Coalition Structure Generation with Worst Case Guarantees

Tuomas Sandholm[1] Kate Larson[2] Martin Andersson[3]
Onn Shehory[4] Fernando Tohmé[5]

[1,2,3,5] Washington University
Department of Computer Science
One Brookings Drive, Campus Box 1045
St. Louis MO 63130-4899
{sandholm,ksl2,mra,tohme}@cs.wustl.edu

[4] Carnegie Mellon University
The Robotics Institute
Pittsburgh PA 15213-3890
onn@cs.cmu.edu

May 18, 1999

Abstract

Coalition formation is a key topic in multiagent systems. One may prefer a coalition structure that maximizes the sum of the values of the coalitions, but often the number of coalition structures is too large to allow exhaustive search for the optimal one. Furthermore, finding the optimal coalition structure is $\mathcal{NP}$-complete. But then, can the coalition structure found via a partial search be guaranteed to be within a bound from optimum?

We show that none of the previous coalition structure generation algorithms can establish any bound because they search fewer nodes than a threshold that we show necessary for establishing a bound. We present an algorithm that establishes a tight bound within this minimal amount of search, and show that any other algorithm would have to search strictly more. The fraction of nodes needed to be searched approaches zero as the number of agents grows.

If additional time remains, our anytime algorithm searches further, and establishes a progressively lower tight bound. Surprisingly, just searching one more node drops the bound in half. As desired, our algorithm lowers the bound rapidly early on, and exhibits diminishing returns to computation. It also significantly outperforms its obvious contenders. Finally, we show how to distribute the desired search across self-interested manipulative agents.[1]

Keywords: Coalition formation, anytime algorithm, multiagent systems, game theory, negotiation, distributed AI, resource-bounded reasoning.

[1]A short early version of this paper appeared at a conference [37].

1 Introduction

Multiagent systems are becoming increasingly important. One reason for this is the technology push of a growing standardized communication infrastructure—Internet, WWW, NII, EDI, KQML, FIPA, Concordia, Voyager, Odyssey, Telescript, Java, *etc*—over which separately designed agents belonging to different organizations can interact in an open environment in real-time and safely carry out transactions [35]. The second reason is strong application pull for computer support for negotiation at the operative decision making level. For example, we are witnessing the advent of small transaction commerce on the Internet for purchasing goods, information, and communication bandwidth. There is also an industrial trend toward virtual enterprises: dynamic alliances of small, agile enterprises which together can take advantage of economies of scale when available (e.g., respond to more diverse orders than individual agents can), but do not suffer from diseconomies of scale.

Multiagent technology facilitates the automated formation of such dynamic coalitions. This automation can save labor time of human negotiators, but in addition, other savings are possible because computational agents can be more effective at finding beneficial short-term coalitions than humans are in strategically and combinatorially complex settings.

This paper discusses coalition structure generation in settings where there are too many coalition structures to enumerate and evaluate due to, for example, costly or bounded computation and/or limited time. Instead, agents have to select a subset of coalition structures on which to focus their search. We study which subset the agents should focus on so that they are guaranteed to reach a coalition structure that has quality within a bound from the quality of the optimal coalition structure.

Although our algorithm reduces the search space drastically and therefore increases the number of agents for which a desirable coalition structure can be guaranteed, our algorithm is still exponential in the number of agents (linear in the number of coalitions), thus it may be inapplicable for very large numbers of agents. As we prove, only by searching this exponential space can one guarantee a bound from optimum.

1.1 The three activities of coalition formation

In many domains, real world parties—e.g., companies or individual people—can save costs by coordinating their activities with other parties. For example, when the planning activities are automated, it can be useful to automate the coordination activities as well. This can be done via a negotiating software agent representing each party. A key issue in such automated negotiation is the formation of coalitions. Coalition formation includes three activities:

1. *Coalition structure generation*: formation of coalitions by the agents such that agents within each coalition coordinate their activities, but agents do not coordinate between coalitions. Precisely, this means partitioning the set of agents into exhaustive and disjoint coalitions. This partition is called a *coalition structure* (CS).

2. *Solving the optimization problem* of each coalition. This means pooling the tasks and resources of the agents in the coalition, and solving their joint problem. The coalition's objective is to maximize monetary value: money received from outside the system for accomplishing tasks minus the cost of using resources. (In some problems, not all tasks have to be handled. This can be incorporated by associating costs with omitted tasks.)

3. *Dividing the value* of the generated solution among agents.

These activities interact. For example, the coalition that an agent wants to join depends on the portion of the value that the agent would be allocated in each potential coalition. While in the long run it would be desirable to construct an integrated theory that encompasses all three activities, in this paper we focus on the coalition structure generation activity. Some ways of tying this work together with methods for addressing the other two activities are discussed in Section 9.

1.2 Outline of the paper

This paper focuses on settings were the coalition structure generation activity is resource-bounded: it is too complex to find the optimal (social welfare maximizing) coalition structure. The paper is organized as follows. Section 2 describes the model of coalition structure generation, analyzes the complexity of finding the optimal coalition structure, and explains why the problem has not received much prior attention. Section 3 formalizes the driving question of the paper: what coalition structures should the search focus on so as to guarantee that the solution is within a bound from optimum? Section 4 shows that none of the previous coalition structure generation algorithms can establish any bound because they search fewer nodes than a threshold that we show necessary for establishing a bound. We present an algorithm that establishes a tight bound within this minimal amount of search, and show that any other algorithm would have to search strictly more. Section 5 describes how the bound can be decreased further using an anytime algorithm if additional time remains. As desired, our algorithm lowers the bound rapidly early on, and exhibits diminishing returns to computation. Section 6 shows how it outperforms its obvious contenders. Section 7 discusses variants of the problem, their complexity, and algorithms for solving them. Section 8 introduces a nonmanipulable scheme for distributing the search across multiple agents. Finally, Section 9 discusses related research, and Section 10 concludes and presents future research directions.

2 Coalition structure generation in characteristic function games

Let A be the set of agents, and $a = |A|$. As is common practice [20, 42, 29, 49, 46, 44, 51, 22, 41], we study coalition formation in *characteristic function games* (CFGs). In such games, the value of each coalition S is given by a characteristic function v_S. (These coalition values v_S may represent the quality of the optimal solution for each coalition's optimization problem, or they may represent the best bounded-rational value that a coalition can get given limited or costly computational resources for solving the problem [41].)

A	The set of agents.
a	The number of agents, i.e. $\|A\|$.
S	A coalition.
v_S	Value of coalition S.
CS	A coalition structure, i.e. a partition of agents, A, into disjoint coalitions.
$V(CS)$	Value of coalition structure CS, see Eq. 2.
CS^*	Social welfare maximizing coalition structure.
M	The set of all possible coalition structures.
N	Coalition structures seen so far in coalition structure generation search.
n	Number of coalition structures seen so far, i.e. $\|N\|$.
n_{min}	Minimum n that guarantees a ratio bound from optimum.
CS^*_N	Coalition structure with highest welfare among the ones seen.
k	Worst case ratio bound on value of the coalition structure, see Eq. 7.

Table 1: *Important symbols used in the paper.*

Not all settings are CFGs. In CFGs, each coalition S has some value v_S, i.e. each coalition's value is independent of nonmembers' actions. However, in general the value of a coalition may depend on nonmembers' actions due to positive and negative externalities (interactions of the agents' solutions). Negative externalities between a coalition and nonmembers are often caused by shared resources. Once nonmembers are using a portion of the resource, not enough of that resource is available to agents in the coalition to carry out the planned solution at the minimum cost. Negative externalities can also be caused by conflicting goals. In satisfying their own goals, nonmembers may actually move the world further from the coalition's goal state(s) [30]. Positive externalities are often caused by partially overlapping goals. In satisfying their goals, nonmembers may actually move the world closer to the coalition's goal state(s). From there the coalition can reach its goals at less expense than it could have without the actions of nonmembers. General settings with possible externalities can be modeled as *normal form games* (NFGs). CFGs are a strict subset of NFGs. However, many (but clearly not all) real-world multiagent problems happen to be CFGs, see e.g. [41]. This is because in many real-world settings, a coalition's possible actions and payoff are unaffected by the actions of nonmembers.

We assume that each coalition's value is nonnegative:

$$v_S \geq 0 \tag{1}$$

However, if some coalitions' values are negative, but each coalition's value is bounded from below (i.e. not infinitely negative), one can normalize the coalition values by subtracting (at least) $\min_{S \subseteq A} v_S$ from all coalition values v_S. This rescales the coalition values so that Equation 1 holds for all coalitions. This rescaled game is strategically equivalent to the original game [20]. Actually, all of the claims of the paper are valid as long as Equation 1 holds for the coalitions that the algorithm *sees*. Coalitions not seen during the search are free to be arbitrarily bad.

A coalition structure CS is a partition of agents, A, into disjoint, exhaustive coalitions. In other words, in a coalition structure each agent belongs to exactly one coalition, and some agents may be alone in their coalitions. We will call the set of all coalition structures M. For example, in a game with three agents, there are 7 possible coalitions: {1}, {2}, {3}, {1,2}, {2,3}, {1,3}, {1,2,3} and 5 possible coalition structures: {{1}, {2}, {3}}, {{1}, {2,3}}, {{2}, {1,3}}, {{3}, {1,2}}, {{1,2,3}}. The value of a coalition structure is

$$V(CS) = \sum_{S \in CS} v_S \tag{2}$$

Usually the goal is to maximize the social welfare of the agents by finding a coalition structure

$$CS^* = \operatorname*{argmax}_{CS \in M} V(CS) \tag{3}$$

2.1 Complexity of finding the optimal coalition structure

The problem is that finding a welfare maximizing coalition structure is computationally complex. The following subsections detail the complexities involved.

2.1.1 Size of the input

The input is exponential in the number of agents. The input to a coalition structure formation algorithm contains the values, v_S, of the coalitions. One value is associated with each coalition, and there are $2^a - 1$ coalitions. This is simply the number of subsets of A (not counting the empty set).

