dividing each coalition's value among its members. For example, if the coalition $\{a_2, a_3\}$ produces a payoff of X then this value

* This work is funded by a BT Exact studentship.

1 Some research also considers non-disjoint coalitions (see section 5 for details).

needs to be divided between a_2 and a_3 according to some scheme (e.g. equality or stability).

Although these activities are distinct and, in a sense, conceptually sequential, it is also clear that they interact. For example, in a competitive environment, the coalition that an agent wants to join depends on the payoff that it is likely to receive (activities 1 and 3). However, in cooperative environments, where the agents work together to maximise the social welfare, payoff distribution is less important, and coalition structure generation that maximises the social welfare is the dominant concern.

To date, most work on coalition formation in game theory and multi-agent systems has tended to focus on payoff distribution (see section 5 for more details), with comparatively little attention paid to the other two activities. However we believe that coalition structure generation is perhaps more fundamental because it provides a means of measuring the global good of the coalition formation process (by computing the global good of the coalition structures). Given this fact, the focus of this paper is on coalition structure generation.

Classically, game theoretic work on coalition formation considers only super-additive environments (meaning any two disjoint coalitions are better off by merging together) [6]. In such cases, coalition structure generation is trivial because all agents are better off by forming the grand coalition (i.e. the coalition that contains all the agents). Moreover, it has even been argued that almost all environments are super-additive because, at worst, the agents in the composite coalition can use solutions as if they are in separate coalitions [6]. However, this assumption is not valid for many real-world problems, because of the cost of forming coalitions and the cost of coordination between members in the same coalition.

In non-super-additive environments, coalition structure generation is a major concern (because of the exponential size of the set of all possible coalition structures). In such cases, the desirable goal is usually to maximise the social welfare. However, it has been shown that this problem is NP-hard and, moreover, even finding a sub-optimal solution requires searching an exponential number of solutions [8]. To tackle this problem, several researchers have proposed algorithms for coalition structure generation (see section 5 for details). However, most of the existing algorithms cannot establish a worst-case bound from the optimal. This is clearly undesirable because it means the solutions they generate can be arbitrarily bad. To overcome this drawback, Sandholm et al. [8] developed an anytime algorithm that can establish a worst-case bound (until our algorithm it was the only one that could do this). However, as their algorithm's computational complexity is exponential, it is desirable to see if its complexity can be reduced in order to make it useable in practical applications.

Against this background, this paper advances the state-of-the-art in the following ways. First, we develop a new coalition structure generation algorithm which is anytime and with which we can establish a worst-case bound from the optimal. Second, while its running time is necessarily exponential, we show that our algorithm is significantly faster than Sandholm et al.'s. Specifically, when computing small bounds from the optimal, our algorithm is shown to be up to 10^7 times faster for small systems (50 agents), 10^{23} times faster for medium systems (100 agents) and 10^{379} times faster for large systems (1000 agents). With larger bounds from the optimal, we show that there is no significant difference in performance between our algorithm and Sandholm et al.'s.

The remainder of the paper is organised as follows. Section 2 formalises the problem of coalition structure generation. Section 3 presents our algorithm. We then evaluate the performance of our algorithm in section 4. Section 5 discuss related work, and, finally, section 6 concludes the paper and presents future work.

2. COALITION STRUCTURE GENERATION IN CHARACTERISTIC FUNCTION GAMES

This section formalises the problem of coalition structure generation. Let A be the set of agents, and n be the number of agents in A (i.e., $|A| = n$). As is common practice in the literature (e.g. [3] [6] [8] [9]), we consider coalition formation in *characteristic function games (CFGs).* In such settings, there is a value $v(S)$ for each and every subset S of A, known as *the value of coalition* S, which is the utility that members of S can jointly attain. Fundamentally, this means each coalition's value is independent of the actions of agents that are not members of the coalition. Although, in general, the value of a coalition may depend on non-members' actions, CFGs can be applied in many real-world multi-agent problems [8].

As in [8], we assume that every coalition's value is non-negative:

$$v(S) \geq 0, \forall S \subseteq A \tag{1}$$

This assumption is not very restrictive, because if there exist some negative coalitional values, and if all coalitional values are bound from below (i.e., they are not infinitely negative), they can always be normalised by subtracting from each of them a value $\min_{S \subseteq A} v(S)$.

