Coalition Structure Generation in Task-Based Settings

Viet Dung Dang and **Nicholas R. Jennings** [1]

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. 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? To date, most work on this problem has concentrated on simple *characteristic function games*. However, this lacks the notion of tasks which makes it more difficult to apply it in many applications. Against this background, this paper studies coalition structure generation in a general task-based setting. Specifically, we show that this problem is NP-hard and that the minimum number of coalition structures that need to be searched through in order to establish a solution within a bound from the optimal is exponential to the number of agents. We then go onto develop an anytime algorithm that can establish a solution within a bound from the optimal with a minimal search and can reduce the bound further if time permits.

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 long studied in game theory [3] and has recently become an important topic in multi-agent systems (where buyer agents may pool their requirements in order to obtain bigger group discounts), in grid computing (where multi-institution virtual organisations are viewed as being central to coordinated resource sharing and problem solving), and in e-business (where agile groupings of agents need to be formed in order to satisfy particular market niches). In all of these cases, the formation of coalitions aims to increase the agents' abilities to satisfy goals and to maximise their personal and/or the system's outcomes.

In this context, the coalition formation process can be viewed as being composed of three main activities [5]:

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[2]. 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: $\{\{a_1, a_2, a_3\}\}, \{\{a_1\}, \{a_2, a_3\}\}, \{\{a_2\}, \{a_3, a_1\}\}, \{\{a_3\}, \{a_1, a_2\}\}$ and $\{\{a_1\}, \{a_2\}, \{a_3\}\}$.
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 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. Here, our focus is on the first of these activities.

To date, most work on coalition structure generation has focused on simple *characteristic function games (CFGs)* (see section 6). 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. However, in many practical applications, it is often the case that a coalition is formed in order to perform some task from a pool of potential tasks that the agent system has to perform [2] [6]. But in CFGs this connection between coalitions and tasks is absent; it is simply assumed that a coalition's value is attained when its agents pool their resources and tasks together somehow. The notion of tasks generally increases the complexity of the problem as there are typically many ways to map coalitions to tasks (as will be shown in more detail in section 2). Also, the assumption of CFGs that a coalition value is not affected by the actions of non-members is no longer valid, as a coalition's value now depends on the tasks that it performs and this set of tasks are accessible by all agents. Recently, there have been some work on such task-based coalition formations [2] [6]. However, the authors typically made some limiting assumptions about either the coalition values or the maximum size of a coalition (see section 6 for more details). In this paper we don't make such assumptions, as we deal with a general case.

Against this background, this paper advances the state-of-the-art in the following ways. First, we analyse the complexity of the coalition structure generation problem in task-based coalition formation settings. We show that the problem is NP-hard and, moreover, the minimum number of coalition structures that need to be searched through in order to establish a solution within a bound from the optimal is exponential to the number of agents. Second, we develop an anytime algorithm that can establish a solution within a bound from

[1] School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK, email: {vdd, nrj}@ecs.soton.ac.uk.

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

the optimal and can reduce the bound further if time permits. We also consider a special case where each coalition carries out at most one task (refered as single-task coalitions in our paper) and show that in this case, the minimum number of coalition structures that need to be searched through in order to establish a bound from the optimal is much smaller than the general case.

The rest of the paper is structured as follows. Section 2 introduces our task-based coalition formation model. Section 3 analyses the complexity of the coalition structure generation problem and presents a minimal search to establish a bound from the optimal. Section 4 presents our anytime algorithm. Section 5 deals with the case of single-task coalitions. Finally, section 6 covers related work and section 7 concludes.

2 Task-Based Coalition Formation

This section formalises coalition structure generation in task-based settings, an extension of coalition formation in CFGs. Let A be the set of agents, and n be the number of agents in A (i.e. $|A| = n$). Let T be the set of tasks and m be the number of tasks: $T = \{t_1, t_2, ..., t_m\}$. As an extension of coalition formation in CFGs, we consider settings where for each subset $C \subseteq A$ and each set of tasks $V \subseteq T$ there is a coalition value $v(C, V)$ for the coalition (C, V). The coalition value $v(C, V)$ can be considered as the utility that agents in coalition C can attain by performing the tasks in V.[3] Thus, while in traditional CFG the coalition value is a function $v : 2^A \to R$, in our case it is a function $v : 2^A \times 2^T \to R$. It can be seen that for each CFG coalition C there are $(2^m - 1)$ corresponding task-based coalitions (C, V) as there are $(2^m - 1)$ possible set of tasks. As there are $(2^n - 1)$ CFG coalitions C, there are $(2^n - 1)(2^m - 1)$ task-based coalitions.