Conceptually one could exclude the values of some coalitions from the input, or decide that the coalition structure generation algorithm ignores part of the input. However, excluding even a single coalition from consideration may cause the value of the best remaining coalition structure, say $V(CS)$, to be arbitrarily far from optimum, i.e. from $V(CS^*)$. This is because the value of the excluded coalition may have been arbitrarily much greater than the values of the other coalitions.

2.1.2 Number of coalition structures

Another problem is that the number of coalition structures grows rapidly (considerably faster than the number of coalitions grows) as the number of agents increases. The exact number of coalition structures is

$$\sum_{i=1}^{a} Z(a,i) \tag{4}$$

where $Z(a,i)$ is the number of coalition structures with i coalitions. The quantity $Z(a,i)$—also known as the Stirling number of the second kind—is captured by the following recurrence:

$$Z(a,i) = iZ(a-1,i) + Z(a-1,i-1), \text{ where } Z(a,a) = Z(a,1) = 1 \tag{5}$$

This recurrence can be understood by considering the addition of a new agent to a game with $a-1$ agents. The first term, $iZ(a-1,i)$, counts the number of coalition structures formed by adding the new agent to one of the existing coalitions. There are i choices because the existing coalition structures have i coalitions. The second term, $Z(a-1,i-1)$, considers adding the new agent into a coalition of its own, and therefore existing coalition structures with only $i-1$ coalitions are counted.

The following proposition characterizes the asymptotic complexity in closed form (the proof is given in Appendix A):

Proposition 1 *The number of coalition structures is* $O(a^a)$ *and* $\omega(a^{a/2})$.

The number of coalition structures is so large that not all coalition structures can be enumerated—unless the number of agents is extremely small (below 15 or so in practice). Therefore, exhaustive enumeration is not a viable method for searching for the optimal coalition structure.

2.2 Lack of prior attention

Coalition structure generation has not received much attention previously. Research has mostly focused [20, 51, 29, 48, 25] on *superadditive* games, i.e. games where $v_{S \cup T} \geq v_S + v_T$ for all disjoint coalitions $S, T \subseteq A$. In such games, coalition structure generation is trivial because the agents are best off by forming the grand coalition where all agents operate together. In other words, in such games, $\{A\}$ is a social welfare maximizing coalition structure.

Superadditivity means that any pair of coalitions is best off by merging into one. Classically it is argued that almost all games are superadditive because, at worst, the agents in a composite coalition can use solutions that they had when they were in separate coalitions.

However, many games are not superadditive because there is some cost to the coalition formation process itself. For example, there might be coordination overhead like communication costs, or possible anti-trust penalties. Similarly, solving the optimization problem of a composite coalition may be more complex than solving the optimization problems of component coalitions. Therefore, under costly computation, component coalitions may be better off by not forming the composite coalition [41]. Also, if time is limited, the agents may not have time to carry out the communications and computations required to coordinate effectively within a composite coalition, so component coalitions may be more advantageous.

Some non-superadditive games are *subadditive*, i.e. $v_{S \cup T} < v_S + v_T$ for all disjoint coalitions $S, T \subseteq A$. In subadditive games, the agents are best off by operating alone, i.e. $\{\{a_1\}, \{a_2\}, ..., \{a_{|A|}\}\}$ is a social welfare maximizing coalition structure.

Some games are neither superadditive nor subadditive because the characteristic function fulfills the condition of superadditivity for some coalitions and the condition of subadditivity for others. In other words, some coalitions are best off merging while others are not. In such cases, the social welfare maximizing coalition structure varies. The grand coalition may be the optimal coalition structure even in games which are not superadditive. Similarly, every agent operating alone may be optimal even in games which are not subadditive.

This paper focuses on games that might be neither superadditive nor subadditive, and if they are, this is not known in advance. In such settings, coalition structure generation is computationally complex as discussed above.

3 Approximate coalition structure generation

Taking an outsider's view, the coalition structure generation process can be viewed as search in a *coalition structure graph*, Figure 1. As we discussed, finding the optimal coalition structure in this graph is infeasible due to computational complexity.

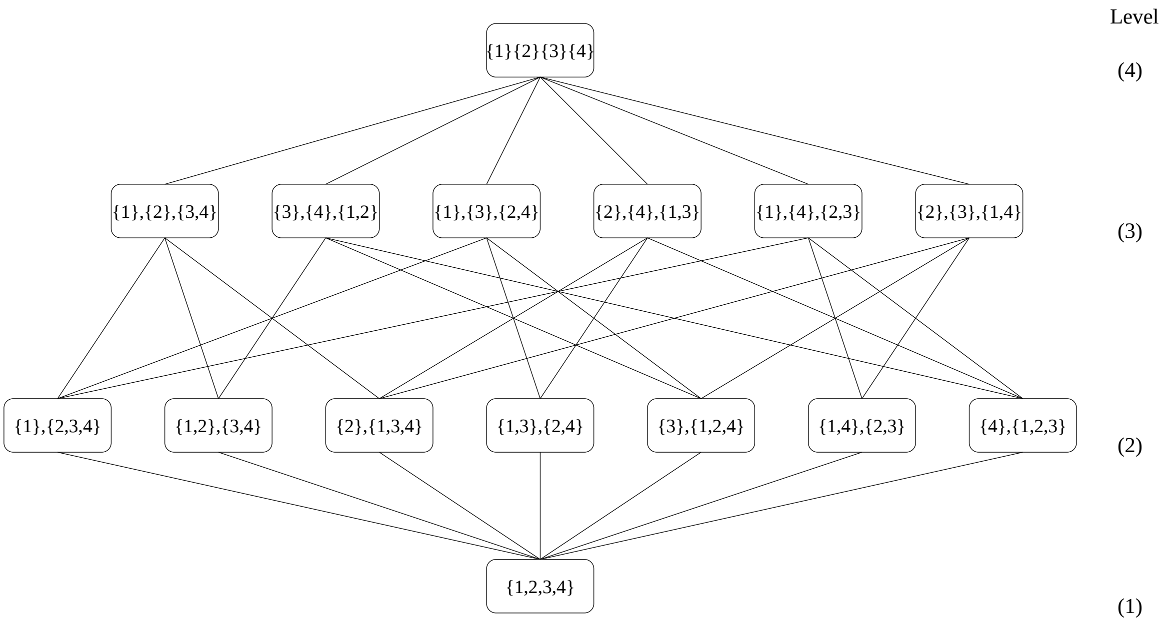

Figure 1: *Coalition structure graph for a 4-agent game. The nodes represent coalition structures. The arcs represent mergers of two coalition when followed downward, and splits of a coalition into two coalitions when followed upward.*

Now, how should such a graph be searched if there are too many nodes to search it completely? Also, how would one justify a particular search over another given that neither is guaranteed to lead to the optimal solution?

We would like to search through a subset $N \subseteq M$ of coalition structures, pick the best coalition structure we have seen:

$$CS_N^* = \underset{CS \in N}{\operatorname{argmax}}\, V(CS) \tag{6}$$

and be guaranteed that this coalition structure is within a bound from optimal, i.e. that there is a finite (and as small as possible) k,

$$k = \min\{\kappa\} \text{ where } \kappa \geq \frac{V(CS^*)}{V(CS_N^*)} \tag{7}$$

We define n_{min} to be the smallest size of N that allows us to establish such a bound k.

Intuition might suggest that a bound cannot be established without searching through all coalition structures. After all, it seems that even leaving out one coalition structure may exclude the coalition structure that is arbitrarily much better than the others. We show that, quite surprisingly, Equations 1 and 2 provide enough structure that this cannot happen.

4 Minimal search to establish a bound

This section discusses how a bound k can be established while searching as little of the graph as possible.

Theorem 1 *To bound k, it suffices to search the lowest two levels of the coalition structure graph (Figure 1). With this search, the bound $k = a$, and the number of nodes searched is $n = 2^{a-1}$.*

Proof. To establish a bound, v_S of each coalition S has to be observed (in some coalition structure). The a-agent coalition can be observed by visiting the bottom node. The second lowest level has coalition structures where exactly one subset of agents has split away from the grand coalition. Therefore, we see all subsets at this level (except the grand coalition). It follows that a search of the lowest two levels sees all coalitions.

In general, CS^* can include at most a coalitions. Therefore,

$$V(CS^*) \leq a \max_S v_S \leq a \max_{CS \in N} V(CS) = aV(CS_N^*).$$

Now we can set $k = a \geq \frac{V(CS^*)}{V(CS_N^*)}$.

The number of coalition structures on the lowest level is 1. The number of coalitions on the second lowest level is $2^a - 2$ (all subsets of A, except the empty set and the grand coalition). There are two coalitions per coalition structure on this level, so there are $\frac{2^a-2}{2}$ coalition structures at the second to lowest level. So, there are $1 + \frac{2^a-2}{2} = 2^{a-1}$ coalition structures (nodes) on the lowest two levels. □

Theorem 2 *For the algorithm that searches the lowest two levels of the graph, the bound $k = a$ is tight.*

Proof. We construct a worst case via which the bound is shown to be tight. Choose $v_S = 1$ for all coalitions S of size 1, and $v_S = 0$ for the other coalitions. Now, $CS^* = \{\{1\}, \{2\}, ..., \{a\}\}$, and $V(CS^*) = a$. Then $CS_N^* = \{\{1\}, \{2, ..., a\}\}$. (This is not unique because all coalition structures where one agent has split off from the grand coalition have the same value.) Because $V(CS_N^*) = 1$, $\frac{V(CS^*)}{V(CS_N^*)} = \frac{a}{1} = a$. □

Theorem 3 *No other search algorithm (than the one that searches the bottom two levels) can establish any bound k while searching only $n = 2^{a-1}$ nodes or fewer.*

Proof. In order to establish a bound k, v_S of each coalition S must be observed. The node on the bottom level of the graph must be observed since it is the only node where the grand coalition appears. Assume that the algorithm omits r nodes on the second level. Each of the omitted nodes has $CS = \{P, Q\}$. Since coalitions P and Q are never again in the same coalition structure, two extra nodes in the graph have to be visited to observe v_P and v_Q. Assume r coalition structures $\{P_1, Q_1\}, \{P_2, Q_2\}, \ldots, \{P_r, Q_r\}$ are omitted. Since for $i, j, i \neq j$, at least one of the following is true, $P_i \cap P_j \neq \emptyset, P_i \cap Q_j \neq \emptyset$, or $Q_i \cap Q_j \neq \emptyset$, at least $r + 1$ coalition structures must be visited to replace the r coalition structures omitted. Therefore, for the algorithm to establish k, it must search $n > 2^{a-1}$ nodes. $\square$

So, $n_{min} = 2^{a-1}$, and this is uniquely established via a search algorithm that visits the lowest two levels of the graph (order of these visits does not matter).