The algorithm proceeds as follows:

- Step 1: Search through the sets L_1, L_2, L_n
- From step 2 onward, search, consequently, through the sets $SL(n, \lceil n(q-1)/q \rceil)$ with q running from $\lfloor \frac{n+1}{4} \rfloor$ down to 2.

  That is, search $SL(n, \lceil n(\lfloor n/4 \rfloor - 1)/\lfloor n/4 \rfloor \rceil$ at step 2, search $SL(n, \lceil n(\lfloor n/4 \rfloor) - 2)/(\lfloor n/4 \rfloor \rceil - 1)$ at step 3 and so on.

  Moreover, from step 3 onward, as $SL(n, \lceil nq/(q+1) \rceil) \subseteq SL(n, \lceil n(q-1)/q \rceil)$ (it is easy to see that $SL(n, \lceil n(a-1)/a \rceil) \subseteq SL(n, \lceil n(b-1)/b \rceil)$ for every $a > b$) we only have to search through the set $SL(n, \lceil n(q-1)/q \rceil) \setminus SL(n, \lceil nq/(q+1) \rceil)$ in order to search through the set $SL(n, \lceil n(q-1)/q \rceil)$.
- At each step return the coalition structure with the biggest value (i.e. best social welfare) so far.

Figure 1. The coalition structure generation algorithm.

A coalition structure CS is a partition of A into disjoint, exhaustive coalitions. That is, each agent belongs to exactly one coalition. The *value* of a coalition structure, $V(CS)$, is expressed in terms of its social welfare. That is:

$$V(CS) = \sum_{S \in CS} v(S) \qquad (2)$$

Also, we define the *size* of a coalition structure as the number of coalitions that it contains and L as the set of all coalition structures.

Given the above terms, the problem of coalition structure generation is then to find a coalition structure CS^* that maximises the social welfare. That is:

$$CS^* = argmax_{cs \in L} V(CS) \qquad (3)$$

However, the problem of coalition structure generation is computationally complex. Sandholm [8] showed that the number of coalition structures (i.e. $|L|$) is exponential, specifically, $O(n^n)$ and $\omega(n^{n/2})$, and that the problem is NP-hard. Moreover, he showed that for any algorithm to establish any bound from the optimal, it must search at least 2^{n-1} coalition structures.

To this end, the next section presents our algorithm for coalition structure generation. In this paper, we aim to establish a bound from the optimal and so, necessarily, our algorithm is not polynomial.

3. THE ALGORITHM

In this section, we present our algorithm for coalition structure generation and prove that the solution it generates is within a finite bound from the optimal.

To this end, let L_k be the set of all coalition structures with size k. Thus we have:

$$L = \bigcup_{k=1}^{n} L_k$$

The number of coalition structures in L_k is $S(n,k)$, widely known in Mathematics as *the Stirling number of the Second Kind* [7]. The value of $S(n,k)$ can be computed by the following formula [7]:

$$S(n,k) = \frac{1}{k!} \sum_{i=0}^{k-1} (-1)^i \binom{k}{i} (k-i)^n$$

Definition 1 *Let $SL(n,k,c)$ be the set of all coalition structures that have exactly k coalitions and at least one coalition whose cardinality is not less than c.*

Definition 2 *Let $SL(n,c)$ be the set of all coalition structures whose cardinality is between 3 and $n-1$ that have at least one coalition whose cardinality is not less than c. That is:*[2]

$$SL(n,c) = \bigcup_{k=3}^{n-1} SL(n,k,c)$$

With these definitions in place, we can now express our algorithm for solving the problem (see Figure 1). Basically, at first it searches all the coalition structures that have one, two or n coalitions (i.e. all the coalition structures in the sets: L_1, L_2 and L_n) (as Sandholm et al.'s algorithm does). But after that, instead of searching through the sets L_k (for $3 \le k \le n-1$) one by one (as Sandholm et al. do), our algorithm only searches some specific subsets of L_k (see Figure 2 for a diagramatic representation). In particular, it searches the set of all coalition structures that have k coalitions and at least one coalition whose cardinality is not less than $\lceil n(q-1)/q \rceil$ (with q running from $\lfloor \frac{n+1}{4} \rfloor$ down to 2 as in Figure 1). Note that we start from $q = \lfloor \frac{n+1}{4} \rfloor$ because Sandholm et al. showed that, after searching L_1, L_2, L_n, the algorithm can establish a bound $b = \lceil n/2 \rceil$