As in [3], we assume that every coalition value is non-negative:

$$v(C, V) \geq 0, \forall C \subseteq A, V \subseteq T \quad (1)$$

Definition 1 *A coalition structure CS is a partition of A into exhaustive disjoint (non-empty) coalitions, each of which will do a different set of tasks such as: i) these sets of tasks are disjoint, ii) at most one of them can be empty (that is, at most one coalition will do nothing, because if two or more of them do nothing then we can merge them together into one)[4]. Thus, $CS = \{(C_1, V_1), (C_2, V_2), ..., (C_k, V_k)\}$, $1 \leq k \leq n$, coalition C_i, $1 \leq i \leq k$, will do the set of tasks V_i such that the following conditions are satisfied:*

- $C_i \subseteq A, C_i \neq \emptyset, \forall 1 \leq i \leq k$
- $C_i \cap C_j = \emptyset, \forall 1 \leq i, j \leq k, i \neq j$
- $\cup_{i=1}^{k} C_i = A$
- $V_i \subseteq T, \forall 1 \leq i \leq k$
- $V_i \cap V_j = \emptyset, \forall 1 \leq i, j \leq k, i \neq j$
- $|\{V_i | V_i = \emptyset\}| \leq 1$

For example, for $A = \{a_1, a_2, a_3, a_4, a_5, a_6\}$ and $T = \{t_1, t_2, t_3, t_4, t_5, t_6\}$, the following is a coalition structure: $\{(\{a_1, a_2\}, \{t_1\}), (\{a_3\}, \{t_2, t_3\}), (\{a_4, a_5\}, \{t_4, t_5\})\}$. In the above coalition structure, coalition $\{a_1, a_2\}$ does task t_1, coalition $\{a_3\}$ does tasks t_2 and t_3, coalition $\{a_4, a_5\}$ carries out tasks t_4 and t_5, and agent a_6 does nothing (also task t_6 is not carried out by any coalition).

[3] In this paper, we use the word *coalition* to refer to both a group of agents (e.g. coalition C) and a group of agents with a set of tasks to perform (e.g. coalition (C, V)).

[4] The value of a coalition that does nothing is 0, that is, $v(C, \emptyset) = 0$.

Proposition 1 *Let the size of a coalition structure be the number of coalitions that it contains. Then the size of any task-based coalition structure will be less than the number of tasks plus 1. That is, for any coalition structure $CS = \{(C_1, V_1), (C_2, V_2), ..., (C_k, V_k)\}$, we have:*

$$k \leq m + 1 \quad (2)$$

PROOF. From definition 1 we have: the sets V_i are disjoint and at most one of them is empty. Thus the union of the sets $V_1, V_2, ...V_k$ must have at least $k - 1$ elements. But this union is a subset of T and T has m elements. Thus we have $k - 1 \leq m$ or $k \leq m + 1$. □

Proposition 2 *For each CFG coalition structure $(C_1, C_2, ..., C_k)$ ($k \leq m+1$), there are $(k+1)^m$ corresponding task-based coalition structures.*

PROOF. For each task $t \in T$, there are $k + 1$ possibilities: t is carried out by a coalition C_i, $1 \leq i \leq k$ or is not carried out by any coalition. As there are m tasks, there are $(k + 1)^m$ combinations of possibilities to form a task-based coalition structure from that CFG coalition structure. □

As in traditional CFG, we consider the *value* of a coalition structure $V(CS)$ in terms of its social welfare. That is, the value of a coalition structure is the sum of the values of its coalitions:

$$V(CS) = \sum_{i=1}^{k} v(C_i, V_i)$$

for $CS = \{(C_1, V_1), (C_2, V_2), ..., (C_k, V_k)\}$.

Also let L be the set of all coalition structures.

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

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

Moreover, a solution CS' is said to be *within a bound b from the optimal solution* iff: $V(CS^*)/V(CS') \leq b$.

Having presented the settings in this section, the next section analyses the complexity of the problem.

3 Complexity Analysis

In this section, we analyse the complexity of the problem and discuss the smallest number of coalition structures that need to be searched through in order to establish a solution that is within a bound from the optimal one.

First, we investigate the size of the search space. Just as in the case of CFGs, the set of all coalition structures can be partitioned into a number of layers, each layer L_k, $1 \leq k \leq n$ holds all coalition structures with size k.