4.1 Positive interpretation

Interpreted positively, our results (Theorem 1) show that—somewhat unintuitively—a worst case bound from optimum can be guaranteed without seeing all CSs. Moreover, as the number of agents grows, the fraction of coalition structures needed to be searched approaches zero, i.e. $\frac{n_{min}}{|M|} \to 0$ as $a \to \infty$. This is because the algorithm needs to see only 2^{a-1} coalition structures while the total number of coalition structures is $\omega(a^{a/2})$. See Figure 2. For example, in a 10-agent game only 0.44 % of the search space needs to be searched to establish a bound, in a 15-agent game only 0.0012 %, and in a 20-agent game only 0.0000010 %.

Furthermore, the bound can be established in linear time in the size of the input. This is because the input has $2^a - 1$ coalition values while only 2^{a-1} coalition structures have to be seen to establish a bound.

4.2 Interpretation as an impossibility result

Interpreted negatively, our results (Theorem 3) show that exponentially many (2^{a-1}) coalition structures have to be searched before a bound can be established. This may be prohibitively complex if the number of agents is large—albeit significantly better than attempting to enumerate all coalition structures.

Viewed as a general impossibility result, Theorem 3 states that no algorithm for coalition structure generation can establish a bound in general characteristic function games without trying at least 2^{a-1} coalition structures. This sheds light on earlier

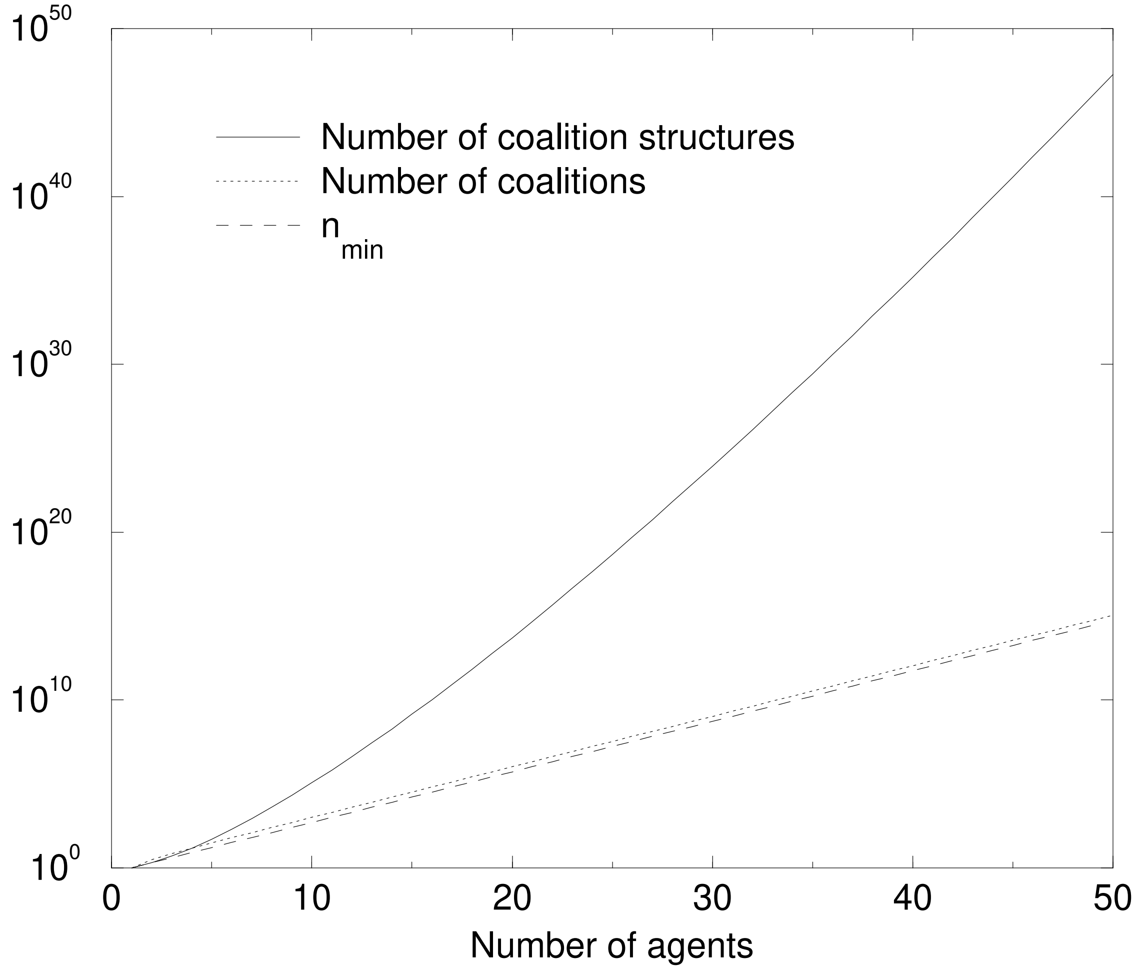

Figure 2: *Number of coalition structures, coalitions, and coalition structures needed to be searched. We use a logarithmic scale on the value axis; otherwise* n_{min} *and the number of coalitions would be so small compared to the number of coalition structures that their curves would be indistinguishable from the category axis.*

algorithms. Specifically, all prior coalition structure generation algorithms for general characteristic function games [44, 22]—which we know of—fail to establish such a bound. In other words, the coalition structure that they find may be arbitrarily far from optimal.

5 Lowering the bound with further search

If the lowest two levels have been searched, and additional time remains, it would be desirable to lower the bound with further search. We have devised the following algorithm that will establish a bound in the minimal amount of search, and then rapidly reduce the bound further if there is time for more search. This further search is an *anytime algorithm [4, 19, 50]*: it can be interrupted at any time, and it establishes a monotonically improving bound, k. (If the domain happens to be superadditive, the algorithm finds the optimal coalition structure (grand coalition) immediately as the first node that it searches.)

Algorithm 1 COALITION-STRUCTURE-SEARCH-1

1. *Search the* **bottom** *two levels of the coalition structure graph.*
2. *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* $a-3$*).*
3. *Return the coalition structure that has the highest welfare among those seen so far.*

In the rest of this section, we analyze how this algorithm reduces the worst case bound, k, as more of the graph is searched. The analysis is tricky because the elusive worst case (CS^*) moves around in the graph for different searches, N. We introduce the notation

$$h = \lfloor \frac{a-l}{2} \rfloor + 2 \tag{8}$$

which is used throughout this section.

Lemma 1 *Assume that Algorithm 1 has just completed searching level* l*. Then*

1. *If* $a \equiv l \ (mod\ 2)$ *coalitions of size* h *will have been seen paired together with all coalitions of size* $h-2$ *and smaller.*
2. *If* $a \not\equiv l \ (mod\ 2)$ *coalitions of size* h *will have been seen paired together with all coalitions of size* $h-1$ *and smaller.*

Proof.

1. At level l the largest coalition in any coalition structure has size $a - l + 1$. Therefore, one of the coalition structures at level l is of the form $S_1, S_2, \ldots, S_l$ where $|S_i| = 1$ for $i < l$ and $|S_l| = a - l + 1$. Since $a \equiv l \,(\mathrm{mod}\ 2)$, $h = \frac{a-l}{2} + 2$. Take coalition S_l and remove h agents from it. Call the new coalition formed by the h agents S_l'. We will distribute the remaining $\frac{a-l}{2} - 1$ agents among the coalitions of size 1. By doing this we can enumerate all possible coalitions that can appear pairwise with coalition S_l' on level l. For $j = 1, 2, \ldots, \frac{a-l}{2} - 1$, place $\frac{a-1}{2} - j$ agents in coalition S_1 and call the new coalition S_1^j. Redistribute the remaining $j - 1$ agents among coalitions $S_2, \ldots, S_{l-1}$. For each j we have listed a coalition structure containing both S_l' and S_1^j. The largest of these S_1^j has size $\frac{a-l}{2}$, or $h - 2$.

2. Since $a \not\equiv l \,(\mathrm{mod}\ 2)$, $h = \frac{a-1-l}{2} + 2$. Follow the same procedure as for case 1 except that this time there are $\frac{a-1-l}{2}$ remaining agents to be redistributed once S_l' has been formed. Therefore, when we redistribute all these agents among the coalitions $S_1, \ldots S_{l-1}$, we get all coalitions that were found in part 1, along with coalitions of size $h - 1$. $\Box$

From Lemma 1, it follows that after searching level l with Algorithm 1, we cannot have seen two coalitions of h members together in the same coalition structure.

Theorem 4 *Assume that Algorithm 1 has just completed searching level l. The bound $k(n)$ is $\lceil \frac{a}{h} \rceil$ if $a \equiv h - 1 \,(\mathrm{mod}\ h)$ and $a \equiv l \,(\mathrm{mod}\ 2)$. Otherwise the bound is $\lfloor \frac{a}{h} \rfloor$.*

Proof. Case 1: Assume $a \equiv h - 1 \,(\mathrm{mod}\ h)$ and $a \equiv l \,(\mathrm{mod}\ 2)$. Let α be an assignment of coalition values which give the worst case. For any other assignment of coalition values, β, the inequality $k(n) = \frac{V_\alpha(CS^*)}{V_\alpha(CS_N)} \geq \frac{V_\beta(CS^*)}{V_\beta(CS_N)}$ holds. Since CS^* is the best coalition structure under α, we can assume that $v_S = 0$ for all coalitions $S \notin CS^*$ without decreasing the ratio $\frac{V_\alpha(CS^*)}{V_\alpha(CS_N)}$. Also, no two coalitions $S, S' \in CS^*$ can appear together if $v_S + v_{S'} > \max_{S'' \in CS^*} v_{S''}$ since otherwise we could decrease the ratio $k(n)$. Therefore $V_\alpha(CS_N) = \max_{S \in CS^*} v_S$. Call this value v^*. We can derive an equivalent worst case, α', from α as follows:

1. Find a coalition structure CS' with $\lfloor \frac{a}{h} \rfloor$ coalitions of size h and one coalition of size $h - 1$.