2 In fact, as $SL(n,k,c) = \emptyset$ for all $n - c + 1 < k \le n - 1$, this formula can be rewritten as: $SL(n,c) = \bigcup_{k=3}^{n-c+1} SL(n,k,c)$

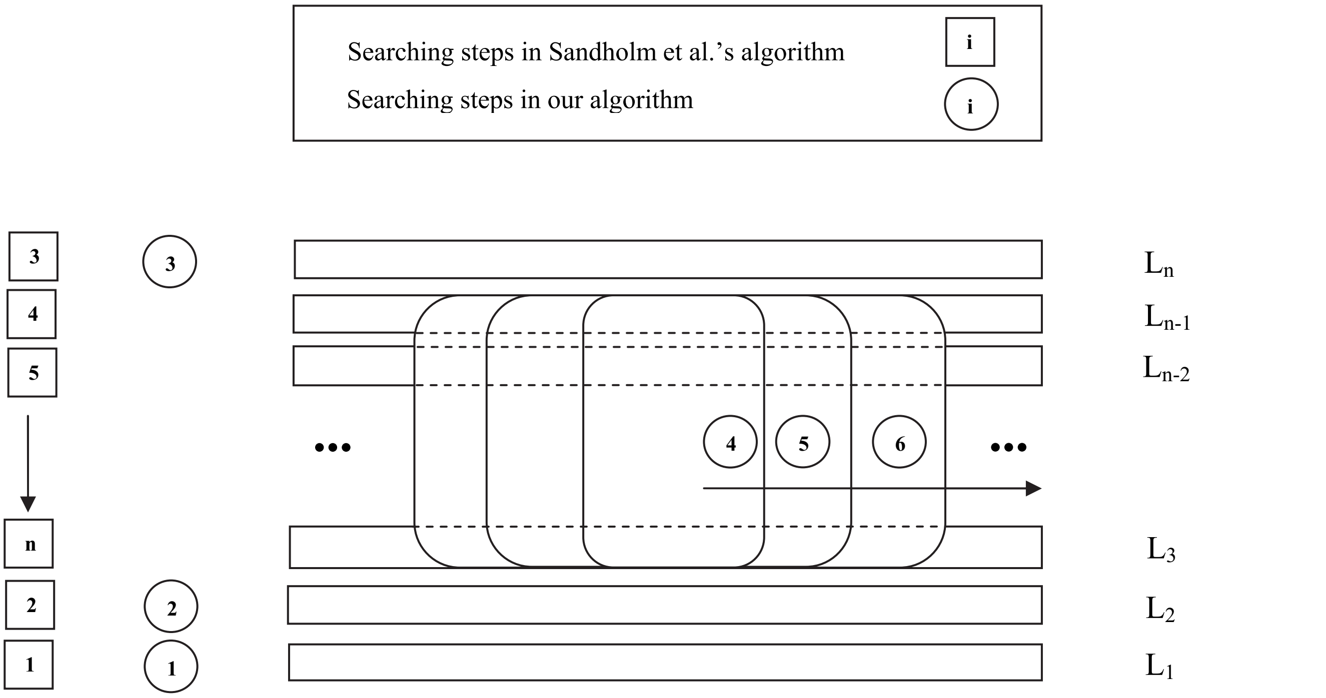

Figure 2. Comparison of the searching paths between our algorithm and Sandholm et al.'s.

and, later in the paper, we will show that after searching $SL(n, \lceil n(q-1)/q \rceil)$, our algorithm can establish a bound $b = 2q - 1$. Thus, we start from the biggest q such that $2q - 1 < \lceil n/2 \rceil$ or $q = \lfloor \frac{n+1}{4} \rfloor$.

The next step is to show that the solution generated by the algorithm is within a bound from the optimal and that the bound is reduced further after each round. Thus ours is an **anytime algorithm**: it can be interrupted at any time and the bound keeps improving with an increase in execution time.[3]

Theorem 1 *Immediately after finishing searching $SL(n, \lceil n(q-1)/q \rceil)$, the solution generated by our algorithm is within a finite bound $b = 2q - 1$ from the optimal.*

Proof. Let CS^a be the coalition structure that our algorithm generates. Let CS^* be an optimal coalition structure. Assume CS^* contains t coalitions $C_1, C_2, ..., C_t$. We have to prove:

$$\frac{V(CS^*)}{V(CS^a)} \leq 2q - 1 \tag{4}$$

For the cases where t equals 1, 2 or n, the proof is trivial, as CS^a will also be an optimal coalition structure. Thus $V(CS^a) = V(CS^*)$, so $\frac{V(CS^*)}{V(CS^a)} = 1 \leq 2q - 1$.