In CFG settings, the number of coalition structures in L_k is $S(n, k)$, widely known in Mathematics as *the Stirling number of the Second Kind* [4]. The value of $S(n, k)$ can be computed by the following standard formula:

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

In task-based settings, the number of coalition structures in L_k can be calculated as follows.

Proposition 3 *The number of coalition structures with size k ($1 \leq k \leq m+1$) can be computed by the following formula:*

$$|L_k| = (k+1)^m \cdot S(n,k)$$

PROOF. As for each CFG coalition structure $(C_1, C_2, ..., C_k)$ ($k \leq m+1$), there are $(k+1)^m$ corresponding task-based coalition structures (proposition 2) and there are $S(n,k)$ CFG coalition structures with size k, the number of coalition structures with size k is $(k+1)^m \cdot S(n,k)$. $\Box$

Theorem 1 *The total number of coalition structures is* $\sum_{k=1}^{\min(n,m+1)}(k+1)^m$.

PROOF. As the number of coalition structures with size k ($1 \leq k \leq m+1$) is $(k+1)^m \cdot S(n,k)$ (proposition 3) and $k \leq n$ (as each coalition has at least one agent), thus the total number of coalition structures is $\sum_{k=1}^{\min(n,m+1)}(k+1)^m$. $\Box$

With this knowledge, we now discuss how a bound from the optimal can be established while searching as little of the space as possible. In CFG settings, it has been shown that searching two CFG layers L_1 and L_2 is the quickest way to ensure a bound from the optimal [5] (quickest here means searching the smallest number of coalition structures). In our task-based settings, however, we will show that we only need to search a portion of the corresponding layers L_1 and L_2 in order to establish a bound.

Theorem 2 *The smallest number of coalition structures that need to be searched in order to establish a bound from the optimal is at least* $(2^{m+n-1} - 1)$.

PROOF. Suppose K^* is the smallest set of coalition structures that need to be searched in order to establish a bound from the optimal. That is, K^* contains the smallest number of coalition structures to be searched in order to establish a bound from the optimal. Because the value of a coalition $v(C,V)$ can be arbitrarily big, in order to establish a bound from the optimal, every coalition (C,V) ($V \neq \emptyset$) needs to be included in one of the coalition structures in K^*.

First, we can see that for any coalition (C,V) such that $C = A$ or $V = T$, the coalition structure that contains it cannot contain any other coalition, as there is either no agent or no task left. Thus the number of coalition structures that need to be searched through to cover these coalitions (and only them) is the number of these coalitions.

Now let us count the number of coalitions (C,V) such that $C = A$ or $V = T$. There are $(2^m - 1)$ coalitions (A,V) as there are $(2^m - 1)$ non-empty subsets of T. Similarly, there are $(2^n - 1)$ coalitions (C,T). However, the coalition (A,T) is counted twice, so we have the number of coalitions (C,V) such that $C = A$ or $V = T$ is: $(2^m - 1) + (2^n - 1) - 1 = 2^m + 2^n - 3$. Thus the number of coalition structures that need to be searched through to cover these coalitions (and only them) is $2^m + 2^n - 3$.

Now, consider the coalitions (C,V) such that $C \subset A$ and $V \subset T$ (that is, C is a strict subset of A and V is a strict subset of T). Let us count the number of agent occurrences in every coalition[5]. For every $1 \leq i \leq n-1$, as there are $\binom{n}{i}$ subsets[6] of size i of A and $(2^m - 2)$ non-empty strict subsets of T, there are $\binom{n}{i} \cdot (2^m - 2)$ coalitions (C,V) where C is of size i.

Thus the number of agent occurrences in all coalitions (C,V) such that $C \subset A$ and $V \subset T$ is:

$$\sum_{i=1}^{n-1}(2^m - 2) \cdot \tbinom{n}{i} \cdot i \quad = \quad (2^m - 2) \cdot \sum_{i=1}^{n-1}\tbinom{n}{i} \cdot i \qquad (3)$$

$$= (2^m - 2)n \cdot \sum_{i=1}^{n-1} C_{n-1}^{i-1} \quad = \quad (2^m - 2)n \cdot (2^{n-1} - 1) \qquad (4)$$

As in each coalition structure, the number of agent occurrences is at most n, we need to search through at least $\frac{(2^m-2)n \cdot (2^{n-1}-1)}{n} = (2^m - 2) \cdot (2^{n-1} - 1)$ coalition structures to cover all coalitions (C,V) such that $C \subset A$ and $V \subset T$.