2. Define $\overline{v} = \frac{V_\alpha(CS^*)}{\lfloor \frac{a}{h} \rfloor + 1}$.

3. Assign value $\overline{v}$ to each coalition in CS' and let all coalitions not in CS' have value 0.

Clearly $V_\alpha(CS^*) = V_{\alpha'}(CS')$. From Lemma 1 we know that no two coalitions in CS' have been seen together. The best value of a coalition structure seen during the search is $V_{\alpha'}(CS_N) = \overline{v}$. Therefore the following inequalities hold:

$$V_{\alpha'}(CS') = (\lfloor \frac{a}{h} \rfloor + 1)\overline{v} = (\lfloor \frac{a}{h} \rfloor + 1)V_{\alpha'}(CS_N)$$

$$(\lfloor \frac{a}{h} \rfloor + 1)V_{\alpha'}(CS_N) \leq (\lfloor \frac{a}{h} \rfloor + 1)v^* \leq (\lfloor \frac{a}{h} \rfloor + 1)V_\alpha(CS_N).$$

Since $V_\alpha(CS^*) = V_{\alpha'}(CS')$ and $V_{\alpha'}(CS_N) \leq V_\alpha(CS_N)$,

$$k(n) = \frac{V_\alpha(CS^*)}{V_\alpha(CS_N)} \leq \frac{V_{\alpha'}(CS')}{V_{\alpha'}(CS_N)} = \lfloor \frac{a}{h} \rfloor + 1 = \lceil \frac{a}{h} \rceil.$$

Therefore the bound is $\lceil \frac{a}{h} \rceil$.

Case 2: This is a similar argument as in Case 1, except that the assignment of values to the coalitions in the equivalent worst case coalition structure is different. Define α as before and let CS^+ be a coalition structure with $\lfloor \frac{a}{h} \rfloor$ coalitions of size h and one possible remainder coalition of size less than h. Define $\overline{v} = \frac{V_\alpha(CS^*)}{\lfloor \frac{a}{h} \rfloor}$. If $S \in CS^+$ and $|S| = h$, then set $v_S = \overline{v}$, otherwise $v_S = 0$. The best coalition structure seen has value $V_{\alpha^+}(CS_N) = \overline{v}$ and we have the following inequalities:

$$V_{\alpha^+}(CS^+) = (\lfloor \frac{a}{h} \rfloor)\overline{v} = (\lfloor \frac{a}{h} \rfloor)V_{\alpha^+}(CS_N)$$

$$(\lfloor \frac{a}{h} \rfloor)V_{\alpha^+}(CS_N) \leq (\lfloor \frac{a}{h} \rfloor)v^+ \leq (\lfloor \frac{a}{h} \rfloor)V_\alpha(CS_N).$$

Therefore the bound $k(n) = \lfloor \frac{a}{h} \rfloor$. $\square$

Theorem 5 *Right after completing level l with Algorithm 1, the bound in Theorem 4 is tight.*

Proof. Case 1: Assume $a \equiv h - 1 \pmod h$ and $a \equiv l \pmod 2$. The bound is $\lceil \frac{a}{h} \rceil$. Assume you have the coalition structure CS' from Theorem 4. Assign value 1 to each coalition $S \in CS'$ and assign value 0 to all other coalitions. Then $V(CS') = \lceil \frac{a}{h} \rceil$. Since (according to Lemma 1) no two of the coalitions in CS' have ever appeared in the same coalition structure, $V(CS_N) = 1$. Therefore $\frac{V(CS')}{V(CS_N)} = \frac{\lceil \frac{a}{h} \rceil}{1} = \lceil \frac{a}{h} \rceil$ and the bound is tight.

Case 2: Assume $a \not\equiv h - 1 \pmod h$ or $a \not\equiv l \pmod 2$. The bound is $\lfloor \frac{a}{h} \rfloor$. Assign value 1 to each coalition $S \in CS^+$ from Theorem 4 and assign value 0 to all other coalitions. Then $V(CS^+) = \lfloor \frac{a}{h} \rfloor$ and $V(CS_N) = 1$. Therefore $\frac{V(CS^+)}{V(CS_N)} = \frac{\lfloor \frac{a}{h} \rfloor}{1} = \lfloor \frac{a}{h} \rfloor$ and the bound is tight. $\square$

As we have shown in the previous section, before 2^{a-1} nodes have been searched, no bound can be established, and at $n = 2^{a-1}$ the bound $k = a$. The surprising fact is that by seeing just one additional node ($n = 2^{a-1} + 1$), i.e. the top node, the bound drops in half ($k = \frac{a}{2}$). Then, to drop k to about $\frac{a}{3}$, two more levels need to be searched. Roughly speaking, the divisor in the bound increases by one every time two more levels are searched, but seeing only one more level helps very little.

Put together, the anytime phase (step 2) of Algorithm 1 has the desirable feature that the bound drops rapidly early on, and there are overall diminishing returns to further search, Figure 3.

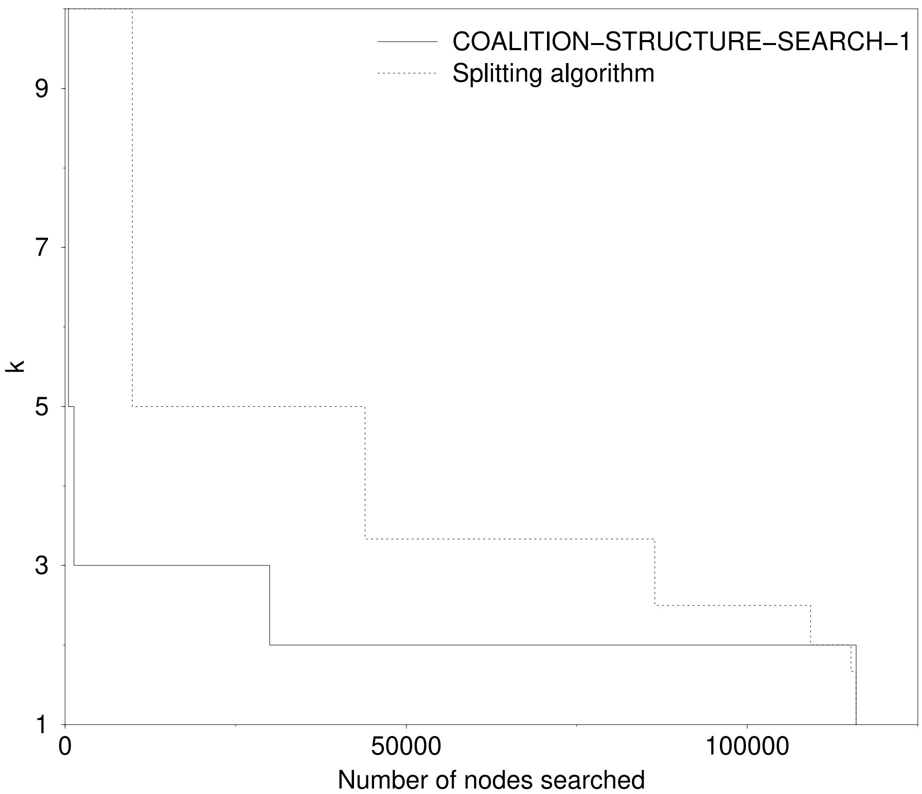

Figure 3: *Ratio bound k as a function of search size in a 10-agent game.*

Trivially, a bound of $k = \frac{a}{2}$ can be established in time that is linear in the size of the input. This is because the input has $2^a - 1$ numbers while a bound $k = \frac{a}{2}$ can be established by seeing $2^a - 1$ coalition structures: the bottom two levels and the top node.

6 Comparison to other algorithms

All previous coalition structure generation algorithms for general CFGs [44, 22]—that we know of—fail to establish any worst case bound because they search fewer than 2^{a-1} coalition structures. Therefore, we compare our Algorithm 1 to two other obvious candidates:

- **Merging algorithm**, i.e. breadth first search from the top of the coalition structure graph. This algorithm cannot establish any bound before it has searched the entire graph. This is because, to establish a bound, the algorithm needs to see every coalition, and the grand coalition only occurs in the bottom node. Visiting the grand coalition as a special case would not help much since at least part of level 2 needs to be searched as well: coalitions of size $a-2$ only occur there.

- **Splitting algorithm**, i.e. breadth first search from the bottom of the graph. This is identical to Algorithm 1 up to the point where 2^{a-1} nodes have been searched, and a bound $k = a$ has been established. After that, the splitting algorithm reduces the bound much slower than Algorithm 1. This can be shown by constructing bad cases for the splitting algorithm (the worst case may be even worse). To construct a bad case, set $v_S = 1$ if $|S| = 1$, and $v_S = 0$ otherwise. Now, $CS^* = \{\{1\}, ..., \{a\}\}$, $V(CS^*) = a$, and $V(CS^*_N) = l - 1$, where l is the level that the algorithm has completed (because the number of unit coalitions in a CS never exceeds $l-1$). So, $\frac{V(CS^*)}{V(CS^*_N)} = \frac{a}{l-1}$. (The only exception comes when the algorithm completes the last (top) level, i.e $l = a$. Then $\frac{V(CS^*)}{V(CS^*_N)} = 1$.) See Figure 3. In other words the divisor drops by one every time a level is searched. However, the levels that this algorithm searches first have many more nodes than the levels that Algorithm 1 searches first.

While this comparison is based on worst (or bad) case performance, a recent conference paper compares the average case performance of these three algorithms experimentally using four different ways of choosing the coalition structure values [23]. While each of the algorithms dominated the others in different settings, COALITION-STRUCTURE-SEARCH-1 performed the most consistently across settings, and its performance was close to that of the best out of the three algorithms in each of the four settings.