3 If the domain happens to be super-additive, the algorithm finds the optimal coalition structure (grand coalition) immediately.

Now we only have to prove for the case where $3 \leq t \leq n - 1$. Without loss of generality, we can assume the cardinalities of the sets $C_1, C_2, ..., C_t$ are in decreasing order. That is:

$$|C_1| \geq |C_2| \geq ... \geq |C_t| \tag{5}$$

For the convenience of presentation, we assume $C_i = \emptyset$ and $v(C_i) = 0$ for every $i > t$.

First, we will show that for every coalition $C \subseteq A$:

$$v(C) \leq V(CS^a)$$

Considering the coalition structure $CS_0 = \{C, A \setminus C\}$. As $CS_0 \in L_2$, we have:

$$\begin{aligned} & V(CS_0) \leq V(CS^a) \\ & \Rightarrow v(C) + v(A \setminus C) \leq V(CS^a) \\ & \Rightarrow v(C) \leq V(CS^a) \text{ (because of assumption (1))} \end{aligned}$$

Thus we have:

$$\begin{aligned} & v(C_i) \leq V(CS^a) \text{ ,} \forall 1 \leq i \leq q-1 \\ & \Rightarrow \sum_{i=1}^{q-1} v(C_i) \leq (q-1) \cdot V(CS^a) \end{aligned} \tag{6}$$

Now let us consider the other coalitions of CS^*, namely, $C_q, C_{q+1}, ..., C_t$.

Considering the following coalition structure:

$$CS_1 = \{C_q, C_{2q}, ..., C_{\lfloor \frac{t}{q} \rfloor q}, D\}$$

That is:

$$D = A \setminus \cup_{i=1}^{\lfloor \frac{t}{q} \rfloor} C_{iq} \tag{7}$$

Let us analyse the cardinality of coalition D. We have for all $1 \leq i \leq \lfloor \frac{t}{q} \rfloor$:

$$\begin{aligned}
&|C_{(i-1)q+1}| \geq |C_{(i-1)q+2}| \geq ... \geq |C_{iq}| \\
&\text{(because of (5))} \\
&\Rightarrow q|C_{iq}| \leq \sum_{j=1}^{q} |C_{(i-1)q+j}| \\
&\Rightarrow q \sum_{i=1}^{\lfloor \frac{t}{q} \rfloor} |C_{iq}| \leq \sum_{i=1}^{\lfloor \frac{t}{q} \rfloor} \sum_{j=1}^{q} |C_{(i-1)q+j}| \\
&\Rightarrow q \sum_{i=1}^{\lfloor \frac{t}{q} \rfloor} |C_{iq}| \leq |C_1| + |C_2| + ... + |C_{\lfloor \frac{t}{q} \rfloor q}| \\
&\Rightarrow q \sum_{i=1}^{\lfloor \frac{t}{q} \rfloor} |C_{iq}| \leq |C_1| + |C_2| + ... + |C_t| = n \\
&\Rightarrow \sum_{i=1}^{\lfloor \frac{t}{q} \rfloor} |C_{iq}| \leq n/q
\end{aligned}$$

As $|D| = n - \sum_{i=1}^{\lfloor \frac{t}{q} \rfloor} |C_{iq}|$ (from (7)), we then have:

$$\begin{aligned}
&|D| \geq n - n/q \\
&\Rightarrow |D| \geq n(q-1)/q \\
&\Rightarrow |D| \geq \lceil n(q-1)/q \rceil \\
&\Rightarrow CS_1 \in SL(n, \lceil n(q-1)/q \rceil) \\
&\Rightarrow V(CS_1) \leq V(CS^a) \\
&\Rightarrow v(C_q) + v(C_{2q}) + ... + v(C_{\lfloor \frac{t}{q} \rfloor q}) + v(D) \\
&\quad \leq V(CS^a) \\
&\Rightarrow v(C_q) + v(C_{2q}) + ... + v(C_{\lfloor \frac{t}{q} \rfloor q}) \leq V(CS^a) \\
&\text{(as } v(D) \geq 0\text{, by assumption (1))}
\end{aligned}$$