From the above results, we have the smallest number of coalition structures that need to be searched through in order to establish any bound from the optimal is $2^m + 2^n - 3 + (2^m - 2) \cdot (2^{n-1} - 1) = 2^{m+n-1} - 1$. $\Box$

Theorem 1 trivially leads to the following corollary.

Corollary 1 *The problem of coalition structure generation for task-based settings is NP-hard.*

Now, the following question naturally arises: is there a set of exactly $2^{m+n-1} - 1$ coalition structures that after searching through all of its coalition structures, a bound from the optimal can be established? We will show that there does indeed exist such a set in the theorem below.

Theorem 3 *The smallest number of coalition structures that need to be searched in order to establish a bound from the optimal is* $2^{m+n-1} - 1$.

PROOF. As we have proved in Theorem 1 that we need to search at least $2^{m+n-1} - 1$ coalition structures in order to establish a bound from the optimal, we just need to show that there exists a set of $2^{m+n-1} - 1$ coalition structures that after searching though this set, a bound from the optimal will be established. Thus we will construct such set of coalition structures below.

Consider the following set of coalition structures: $K = K_1 \cup K_2$ where K_1 is the set of coalition structures $\{(A,V)\}$ in which V is a non-empty subset of T and K_2 is the set of coalition structures $\{(C,V), (A \setminus C, T \setminus V)\}$ where C is a strict subset of A and V is a subset of T.

First, we will show that the number of coalition structures in K is exactly $(2^{m+n-1} - 1)$. As there are $2^m - 1$ non-empty subsets of T, K_1 contains $2^m - 1$ coalition structures. For the set K_2: there are $2^n - 2$ non-empty strict subsets C of A and 2^m subsets V of T, so there are $2^m(2^n - 2) = 2^{m+1}(2^{n-1} - 1)$ (C,V) coalitions. However, each coalition structure in K_2 contains two such (C,V) coalitions and each coalition (C,V) appears exactly once in a coalition structure in K_2, thus the number of coalition structures in K_2 is $2^{m+1}(2^{n-1} - 1)/2 = 2^m(2^{n-1} - 1)$. Thus we can count the number of coalition structures by summing up the number of coalition structures in K_1 and K_2 together:

$$2^m - 1 + 2^m(2^{n-1} - 1) = 2^{m+n-1} - 1$$

[5] When we count the number of agent occurrences, we take into account the repetition of agent occurrence. For example, in coalitions $\{a_1, a_2\}$ and $\{a_1, a_3\}$ the number of agents is 3, but the number of agent occurrences is 4.

[6] $\binom{n}{k}$, or ${}_nC_k$, is the standard mathematical notation to denote the number of different (unordered) combinations of k objects, taken from a pool of n objects and is calculated using the following formula: $\binom{n}{k} = \frac{n!}{k!(n-k)!}$.

Now we will show that after searching through all the coalition structures in K, a bound from the optimal can be established. Let $v(C^*, V^*) = max_{C \subseteq A, V \subseteq T} v(C, V)$. Consider the following coalition structure: $CS^1 = \{(C^*, V^*)\}$ (if $C = A$) or $\{(C^*, V^*), (A \setminus C^*, T \setminus V^*)\}$ (if C is a strict subset of A), we then have $V(CS^1) \geq v(C^*, V^*)$ (according to assumption 1).

Now, for any coalition structure $CS = \{(C_1, V_1), (C_2, V_2), ..., (C_k, V_k)\}$ we have:

$$V(CS) = \sum_{i=1}^{k} v(C_i, V_i) = \sum_{i=1, C_i \neq \emptyset, V_i \neq \emptyset}^{k} v(C_i, V_i)$$

Now, as there are n agents and m tasks, there are at most only $\min(m, n)$ coalitions with a non-zero value in any coalition structures. Thus:

$$V(CS) \leq min(m, n) \cdot v(C^*, V^*) \leq min(m, n) \cdot V(CS^1)$$

Thus $V(CS^*) \leq min(m, n) \cdot V(CS^1)$. As $CS^1 \in K$, after searching through all the coalition structures in K, a bound $b = min(m, n)$ from the optimal can be establish. □

Now, it can be seen that the set K contains most of layer L_1 (except $\{(A, \emptyset)\}$) and a subset of layer L_2 (as L_2 contains all coalition structures $\{(C, V_1), (A \setminus C, V_2)\}$ while K contains only coalition structures $\{(C, V_1), (A \setminus C, V_2)\}$ where $V_1 \cup V_2 = T$).