7 Variants of the problem

This section discusses variations to the coalition structure generation problem.

7.1 Anytime vs. design-to-time algorithms

In general, one would want to construct an anytime algorithm that establishes a lower k for any amount of search n, compared to any other anytime algorithm. However, such an algorithm might not exist. It is conceivable that the search which establishes the minimal k while searching n' nodes ($n' > n$) does not include all nodes of the search which establishes the minimal k while searching n nodes. This hypothesis is supported by the fact that the curves in Figure 3 cross in the end. However, this is not conclusive because Algorithm 1 might not be the optimal anytime algorithm, and because the bad cases for the splitting algorithm were not shown to be worst cases.

If it turns out that no anytime algorithm is best for all n, one could use information (e.g. exact, probabilistic, or bounds) about the termination time to construct a *design-to-time algorithm [13, 50, 4]* which establishes the lowest possible k for the specified amount of search.

7.2 Off-line vs. on-line search control policies

In this paper we have discussed algorithms that have an *off-line search control* policy, i.e. the nodes to be searched have to be selected without using information accrued from the search so far. With *on-line search control*, one could perhaps establish a lower k with less search because the search can be redirected based on the values observed in the nodes so far.

Especially with on-line search control, it might make a difference whether the search observes only values of coalition structures, $V(CS)$, or values, v_S, of individual coalitions, $S \subseteq A$, in those structures. The latter gives more information. The algorithms presented so far in the paper do not rely on such information: they work in both settings. However, it is conceivable that one could do better by capitalizing on the extra information that is available in the latter setting. As a special case, when an algorithm that observes coalition values, v_S, has searched the bottom two layers, it could, in theory, jump to the optimal coalition structure immediately. However, this shifts the burden of search to the meta-level (search control level): that level still needs to figure out where the optimal coalition structure lies (given the values of the coalitions) so that it can direct the search to jump there.

7.3 Observing coalition structure values, $V(CS)$, vs. observing coalition values, v_S

We now discuss the complexity of determining the desired coalition structure if the algorithm has the luxury of observing values, v_S, of individual coalitions, $S \subseteq A$ instead of merely observing values $V(CS)$ of coalition structures. In such settings, the coalition structure generation problem is the same problem as weighted set packing [18], weighted independent set, weighted maximum clique, and winner determination in combinatorial auctions [36, 31].

7.3.1 Observing coalition values, v_S: Full input

If the algorithm observes the v_S values directly, the optimal coalition structure can be determined using the following dynamic programming algorithm which was originally developed for winner determination in combinatorial auctions [31].

Algorithm 2 (Dynamic programming for coalition structure generation)
INPUT: v_S *for all* $S \subseteq A$. *If no* v_S *is specified then* $v_S = 0$.
OUTPUT: the optimal coalition structure, CS^*.

1. *For all* $x \in A$, *set* $f(x) := v_{\{x\}}$, $\mathcal{C}(\{x\}) := \{x\}$
2. *For* $i := 2$ *to* a, *do:*
   For all $S \subseteq A$ *such that* $|S| = i$, *do:*
   (a) $f(S) := max\{f(S \setminus S') + f(S') : S' \subseteq S$ *and* $1 \leq |S'| \leq \frac{|S|}{2}\}$
   (b) *If* $f(S) \geq v_S$, *then set* $\mathcal{C}(S) := S^*$ *where* S^* *maximizes the right hand side of (a).*
   (c) *If* $f(S) < v_S$, *then set* $f(S) := v_S$ *and* $\mathcal{C}(S) := S$.
3. *Set* $CS^* := \{A\}$.
4. *For every* $S \in CS^*$, *do:*
   If $\mathcal{C}(S) \neq S$, *then*
   (a) *Set* $CS^* := (CS^* \setminus \{S\}) \cup \{\mathcal{C}(S), S \setminus \mathcal{C}(S)\}$.
   (b) *Goto 4 and start with new* CS^*.

Clearly, this algorithm takes $\Omega(2^a)$ time because it looks at all coalitions $S \subseteq A$. It also runs in $O(3^a)$ time [31]. This is significantly less than exhaustive enumeration of all coalition structures. The savings come from the fact that the solutions for the subsets need not be computed over and over again, but only once. In a trivial sense, the algorithm is polynomial. If the input includes a value for every $S \subseteq A$, the input includes 2^a numbers. The algorithm runs in $O(3^a) = O(2^{(\log_2 3)a}) = O((2^a)^{\log_2 3})$ time. Thus the complexity is $O(y^{\log_2 3})$, where y is the number of values in the input.[2]

Put together, if the algorithm has the luxury of observing coalition values v_S directly, it may still be desirable to use our search-based algorithm since it establishes a bound in 2^{a-1} steps while dynamic programming takes significantly longer. However, if it is known that there will be enough time ($O(3^a)$) to run the dynamic programming to completion, it is better to opt to do that than to run our search-based algorithm since the latter has to search the entire graph—i.e. $O(a^a)$ and $\omega(a^{a/2})$ coalition structures—to guarantee that the optimal solution has been found. Note also that the dynamic programming algorithm serves as the optimal ($k = 1$) design-to-time algorithm if it is known that there will be enough time to run it to completion.

7.3.2 Observing coalition values, v_S: Partial input

Another setting arises if the input to the algorithm only includes values of some coalitions, and the unmentioned coalitions have $v_S = 0$ by default. The dynamic programming algorithm will still run the same number of steps, and will therefore not, in general, be polynomial in the size of the input.

It turns out that, unless $\mathcal{P} = \mathcal{NP}$, no algorithm can find the optimal coalition structure in polynomial time in the size of the input in general:

Proposition 2 *Finding the optimal coalition structure is $\mathcal{NP}$-complete.*

Proof. The decision problem is the following: given coalition values, v_S, for some coalitions $S \subseteq A$, and a real number, k, does there exist a coalition structure whose value is at least k?

The problem is in $\mathcal{NP}$ because verifying the value of a solution can be done in polynomial time. Specifically, it only involves summing the values of the coalitions in the structure, and there are at most a coalitions in any coalition structure.

What remains to be shown is that the problem is $\mathcal{NP}$-hard. We prove this by reducing the *set packing* problem to our problem. The set packing decision problem is the following: given a family of sets $\{T_j\}$ and a positive integer q, does $\{T_j\}$ contain q mutually disjoint sets? Karp showed that this problem is $\mathcal{NP}$-complete [21].

We use the following reduction. Let $k = q$. Let $v_S = 1$ for every S that corresponds to a T_j, and let $v_S = 0$ otherwise. Now, a coalition structure with value at least k exists if and only if there exist q mutually disjoint sets in $\{T_j\}$. Thus our

[2]Furthermore, each value, v_S, takes $\log v_S$ bits to represent in the input.

problem is $\mathcal{NP}$-hard. □

Furthermore, in this setting, unless probabilistic polynomial time = $\mathcal{NP}$, no polynomial time algorithm can even establish a bound $k = z^{1-\epsilon}$ for any $\epsilon > 0$ in general where z is the number of coalition values in the input [16, 36]. [3]

Recently, a search-based algorithm was developed for the partial-input setting [36]. Again, it was developed for winner determination in combinatorial auctions, but can be used for optimal coalition structure generation. It constructs a tree where the nodes are coalitions instead of coalition structures as in this paper. It is computationally very efficient if the input is sparse, i.e. if only a small fraction of the coalitions are mentioned in the input ($v_S = 0$ for most $S \subseteq A$). The algorithm does this by provably sufficient selective generation of children in the search tree, by using a secondary search for fast child generation, by heuristics that are relatively accurate and optimized for speed, and by four methods for preprocessing the search space.

7.4 Centralized vs. distributed

Coalition structure generation could occur centrally where one agent tries to find a good coalition structure given the values of all coalitions. In this case the arcs of the coalition structure graph represent deliberative search steps.

Alternatively, coalition structure generation could be a distributed process where no agent knows the values of all coalitions up front, but the coalitions are formed by negotiating over which coalitions should form (and how each coalitions' optimization problem should be solved and how payoff should be divided). In this case, the arcs of the coalition structure graph represent committal agreements to make/break coalitions.

In both settings there is a potential distinction between a deliberative phase where agents search locally for good options, and a committal phase where the agents make binding agreements. For example, under the centralized model, one agent can do the deliberative search, but the outcome might not be imposable on the others without their consent. Similarly, in the distributed model, each agent could do deliberative lookahead into the future of the coalition structure generation process amidst making binding agreements with others.

[3] If the coalitions whose values are mentioned in the input ($v_S \neq 0$) have certain types of additional structure, known polynomial time approximation algorithms for weighted set packing and weighted independent set can be used to establish somewhat lower bounds [5, 14, 17, 15]. These bounds are still so high that they are irrelevant for coalition structure generation in practice. For a recent review, see [36]. If the coalitions that have $v_S \neq 0$ happen to have very particular structure, polynomial time algorithms can solve for the optimal coalition structure [31].

A third method is for the coalition values to be known to all but to distribute the search among the agents. This variant is discussed in Section 8.

7.5 Degree of generality of our results

None of these variants would affect our results that in general, searching the bottom two levels of the coalition structure graph is the unique minimal way to establish a worst case bound, and that the bound is tight. However, the results on searching further might differ in the anytime vs. design-to-time, and off-line vs. on-line search control variants. This is a focus of our future research.

8 Distributing coalition structure search among insincere agents

This section discusses the distribution of coalition structure search across agents (e.g. because the search can be done more efficiently in parallel, and the agents will share the burden of computation) and the methods of motivating self-interested agents to actually follow the desired search method. Self-interested agents prefer greater personal payoffs, so they will search for coalition structures that maximize personal payoffs, ignoring k. In order to motivate such agents to follow a particular search that leads to a socially desirable outcome (e.g. a search that guarantees a desirable worst case bound k), the interaction protocol has to be carefully designed.[4] The following method for distributing search motivates the agents to abide:

1. **Deciding what part of the coalition structure graph to search:** This decision can be made in advance (outside the distributed search mechanism), or be dictated by a central authority, or by a randomly chosen agent,[5] or be decided using some form of negotiation. The earlier results in this paper give prescriptions about which part to search. For example, the agents can decide to search the bottom two levels of the coalition structure graph.