For all $1 \leq j \leq q-1$, by proving similarly to the above, we have:

$$v(C_{q+j}) + v(C_{2q+j}) + ... + v(C_{\lfloor \frac{t}{q} \rfloor q+j}) \leq V(CS^a)$$

Thus for all $0 \leq j \leq q-1$, we have:

$$\begin{aligned}
&\sum_{i=1}^{\lfloor \frac{t}{q} \rfloor} v(C_{iq+j}) \leq V(CS^a) \\
&\Rightarrow \sum_{j=0}^{q-1} \sum_{i=1}^{\lfloor \frac{t}{q} \rfloor} v(C_{iq+j}) \leq q \cdot V(CS^a) \\
&\Rightarrow \sum_{i=q}^{\lfloor \frac{t}{q} \rfloor q+q-1} v(C_i) \leq q \cdot V(CS^a)
\end{aligned} \tag{8}$$

Also:

$$\begin{aligned}
&\lfloor \frac{t}{q} \rfloor q + q - 1 > (\frac{t}{q} - 1)q + q - 1 \\
&\Rightarrow \lfloor \frac{t}{q} \rfloor q + q - 1 > t - 1 \\
&\Rightarrow \lfloor \frac{t}{q} \rfloor q + q - 1 \geq t
\end{aligned} \tag{9}$$

Thus, from (8) and (9), we have:

$$\sum_{i=q}^{t} v(C_i) \leq q \cdot V(CS^a) \tag{10}$$

From (6) and (10) we have:

$$\sum_{i=1}^{t} v(C_i) \leq (2q-1) \cdot V(CS^a) \tag{11}$$

□

From theorem 1, we can also see that the bound decreases after each round, because $2q-1$ decreases as q decreases. Thus our algorithm is an anytime one.

Having presented our algorithm for coalition structure generation, the next section compares it with Sandholm et al.'s.

4. PERFORMANCE EVALUATION

To evaluate the effectiveness of our algorithm we compare it against Sandholm et al.'s [8] since this is the only other known algorithm with worst-case bounds. In more detail, Sandholm et al.'s algorithm operates as described in Figure 3. Basically, it first searches all the coalition structures that have 1 or 2 coalitions, then it continues to search all the coalition structures that have n, $n-1$, ..., 3 coalitions (in that order). Sandholm et al. then prove that after having completed searching L_k, the solution the algorithm generates is within a bound b', where $b' = \lceil \frac{n}{h} \rceil$ if $n \equiv h-1 (mod\ h)$ and $n \equiv k (mod\ 2)$, or $b' = \lfloor \frac{n}{h} \rfloor$ otherwise ($h = \lfloor \frac{n-k}{2} \rfloor + 2$).

To evaluate the performance of our algorithm, we compare it with Sandholm et al.'s on a worst-case basis. That is, we compare the size of the search space of the two algorithms (i.e. the number of coalition structures each algorithm has to search) in order to establish the same bound from the optimal.

To calculate the number of coalition structures that our algorithm needs to search, we present the following formula.

Lemma 1 *For all $n > k \geq 3$ and $c \geq \lceil n/2 \rceil$, the cardinality of $SL(n,k,c)$ can be calculated as follows:*

$$|SL(n,k,c)| = \sum_{i=c}^{n-k+1} S(n-i, k-1) \cdot \frac{n!}{i!(n-i)!} \tag{12}$$

- Search the bottom two levels of the coalition structure graph (Note that level k of the coalition structure graph in Sandholm et al.'s algorithm corresponds exactly to the set L_k in ours).
- Continue with a breadth-first search from the top of the graph as long as there is time left, or until the entire graph has been searched (this occurs when this breadth-first search completes level 3 of the graph, i.e., depth $n-3$)
- Return the coalition structure that has the highest welfare among those seen so far.

Figure 3. Sandholm et al.'s algorithm.