After establishing the bound $b = min(m + 1, n)$, if additional time remains, it would be desirable to lower the bound with further search Thus, the next section will present an anytime algorithm that can do this if time permits.

4 The Anytime Algorithm

In this section we present an *anytime algorithm* that can be interrupted at any time, and it establishes a monotonically improving bound. This algorithm is an adaptation of the anytime coalition structure generation algorithm in CFG settings developed in [1].

Definition 2 *Let $SL(n, k, c)$ be the set of all coalition structures that have exactly k coalitions and at least one coalition that contains at least c agents.*

Definition 3 *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 that contains at least c agents. That is:*

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

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 in the set K. Then after that, our algorithm searches $SL(n, \lceil n(q-1)/q \rceil)$ with q running from $\lfloor \frac{\min(m,n)}{2} \rfloor$ down to 2. Note that we start from $q = \lfloor \frac{\min(m,n)}{2} \rfloor$ because we have shown that after searching through the set K a bound $b = \min(m, n)$ is established and, later in this section, 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 $2q - 1 < \min(m, n)$ or $q = \lfloor \frac{\min(m,n)}{2} \rfloor$.

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.

The algorithm proceeds as follows:

- Step 1: Search through the set K
- From step 2 onwards, search, consecutively, through the sets $SL(n, \lceil n(q - 1)/q \rceil)$ with q running from $\lfloor \frac{\min(m,n)}{2} \rfloor$ down to 2. That is, search $SL(n, \lceil n(\frac{min(m,n)}{2} - 1)/\frac{\min(m,n)}{2} \rceil)$ in step 2, $SL(n, \lceil n(\frac{\min(m,n)}{2} - 2)/(\frac{\min(m,n)}{2} \rceil - 1))$ in step 3 and so on. Moreover, from step 3 onwards, as $SL(n, \lceil nq/(q + 1) \rceil) \subseteq SL(n, \lceil n(q - 1)/q \rceil)$, 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 social welfare so far.

Figure 1. The coalition structure generation algorithm

Theorem 4 *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. Due to the lack of space we omit the proof. It is, however, similar to the one in [1]. □

5 Single-Task Coalitions

Up to now, we have considered the general setting in which a coalition can carry out an arbitrary number of tasks. However, in some settings, there is a natural restriction such that each coalition can only undertake a single task at any one time. Now, with this additional structure in place, we can improve upon the general results that we have derived so far.

In this special case, we consider settings where for each subset $C \subseteq A$ and each task t there is a coalition value $v(C, t)$ for the coalition (C, t). Thus, here the coalition value is a function $v : 2^A \times T \rightarrow R$. It can be seen that for each CFG coalition C there are m corresponding single-task coalitions (C, t) as there are m possible tasks. As there are 2^n CFG coalitions C, there are $m2^n$ single-task coalitions.

A coalition structure CS is an exhaustive partition of A into $m+1$ disjoint (possibly empty) coalitions, each of the first m coalitions will do a different task and the last coalition does nothing. That is, $CS = \{(C_1, t_1), (C_2, t_2), ..., (C_m, t_m), C_{m+1}\}$, coalition C_i, $1 \leq i \leq m$, will do task t_i, such as the following conditions are satisfied:

- $C_i \cap C_j = \emptyset$
- $\cup_{i=1}^{m+1} C_i = A$

For example, for $A = \{a_1, a_2, a_3, a_4, a_5, a_6\}$ and $T = \{t_1, t_2, t_3\}$, the following is a coalition structure: $\{(\{a_1, a_2\}, t_1), (\{a_3\}, t_2), (\{a_4, a_5\}, t_3)\}$.

For the convenience of presentation, we will also write $CS = \{(C_1, t_1), (C_2, t_2), ..., (C_m, t_m)\}$ or $CS = \{C_1, C_2, ..., C_m\}$ in short.

Like the general case, the problem of coalition structure generation is then to find a coalition structure $CS*$ that maximises the social welfare (i.e. the sum of the values of all its coalitions).

By following a similar analysis as in the general case, we have the following corresponding results. First, we present the results regarding the size of the search space.

Proposition 4 *For each CFG coalition structure* $(C_1, C_2, ..., C_k)$ *(*$k \leq m+1$*), there are* $P(m+1, k)$ *corresponding task-based coalition structures*[7]

Theorem 5 *The total number of coalition structures is* $(m+1)^n$.