2. **Partitioning the search space among agents:** Each agent is assigned some part of the coalition structure graph to search. The enforcement mechanism, presented later, will motivate the agents to search exactly what they are assigned, no matter how unfairly the assignment is done. One way of achieving *ex*

[4]In some domains, the protocol designer cannot prevent agents from opting out. However such agents receive null excess payoffs since they do not collude with anyone. That is, for $|S| = 1$, the payoff to the agent equals its coalition value v_S. This assumes that agents do not recollude outside the protocol, which may be out of the protocol designer's control.

ante fairness is to randomly allocate the set search space portions to the agents. In this way, each agent searches equally on an expected value basis, although *ex post*, some may search more than others.[5] The fairest option would be to distribute the space so that each agent gets an equal share. Unfortunately the number of search nodes in (the to-be-searched portion of) the coalition structure graph might not be divisible by the number of agents, a, so an exactly equal division of computation may not be possible. Appendix B presents a method for dividing the search space into a disjoint, exhaustive parts—of possibly unequal sizes. If *ex post* equality is required, a payment scheme could be included for compensating those agents that carry out a larger portion of the search.

3. **Actual search:** Each agent searches part of the search space. The enforcement mechanism guarantees that each agent is motivated to search exactly the part of the space that was assigned to that agent. Each agent, having completed the search, tells the others which CS maximized $V(CS)$ in its search space.

4. **Enforcement of the protocol:** One agent, i, and one search space of an agent j, $j \neq i$, will be selected at random.[5] Agent i will re-search the search space of j to verify that j has performed its search as required. Agent j gets caught of mis-searching if i finds a better CS in j's space than j reported (or i sees that the CS that j reported does not belong to j's space at all). If j gets caught, it has to pay a penalty P. To motivate i to conduct this additional search, i should be made the claimant of P.

   There is no pure strategy Nash equilibrium in this mechanism.[6] If i searches any given node in j's space, and the penalty is high enough, then j is motivated to search that node. However, then i is not motivated to search that node since it cannot receive P. On the other hand, if i skips a given node in j's space, j is motivated to skip it as well (unless searching that node has a high chance of improving the best solution found by all agents so much that j's share of that improvement outweighs the search cost). However, then i is motivated to search that node since it would receive P.

   Instead, there can be a mixed strategy Bayes-Nash equilibrium where i and j search nodes with some probabilities. By increasing P, the probability that j searches all of the nodes in its space could be made close to one. The probability

[5]The randomization can be done without a trusted third party by using a recent distributed nonmanipulable protocol for randomly permuting the agents [51]. Distributed randomization is also discussed in [24].

[6]In short, agents' strategies are said to be in Nash equilibrium if each agent's strategy is a best response to the strategies of the others [27, 26].

that i searches would go close to zero, which minimizes enforcement overhead.[7]

5. **Additional search:** The previous steps of this distributed mechanism can be repeated if more time to search remains. For example, the agents could first do step 1 of Algorithm 1. Then, they could repeatedly search more and more as time allows, again using the distributed method.

6. **Payoff division:** Many alternative methods for payoff division among agents could be used here. The only concern is that the division of $V(CS)$ may affect what CS an agent wants to report as a result of its search since different CSs may give the agent different payoffs (depending on the payoff division scheme). However, by making P high enough compared to the $V(CS)$ values, this consideration can be made negligible compared to the risk of getting caught.

9 Related research

Coalition formation has been widely studied in game theory [20, 2, 1, 3, 26, 29, 48]. However, most of that work has not taken into account the computational limitations involved. This section reviews some of the research that has been done on the computational aspects. We will discuss payoff division among agents, coalition structure generation, and optimization within each coalition.

9.1 Payoff division

The bulk of the game theoretic coalition formation work focuses on the question of how to divide $V(CS^*)$ among agents so as to achieve stability of the payoff configuration. Several different notions of stability in characteristic function games have been proposed, see [20] for a review. In most cases, computing a stable payoff configuration is intractable.

Transfer schemes represent a dynamic approach to the payoff division activity of coalition formation in CFGs [20, 46]. The agents stay within a given coalition structure and iteratively exchange payments in a prespecified manner. Again, there is no guarantee that a self-interested agent would be best off by using the specified

[7]Agent j will try to trade off the cost of search against the risk of getting caught, and could decide that the risk is worth taking. Similarly, agent i will try to trade off the cost of re-searching against the chance of catching j. This problem can be minimized by choosing a high enough P. A further complication arises from the fact that an agent can decide whether to search further in its space—and where to search within that space—based on the results of its search so far. These result signal to the agent about how good the unsearched coalition structures would be. Due to this update, the equilibrium would not be a basic mixed strategy Nash equilibrium, but a mixed strategy Bayes-Nash equilibrium. The detailed analysis of this equilibrium is part of our current research.

local strategy: by using some other strategy, an agent may be able to drive the negotiation to a final solution that is better for the agent. Transfer schemes address the payoff distribution activity but not the optimization activity or coalition structure generation.

For example, a transfer scheme for the *core* solution concept has been developed [49].[8] This scheme alternates between two operators. In the first, an agent's payoff is incremented by its coalition's excess (value of the coalition minus the sum of the members' current payoffs) divided by the number of agents in the coalition. In the second, every agent's payoff is decremented equally, just enough to keep the total payoff vector feasible. The method can be implemented in a largest-excess-first manner, or in a round-robin manner among agents. Both schemes converge to a payoff vector in the core, if the core is nonempty, i.e. if such a stable payoff vector exists.

Transfer schemes reduce the cognitive demands placed on the agents. For example, in the case of the core, the agents do not need to search for a payoff vector that satisfies the numerous constraints in the definition of the core. Instead they can simply follow the transfer scheme until a payoff division in the core has been reached.

Transfer schemes could be used in conjunction with our work. Our approach would be used for coalition structure generation, and a transfer scheme could be used for dividing the resulting value of the coalition structure among agents.

Zlotkin and Rosenschein [51] analyze payoff division in "Subadditive Task Oriented Domains" (STODs), which are a strict subset of CFGs. In their work, the coalition structure generation is trivial since the agents always form the grand coalition. Specifically, one agent handles the tasks of all agents. In STODs this is optimal because STODs never exhibit diseconomies of scale. Their method guarantees each agent an expected value that equals its Shapley value [42]. The Shapley value is a specific payoff division among agents that motivates individual agents to stay in the coalition structure. The group of all agents is also motivated to stay in the coalition structure. Unlike the core, the Shapley value does not in general motivate every subgroup of agents to stay. In a subset of STODs, "Concave Task Oriented Domains", the Shapley value also motivates every subgroup to stay, i.e. that payoff configuration is in the core [51].

A naive method that guarantees each agent an expected payoff equal to the Shapley value has exponential complexity in the number of agents, but Zlotkin and Rosenschein present a novel way of achieving this with linear complexity in the number of agents. Each agent gets paid its marginal contribution to the coalition. Because this depends on the order in which the agents join the coalition, the joining order is randomized. The randomization is carried out in a distributed nonmanipulable way

[8] A payoff configuration is in the core if no subgroup of agents can achieve higher payoff by moving out of the payoff configuration and forming a coalition of their own [20].

so as to avoid the need for a trusted third party to carry out the randomization.

Zlotkin and Rosenschein's payoff division method could be used in conjunction with the coalition structure generation methods of this paper. The Shapley value is well-defined for every coalition structure, not only the grand coalition. Therefore, our methods could be used to choose the coalition structure, and the agents could get paid their marginal contribution to the coalition structure. Again, the joining order would be randomized. This would give each agent an expected payoff equal to its Shapley value.

Recently Deb et al [9] investigated games where all effective coalitions must contain at least q agents, where q is a constant, $1 \leq q \leq a$. It is also assumed that there is a finite number of payoff configurations. They determine an upper bound on the size of the space of payoff configurations that guarantees existence of a stable coalition structure. They do not address the problem of coalition structure generation, nor how to restrict the space of payoff configurations. They also do not discuss bounds on solution quality.

9.2 Coalition structure generation

Shehory and Kraus [43] present an algorithm for coalition structure generation among cooperative—social welfare maximizing, i.e. not self-interested—agents. Among such agents the payoff distribution is a non-issue and is thus not addressed. The distributed algorithm forms disjoint coalitions—which by their definition can only handle one task each—and allocates tasks to the coalitions. Recently Shehory and Kraus have extended this work to overlapping coalitions and coalitions that can jointly handle more than one task [45]. The complexity of the problem is reduced by limiting the number of agents per coalition. The greedy algorithm guarantees that the solution is within a loose ratio bound from the best solution that is possible *given the limit on the number of agents*. However, this benchmark can, itself, be arbitrarily far from optimum. On the other hand, our work computes the bound based on the actual optimum.

Our result that no algorithm can establish a bound while searching less than 2^{a-1} nodes does not apply to their setting because they are not solving CFGs. First, their v_S values have special structure. Second, they have dependencies between v_S values. Third, in their work, a given coalition may have several values that correspond to different tasks. While the third difference may cause more complexity than is present in CFGs, the first difference may allow a worst case bound to be established with less search than in general CFGs. In CFGs where the v_S values are known to satisfy additional constraints, it may be possible to exploit such structure and establish a worst case bound with less search than in general CFGs.

De Vany [10] uses information theory to measure the complexity of coalition structures. He observes that finding the optimal coalition structure rapidly becomes intractable as the number of agents grows. He therefore concludes that "almost optimal" coalition structures should be sought instead, and proposes this as fruitful future research. Unlike our work, he does not present algorithms for doing this.

9.3 Combining coalition structure generation and payoff division

While payoff division and coalition structure generation are important problems to study in isolation, it would be desirable to extend those studies by focusing on both of these activities simultaneously as well. This is important from the perspective of many practical applications since in many settings these two activities interact. Among self-interested agents, the coalition that an agent wants to join often depends on the agent's payoffs in alternative coalitions.

9.3.1 Non-normative approaches

One strand of research in this field is focused on asking the question of what types of outcomes follow if the agents follow particular strategies.