PROOF. For each i such that $c \leq i \leq n-k+1$, let $T(n,k,i) \subseteq SL(n,k,c)$ be the set of all coalition structures that have exactly k coalitions and at least one coalition whose cardinality equals i. As for every coalition structure CS in $SL(n,k,c)$, any coalition in CS has at most $n-k+1$ agents (because $(k-1)$ other coalitions in CS must contain at least $(k-1)$ agents), we have:

$$SL(n,k,c) = \bigcup_{i=c}^{n-k+1} T(n,k,i)$$

Now we will show that $T(n,k,i_1) \bigcap T(n,k,i_2) = \emptyset$ for every $i_1 \neq i_2$, $i_1 \geq c$ and $i_2 \geq c$. This can be proved by contradiction. Suppose there exist i'_1 and i'_2 such that $T(n,k,i'_1) \bigcap T(n,k,i'_2) \neq \emptyset$. This means there exists a coalition structure CS' such that: $CS' \in T(n,k,i'_1) \bigcap T(n,k,i'_2)$. Now $CS' \in T(n,k,i'_1)$ means it has at least one coalition whose cardinality equals i'_1, and, similarly, $CS' \in T(n,k,i'_2)$ means it has at least one coalition whose cardinality equals i'_2. Moreover, CS' has at least 3 coalitions (as $k \geq 3$), so the number of agents in CS' will be greater or equal than $i'_1 + i'_2 + 1$. Thus:

$$\begin{aligned} & n \geq i'_1 + i'_2 + 1 \\ & \Rightarrow n \geq c + c + 1 = 2c + 1 \\ & \Rightarrow n \geq 2\lceil n/2 \rceil + 1 \\ & \Rightarrow n \geq 2n/2 + 1 \\ & \Rightarrow n \geq n + 1 \end{aligned}$$

As we reach contradiction, we must have:

$$T(n,k,i_1) \bigcap T(n,k,i_2) = \emptyset$$

for every $i_1 \neq i_2$, $i_1 \geq c$ and $i_2 \geq c$. Thus we have:

$$\Rightarrow |SL(n,k,c)| = \sum_{i=c}^{n-k+1} |T(n,k,i)| \tag{13}$$

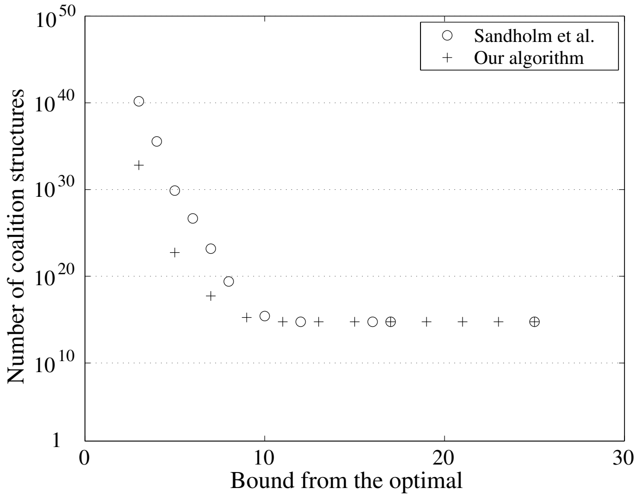

Figure 4. The case $n = 50$.

Now let us consider a coalition structure $CS' \in T(n,k,i)$. As one of the coalitions in CS' has exactly i agents, $k-1$ other coalitions in CS' must have exactly $n-i$ agents. Also, the number of ways to choose an i-agent set from n agents is $\frac{n!}{i!(n-i)!}$. Thus the number of coalition structures in $T(n,k,i)$ equals the number of coalition structures that have exactly $k-1$ coalitions in a multi-agent system with $n-i$ agents multiplied with $\frac{n!}{i!(n-i)!}$:

$$|T(n,k,i)| = S(n-i,k-1) \cdot \frac{n!}{i!(n-i)!} \tag{14}$$

From (13) and (14) we have:

$$|SL(n,k,c)| = \sum_{i=c}^{n-k+1} S(n-i,k-1) \cdot \frac{n!}{i!(n-i)!}$$

□

With this in place, we test the algorithms with the number of agents $n = 50$, 100, 500, and 1000.[4] The result of the tests are presented in the following graphs.[5] As we are calculating the number of coalition structures each algorithm has to search in order to establish a bound from the optimal, the smaller the number of coalition structures the better.

4 We observe similar patterns with other values of n varying from 50 to 1000.

5 The big bounds are not shown in the graphs (that is, bounds greater than 30 in the case $n = 100$, greater than 40 in the case $n = 500$ and greater than 50 in the case $n = 1000$), because the results for the two algorithms are nearly the same for these bounds.