Definition 4 *The size of a coalition structure is the number of non-empty coalitions that it contains.*

Proposition 5 *The number of coalition structures with size* k *(*$1 \leq k \leq m+1$*) can be computed by the following formula:*

$$|L_k| = P(m+1, k)S(n, k)$$

We can can see that the size of the search space in single-task coalitions case is much smaller than one in the general case. Thus, we can expect that the smallest number of coalition structures that need to be searched in order to establish a bound from the optimal will also be much smaller. This expectation is true, as will be shown below.

Theorem 6 *The smallest number of coalition structures that need to be searched in order to establish a bound from the optimal is at least* $m \cdot 2^{n-1}$.

Corollary 2 *The problem of coalition structure generation for task-based coalition formation is NP-hard.*

Theorem 7 *The smallest number of coalition structures that need to be searched in order to establish a bound from the optimal is* $m \cdot 2^{n-1}$.

In this case, the set of $m \cdot 2^{n-1}$ coalition structures that after searching through, a bound of the optimal can be established is:

$$K = K_1 \cup K_2 \cup ... \cup K_m$$

with $K_i = \{\{C_1, C_2, ..., C_m\}\}$ where for C is any subset of A:

- $C_j = \emptyset, \forall 1 \leq j < i$ or $j > i+1$
- $C_i = C$
- $C_{i+1} = A \setminus C$ (for ease of presentation let $C_{m+1} \equiv C_1$)

Note that, for the above K_i, if a subset C of A is already considered, then we won't consider $A \setminus C$ as it will result in double counting. With this set K, a bound $b = \min(m, n)$ from the optimal can be guaranteed. As in the general case, after searching this set K, we can lower the bound if time permits by searching through the sets $SL(n, \lceil n(q-1)/q \rceil)$ with q running from $\lfloor \frac{min(m,n)}{2} \rfloor$ down to 2.

6 Related Work

As introduced in section 1, to date, most work on coalition structure generation has focused on CFGs. In particular, Sandholm et al. [5] analysed the problem of coalition structure generation in CFG theoretically and proved that this problem is NP-hard and, moreover, even finding a sub-optimal solution requires searching an exponential number of solutions. They also developed an anytime algorithm that can establish a worst-case bound from the optimal. Dang and Jennings [1] later developed another anytime algorithm with worst-case bound guarantees that was significantly faster than Sandholm et al.'s algorithm when small bounds are desirable.

More specifically related to this work, Shehory and Kraus [6] consider a task-based setting where the coalitions can overlap. In their 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). Li and Sycara [2] addressed a similar task-based setting to our single-task based one. However they made some limiting assumptions (including the fact that the cost for a task only depends on the number of agents and the task cost per agent decreases as the number of agents increases). We place no such restrictions on our environment.

7 Conclusions and Future Work

In this paper, we examined the problem of coalition structure generation in task-based settings, which are an important extension over traditional CFGs. For this setting, we showed that the problem is NP-hard and, moreover, even a sub-optimal solution (which is within a finite bound from the optimal) requires searching an exponential number of coalition structures. We also developed an anytime algorithm that can establish a solution within a bound from the optimal with a minimal search and can reduce the bound further if time permits. We then considered the single-task coalitions case where each coalition can carry out at most one task and show that in this case, the minimum number of coalition structures that need to be searched through in order to establish a bound from the optimal is much smaller than the general case.

In future work, we intend to further reduce the complexity of the algorithm, especially if more assumptions can be made. For example, we may consider the case where the task-based coalition value is monotonic for each task, (that is, the larger number of agents in a coalition, the bigger its coalition value).

8 Acknowledgement

This research was undertaken as part of the ARGUS II DARP. This is a collaborative project involving BAE SYSTEMS, QinetiQ, Rolls-Royce, Oxford University and Southampton University, funded by the industrial partners together with the EPSRC, MoD and DTI.

REFERENCES

[1] V. D. Dang and N. R. Jennings, 'Generating coalition structures with finite bound from the optimal guarantees', in *Proceedings of the Third International Conference on Autonomous Agents and Multi-Agent Systems*, pp. 564–571, (2004).

[2] C. Li and K. Sycara, 'A stable and efficient scheme for task allocation via agent coalition formation', *Algorithms for Cooperative Systems, World Scientific*, (2004).

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

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

[5] 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).

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

[7] J. V. Uspensky, *Introduction to Mathematical Probability*, New York: McGraw-Hill, 1937.

[7] $P(n, k)$, or ${}_nP_k$, is the standard mathematical notation to denote the number of different permutations of k objects, taken from a pool of n objects and is calculated using the following formula [7]: $P(n, k) = \frac{n!}{(n-k)!}$.