Early work in this field includes Friend's program that simulates a 3-agent coalition formation situation where agents can make offers, acceptances, and rejections to each other regarding coalitions and payoffs [12, 20]. In the model, at most one offer regarding each agent can be active at a time. A new offer makes old offers regarding that agent void. Players consider only current proposals: they are assumed to do no lookahead or to have no memory. The negotiations terminate when two agents have reached a dyad and the third one has given up. Specifically, the termination criterion is based on a local threat-counterthreat examination: an agent does not necessarily accept a new better offer if that introduces a risk of being totally excluded in the next step. The model is not normative: there is no guarantee that a self-interested agent would be best off by using the specified local strategy.

Ketchpel [22] presents a coalition formation method which addresses coalition structure generation as well as payoff distribution. These are handled simultaneously. His algorithm uses cubic time in the number of agents, but each individual step may be very complex due to an inefficient pairwise auction protocol. His coalition formation method guarantees neither a bound from optimum nor stability of the coalition structure (which is normally achieved via appropriate payoff division). There is no mechanism for motivating self-interested agents to follow his algorithm.

Ketchpel's method is related to a contracting protocol of Sandholm [33, 39, 34] (TRACONET) where agents construct the global solution by reallocating a small number of tasks among themselves at a time. Payments are made regarding each such contract before new contracts take place. An agent updates its approximate local solution after each task transfer.

Shehory and Kraus [44] analyze coalition formation among self-interested agents with perfect information in CFGs. Their protocol guarantees that if agents follow it (nothing necessarily motivates them to do so), a certain stability criterion, kernel-stability, is met. Their other protocol reduces the complexity somewhat by requiring only a weaker form of stability, polynomial kernel-stability. Their algorithm switches from one coalition structure to another and guarantees improvements at each step: it is an anytime algorithm. However, it does not guarantee a bound from optimum.

9.3.2 Normative approaches

Another strand of research in this field asks the question of what happens if each agent uses the strategy that is best for itself. In other words, agents are assumed to act rationally instead of using some particular ad hoc strategies. The fact that ad hoc strategies are not acceptable introduces the analytical burden of showing what the best strategy for each agent is. This may depend on the strategies of other agents. Such settings can be analyzed using solution concepts from noncooperative game theory. This area of study is called coalitional bargaining. It addresses both coalition structure generation and payoff division. Coalitional bargaining can be seen as a generalization of the Rubinstein alternating offers bargaining model [32]. A typical model has agents in a characteristic function setting sequentially making proposals to the group. A proposal consists of a possible coalition to be formed and a payoff vector determining how the value of the coalition would be divided among the members. Unanimous agreement among the members of the proposed coalition leads to that coalition being formed, otherwise no coalition is formed and another proposal is made. Time discounting is usually included in the model.

Chatterjee et al [6] discuss efficiency properties of stationary equilibria of strictly superadditive games. Responses to a proposal are always done in the order defined in the protocol and the first agent to reject a proposal becomes the next agent allowed to make a proposal. They show that inefficiencies can arise in the form of delay and non-formation of the grand coalition. The protocol, or ordering of agents, significantly affects the efficiency of the outcome, and grants different agents differing amounts of power.

Okada [28] studies a similar setting, but is interested in how efficiency of agreement is affected by the bargaining procedure, in particular by the rule governing the selection of proposers. Unlike the work of Chatterjee et al, the proposer is chosen randomly with equal probability from among the group of remaining agents. It is shown that no delay of agreement occurs in equilibrium for superadditive games and if players are patient enough, the grand coalition is formed with an equal payoff allocation if and only if it has the largest value per capita among all coalitions.

Evans [11] investigates a situation where there is no time discounting and where the protocol does not specify which agent will make a proposal. Instead, at each round of bargaining, agents compete for the right to make a proposal by investing resources. The agent that invests the most in a round is awarded the ability to make the proposal for that round. A player's final payoff is the resources awarded to it in the final agreement minus the amount spent in the proposal competition. In this game, pure stationary subgame perfect equilibrium payoffs coincide with the core, if the core exists. When a time discount is included, no stationary pure strategy equilibria exist.

Put together, on the positive side, coalitional bargaining does not make ad hoc assumptions about the agents' behavior. On the negative side, the computational complexity has usually not been addressed.

9.4 Optimization within each coalition

Sandholm and Lesser [41, 38] study coalition formation with a focus on the optimization activity: how do computational limitations affect which coalition structure should form, and whether that structure is stable? That work used a normative model of bounded rationality based on the agents' algorithms' performance profiles and the unit cost of computation. All coalition structures were enumerated because the number of agents was relatively small, but it was not assumed that they could be evaluated exactly because the optimization problems could not be solved exactly due to intractability. The methods of this paper can be combined with their work if the performance profiles are deterministic. In such cases, the v_S values represent the value of each coalition, given that the coalition would strike the optimal tradeoff between quality of the optimization solution and the cost of that computation. Our algorithm can be used to search for a coalition structure, and only afterwards would the coalitions in the chosen coalition structure actually attack their optimization problems. If the performance profiles include uncertainty, this separation of coalition structure generation and optimization does not work e.g. because an agent may want to re-decide its membership if its original coalition receives a worse optimization solution than expected.

9.5 Combining coalition structure generation, optimization within each coalition, and payoff division

As discussed above, prior work on computational coalition formation has reduced the complexity of coalition structure generation (in the number of agents), payoff division (in the number of agents), or the optimization activity (in the size of the coalitions' optimization problems). However, to our knowledge, no work up to date has reduced the complexity of these three activities simultaneously. We postulate this as a desirable direction for future research. Above we also presented steps toward this direction by discussing ways of combining our coalition structure generation methods with existing methods for reducing the complexity of payoff division or optimization.

10 Conclusions and future research

Coalition formation is a key topic in multiagent systems. One may prefer a coalition structure that maximizes the sum of the values of the coalitions. However, often the number of coalition structures is too large to allow exhaustive search for the optimal one. Furthermore, the problem is $\mathcal{NP}$-complete, so unless $\mathcal{P} = \mathcal{NP}$, no algorithm can find the optimal coalition structure in time that is polynomial in the size of the input (which itself is exponential because it includes the values of all coalitions). This paper focused on establishing a worst case bound on the quality of the coalition structure while only searching a small fraction of the coalition structures.

We showed that none of the prior coalition structure generation algorithms for general characteristic function games can establish any bound because they search fewer nodes than a threshold that we showed necessary for establishing a bound (in special cases where the coalition values have additional structure, it may be possible to establish a bound with less search). We presented an algorithm that establishes a tight bound within this minimal amount of search, and showed that any other algorithm would have to search strictly more. The fraction of nodes needed to be searched approaches zero as the number of agents grows.

If additional time remains, our anytime algorithm searches further, and establishes a progressively lower tight bound. Surprisingly, just searching one more node drops the bound in half. As desired, our algorithm lowers the bound rapidly early on, and exhibits diminishing returns to computation. It also drastically outperforms its obvious contenders: the merging algorithm and the splitting algorithm. Finally, we showed how to distribute the desired search across self-interested manipulative agents.

Our results can also be used as prescriptions for designing negotiation protocols for coalition structure generation. To optimize worst case performance, the agents should not start with everyone operating separately—as one might do intuitively. Instead, they should start from the grand coalition, and consider different ways of splitting off exactly one coalition. After that, they should try everyone operating separately, and can continue from there by considering mergers of two coalitions at a time.

Future research includes studying design-to-time algorithms and on-line search control policies for coalition structure generation. We are also analyzing the interplay of dynamic coalition formation and belief revision among bounded-rational agents [47]. When coalition values have uncertainty, agents may want to redecide their coalitions. The design of applicable backtracking methods for self-interested agents is nontrivial. In the future we plan to extend Sandholm and Lesser's non-manipulable leveled commitment contracts [40, 34] to coalition formation deals as one possible way of implementing backtracking in this setting. The long term goal is to construct normative methods that reduce the complexity of all three activities of coalition formation simultaneously: coalition structure generation, optimization within each coalition, and payoff division.

A Appendix: number of coalition structures

Proof. (Proposition 1). We first prove that the number of coalition structures is $O(a^a)$. Let there be a agents. Let there be a set of *locations* where coalitions can form, at most one coalition per location. Let the number of locations be a. This allows for any coalition structure to form since a coalition structure can have at most a coalitions. Now, say that the agents get placed in locations one agent at a time. Each agent could be placed in a different locations, and there are a agents to place. Therefore, the number of possible overall assignments is a^a. Thus, the number of coalition structures is $O(a^a)$.[9]

What remains to be proven is that the number of coalition structures is $\omega(a^{a/2})$. We use the following lemma in the proof.

Lemma 2 *Let a and b be positive integers. Then $\lceil \frac{a}{b} \rceil ! \leq (\frac{a}{b})^{\frac{a}{b}}$ for $\frac{a}{b} \geq 2$.*

Proof.

$$\begin{aligned}
\frac{\lceil \frac{a}{b} \rceil !}{(\frac{a}{b})^{\frac{a}{b}}} \leq \frac{\frac{a}{b}+1}{\frac{a}{b}} \cdot \frac{\frac{a}{b}}{\frac{a}{b}} \cdot \frac{\frac{a}{b}-1}{\frac{a}{b}} \cdots \frac{2}{\frac{a}{b}} &\leq \frac{\frac{a}{b}+1}{\frac{a}{b}} \cdot \frac{\frac{a}{b}-1}{\frac{a}{b}} \\
&= (1+\frac{b}{a})(1-\frac{b}{a}) \\
&= 1-(\frac{b}{a})^2 \\
&\leq 1
\end{aligned}$$

Therefore $\lceil \frac{a}{b} \rceil ! \leq (\frac{a}{b})^{\frac{a}{b}}$. □

[9]Note that a^a overestimates the number of coalition structures. Some coalition structures are counted multiple times because for any given coalition in the structure, the structure is counted once for each location where the coalition can form (although it should be counted only once).