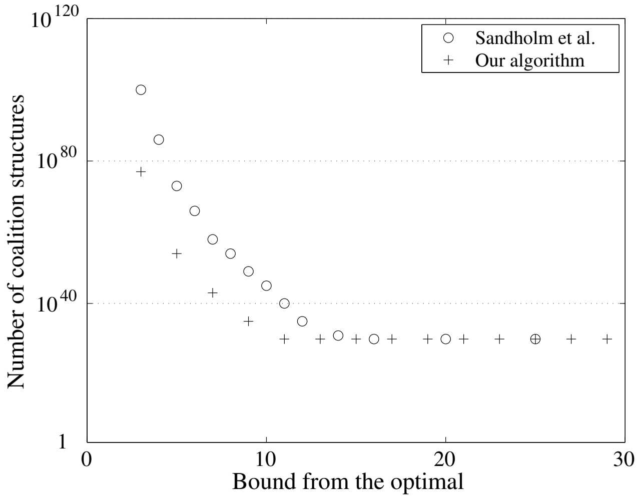

Figure 5. The case $n = 100$.

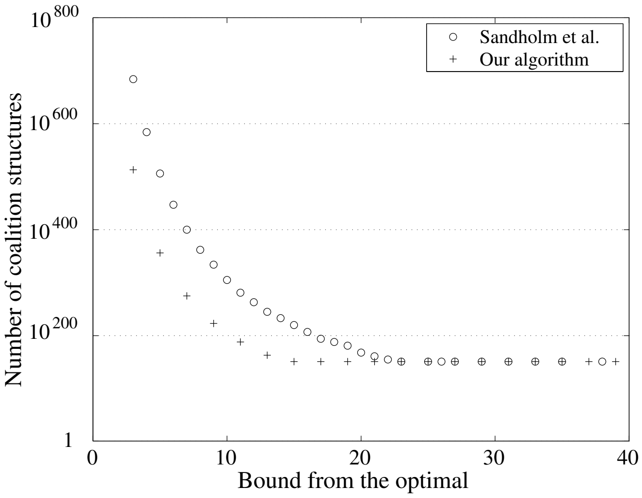

Figure 6. The case $n = 500$.

As we can see from the graphs, for large bounds from the optimal, there is no significant difference between the performance of our algorithm and Sandholm et al.'s: the number of coalition structures that each has to search are similar. However, for small bounds from the optimal, our algorithm is much faster (up to 10^{379} times faster in the graphs shown here). In these cases, the number of coalition structures that our algorithm has to search is much smaller because of our greater selectivity in searching through the subsets of L_k.

Moreover, for small bounds, our algorithm scales

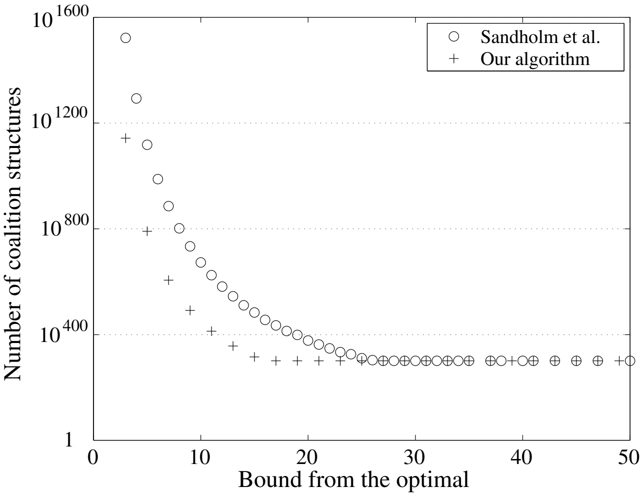

Figure 7. The case $n = 1000$.

very well as n increases. Thus the bigger n is, the more our algorithm outperforms Sandholm et al.'s. For example, with bound 3, our algorithm is more than 10^7 times faster for $n = 50$, more than 10^{23} times faster for $n = 100$, more than 10^{171} times faster for $n = 500$, and more than 10^{379} times faster for $n = 1000$. Note that these numbers would continue to increase the bigger we made n.

5. RELATED WORK

As mentioned in section 1, most of the existing work in coalition formation in game theory [6] has focused on payoff distribution (activity 3 in the coalition formation process) where it is usually assumed that a coalition structure has been formed, and the question is then how to divide the payoff so that the coalition structure is stable. In this context, many solutions have been proposed based on different stability concepts (e.g. the core, the Shapley value, the kernel, the stable set, and the bargaining set). *Transfer schemes* have also been developed to transfer non-stable payoff distributions to stable ones (while keeping the coalition structure unchanged).[6]

Recently, however, researchers in multi-agent systems have paid more attention to the problem of coalition structure generation. As mentioned before, Sandholm et al. [8] developed an anytime algorithm that guarantees to produce solutions within a finite bound from the optimal. However, as we have demonstrated,

6 For a comprehensive review on stability concepts and transfer schemes in game theory, see [6].

this algorithm is significantly slower than ours. On the other hand, Shehory and Kraus [10] consider a somewhat broader environment, where the coalitions can be overlapped. In this work, however, they reduce the complexity of the problem by limiting the size of the coalitions. They then develop a greedy algorithm that guarantees to produce a solution that is within a bound from the best solution possible given the limit on the number of agents. However, this best solution can be arbitrarily far from the actual optimal solution (without the limit on the size of the coalitions).

Some other researchers address both coalition structure generation and payoff distribution in competitive environments. Specifically, Ketchpel [3] presents a coalition formation method with cubic running time in the number of agents, but his method can neither guarantee a bound from the optimal nor stability. Shehory and Kraus's protocol [9] guarantees that if the agents follow it, a certain stability (kernel-stability) is met. In the same paper, they also present an alternative protocol that offers a weaker form of stability with polynomial running time. However, in both cases, no bound from the optimal is guaranteed.

More recent research in coalition formation area has also begun to pay attention to dynamic environments, where agents may enter or leave the coalition formation process and many uncertainties are present (e.g. the coalition value is not fixed, but it is context-based [4]). However, to date, no algorithm with bound guarantees has been developed for this environment.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we developed an anytime algorithm for coalition structure generation that can produce solutions within a finite bound from the optimal. We then benchmarked our algorithm against [8] which is the only other known algorithm for this task that can also establish a worst-case bound from the optimal. This comparison showed our algorithm to be significantly faster; for example, being up to 10^{379} times faster for systems containing 1000 agents for small bounds.

Future work will concentrate on undertaking a more detailed comparison of our algorithm with Sandholm et al.'s theoretically. That is, we will compare the number of coalition structures that each algorithm has to search (in order to establish a specific bound) mathematically. We then aim to apply the algorithm in a virtual organisation setting [5], where it will be used to increase the efficiency of the coordination between the constituent members of the virtual organisation. Specifically, once the virtual organisation has been formed (using our previous work on clearing in combinatorial auctions [1]), the coalition structure generation algorithm can be used to partition it into several sub-groups, each of which works on different activities in order to maximise the payoff of the whole virtual organisation.

References

[1] V. D. Dang and N. R. Jennings. Optimal clearing algorithms for multi-unit single item and multi-unit combinatorial auctions with demand/suppy function bidding. In *Proceedings of the Fifth International Conference on Electronic Commerce*, pages 25–30, 2003.

[2] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid. *The International Journal of High Performance Computing Applications*, 15(3):200–222, 2001.

[3] S. P. Ketchpel. Forming coalitions in the face of uncertain rewards. In *Proceedings of the Twelth National Conference on Artificial Intelligence*, pages 414–419, 1994.

[4] M. Klusch and A. Gerber. Dynamic coalition formation among rational agents. *IEEE Intelligent Systems*, 17(3):42–47, 2002.

[5] T. J. Norman, A. Preece, S. Chalmers, N. R. Jennings, M. Luck, V. D. Dang, T. D. Nguyen, V. Deora, J. Shao, A. Gray, and N. Fiddian. Conoise: Agent-based formation of virtual organisations. *Knowledge Based Systems Journal*, (to appear).

[6] A. Rapoport and J. P. Kahan. *Theories of Coalition Formation*. Lawrence Erlbaum Associates, 1984.

[7] S. Roman. *The Umbral Calculus*. Academic Press, 1984.

[8] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohme. Coalition structure generation with worst case guarantees. *Artificial Intelligence*, 111(1-2):209–238, 1999.

[9] O. Shehory and S. Kraus. A kernel-oriented model for coalition-formation in general environments: Implementation and results. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 134–140, 1996.

[10] O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence Journal*, 101(1-2):165–200, 1998.

[11] M. Tsvetovat and K. Sycara. Customer coalitions in the electronic marketplace. In *Proceedings of the Fourth International Conference on Autonomous Agents*, pages 263–264, 2000.