One way to count the number of coalitions structures is to use *Bell numbers*. The Bell number B_a is equal to the number of ways a set of a elements can be partitioned into nonempty subsets. That is, B_a is equal to the number of coalition structures that can be generated with a agents. There are several ways of calculating Bell numbers, including Dobinski's formula [7]:

$$B_a = \frac{1}{e} \sum_{i=0}^{\infty} \frac{i^a}{i!}$$

To show that the number of coalition structures is $\omega(a^{a/2})$ it suffices [8] to show that

$$\lim_{a \to \infty} \frac{B_a}{a^{a/2}} = \infty$$

Since each term in the series of Dobinski's formula is positive, it suffices to take one term, $\frac{i^a}{i!}$ and show that

$$\lim_{a \to \infty} \frac{\frac{i^a}{i!}}{a^{a/2}} = \infty$$

Set $i = \lceil \frac{a}{b} \rceil$ for some constant b, where $b > 2$. Then the expression we are interested in is

$$\begin{aligned} \frac{(\lceil \frac{a}{b} \rceil)^a}{(\lceil \frac{a}{b} \rceil)! a^{a/2}} &\geq \frac{(\frac{a}{b})^a}{(\lceil \frac{a}{b} \rceil)! a^{a/2}} \\ &\geq \frac{(\frac{a}{b})^a}{(\frac{a}{b})^{a/b} a^{a/2}} \\ &= \frac{a^{\frac{a(b-2)}{2b}}}{b^{\frac{a(b-1)}{b}}} \end{aligned}$$

We now calculate

$$\lim_{a \to \infty} \frac{a^{\frac{a(b-2)}{2b}}}{b^{\frac{a(b-1)}{b}}}$$

Since the natural logarithm and exponential functions are continuous, we can calculate the limit as follows. Set

$$\begin{aligned} \phi(a) &= \ln \frac{a^{\frac{a(b-2)}{2b}}}{b^{\frac{a(b-1)}{b}}} \\ &= \frac{a(b-2)}{2b} \ln a - \frac{a(b-1)}{b} \ln b \\ &= \frac{a}{b} [\frac{b-2}{2} \ln a - (b-1) \ln b] \end{aligned}$$

Then

$$\begin{aligned}
\lim_{a\to\infty} \frac{a^{\frac{a(b-2)}{2b}}}{b^{\frac{a(b-1)}{b}}} &= \lim_{a\to\infty} e^{\phi(a)} \\
&= \lim_{a\to\infty} e^{\frac{a}{b}} e^{\frac{b-2}{2}\ln a-(b-1)\ln b} \\
&= \lim_{a\to\infty} e^{\frac{a}{b}} \lim_{a\to\infty} e^{\frac{b-2}{2}\ln a-(b-1)\ln b} \\
&= \infty\cdot\infty \\
&= \infty
\end{aligned}$$

Thus, we have shown that $B_a \in \omega(a^{\frac{a}{2}})$. □

B Appendix: dividing the search space among agents

In Section 8 we presented a nonmanipulable method for distributing the search among agents. To allow the agents to perform the distributed search, it is necessary to partition any given search space $M' \subseteq M$ among them. This appendix presents a method for choosing which agent searches which portion of the space. It would be desirable to partition the space so that the agents' portions are exhaustive (i.e. they cover M', the part of the overall space that the agents have decided to cover), disjoint (i.e. there is no redundant search), and of the same size (each agent searches equally many coalition structures[10]). Unfortunately it may not be possible to divide the space equally because, for example, $|M'|$ might not be divisible by a.

It would also be desirable to define the coalition structures that an agent needs to search without actually enumerating those coalition structures. This is because enumeration can be almost as costly as the search itself.

This appendix presents a non-enumerative method for dividing the space exhaustively and disjointly, albeit unequally in general. Giving the entire search to one agent would achieve this trivially. On the other hand, our method distributes the search more evenly,[11] and assigns each agent the same *expected* amount of search. The method works as follows:

1. The agents draw a random permutation of the numbers $1 \ldots a$ (again, the randomizations can be done via a trusted third party, or using a nonmanipulable distributed method [51]). This permutation sets a size ordering criterion for coalitions in coalition structures. For example, if $a = 5$ and the permutation

[10] This is desirable if each coalition structure requires the same amount of computation.

[11] It is possible to decrease the size differences of the portions further by attaching non-uniform probabilities to the outcomes in the randomizations.

$P_1 = 32154$ is drawn, then coalitions of size 3 will appear first in coalition structures, coalitions of size 2 will appear second, *etc.*

2. For each coalition size the agents draw a random permutation. According to this permutation the agents within the coalition are ordered. For example, if $P_2 = 25341$ is drawn for coalition size three, then the coalition of agents 1, 3, and 5 will be ordered 531.

   These orderings give implicit names to the coalitions. These names can be used to break ties regarding step 1: coalitions of the same size can be ordered, for example, in lexicographically increasing order within a coalition structure. However, this method gives significantly more search to agents with small indices. Therefore we suggest that the lexicographic comparison is done on the reversed name of the coalition. For example, for the coalition 531, the key to compare would be 135. This still imposes some more search on agents with small indices, but only if the lexicographic comparison has to use each digit of the name. Furthermore, this slight *ex post* unfairness does not affect *ex ante* equality because in the main algorithm (Section 8), the search portions get randomly assigned to agents anyway. In other words, the agents get randomly assigned indices.

3. Each agent is responsible for searching coalition structures where it is the first agent of the first coalition. For example, assume that P_1 and P_2 were drawn as above. Now, coalition structure $\{541, 23\}$ should be searched by agent 5.

The complexity of this assignment scheme is very low compared to the complexity of the actual coalition structure generation search. Each distributed random permutation takes only $O(a^2)$ operations, and our scheme uses a such permutations (one in step 1, and one for each coalition size in step 2 (except that coalitions of size one do not need a permutation)).

Acknowledgments

This material is based upon work supported by the National Science Foundation under CAREER Award IRI-9703122, Grant IRI-9610122, and Grant IIS-9800994.

References

[1] R. Aumann. Acceptable points in general cooperative n-person games. volume IV of *Contributions to the Theory of Games*. Princeton University Press, 1959.

[2] B. D. Bernheim, B. Peleg, and M. D. Whinston. Coalition-proof Nash equilibria: I concepts. *Journal of Economic Theory*, 42(1):1–12, June 1987.

[3] B. D. Bernheim and M. D. Whinston. Coalition-proof Nash equilibria: II applications. *Journal of Economic Theory*, 42(1):13–29, June 1987.

[4] M. Boddy and T. Dean. Deliberation scheduling for problem solving in time-constrained environments. *Artificial Intelligence*, 67:245–285, 1994.

[5] B. Chandra and M. M. Halldórsson. Greedy local search and weighted set packing approximation. In *10th Annual SIAM-ACM Symposium on Discrete Algorithms (SODA)*, Jan. 1999. To appear.

[6] K. Chatterjee, B. Dutta, D. Ray, and K. Sengupta. A noncooperative theory of coalitional bargaining. *Review of Economic Studies*, 60:463–477, 1993.

[7] L. Comtet. *Advanced Combinatorics*. D. Reidel Pub. Co., 1974.

[8] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

[9] R. Deb, S. Weber, and E. Winter. The Nakamura theorem for coalition structures of quota games. *International Journal of Game Theory*, 25(2):189–198, 1996.

[10] A. DeVany. Information, bounded rationality, and the complexity of economic organization. *Taiwan Journal of Political Economy*, 1, 1996.

[11] R. Evans. Coalitional bargaining with competition to make offers. *Games and Economic Behavior*, 19:211–220, 1997.

[12] K. E. Friend. *An information processing approach to small group interaction in a coalition formation game*. PhD thesis, Carnegie-Mellon University, 1973.

[13] A. Garvey and V. Lesser. Design-to-time real-time scheduling. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(6):1491–1502, 1993.

[14] M. M. Halldórsson. Approximations of independent sets in graphs. In K. Jansen and J. Rolim, editors, *The First International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 1–14, Aalborg, Denmark, July 1998. Springer LNCS 1444.

[15] M. M. Halldórsson and H. C. Lau. Low-degree graph partitioning via local search with applications to constraint satisfaction, max cut, and 3-coloring. *Journal of Graph Algo. Applic.*, 1(3):1–13, 1997.

[16] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 1999. To appear. Draft: Royal Institute of Technology, Sweden, 8/11/98. Early version: Proc. 37th IEEE Symposium on Foundations of Computer Science (1996), 627—636.

[17] D. S. Hochbaum. Efficient bounds for the stable set, vertex cover, and set packing problems. *Discrete Applied Mathematics*, 6:243–254, 1983.

[18] D. S. Hochbaum. *Approximation algorithms for NP-hard problems*. PWS Publishing Company, 1997.

[19] E. Horvitz. Reasoning about beliefs and actions under computational resource constraints. In *Proceedings of Third Workshop on Uncertainty in Artificial Intelligence*, pages 429–444, Seattle, Washington, July 1987. American Association for Artificial Intelligence. Also in L. Kanal, T. Levitt, and J. Lemmer, ed., *Uncertainty in Artificial Intelligence 3,* Elsevier, 1989, pps. 301-324.

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

[21] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, NY, 1972.

[22] S. Ketchpel. Forming coalitions in the face of uncertain rewards. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 414–419, Seattle, WA, July 1994.

[23] K. S. Larson and T. W. Sandholm. Anytime coalition structure generation: An average case study. In *Proceedings of the Third International Conference on Autonomous Agents (AGENTS)*, pages 40–47, Seattle, WA, May 1999.

[24] N. Linial. Games computers play: Game-theoretic aspects of computing. Technical Report 92-5, Leibniz Center for Computer Science, Hebrew University, Jerusalem, 1992.

[25] M. G. Lundgren, K. Jörnsten, and P. Värbrand. On the nucleolus of the basic vehicle routing game. Technical Report 1992-26, Linköping Univ., Dept. of Mathematics, Sweden, 1992.

[26] A. Mas-Colell, M. Whinston, and J. R. Green. *Microeconomic Theory.* Oxford University Press, 1995.

[27] J. Nash. Equilibrium points in n-person games. *Proc. of the National Academy of Sciences*, 36:48–49, 1950.

[28] A. Okada. A noncooperative coalitional bargaining game with random proposers. *Games and Economic Behavior*, 16:97–108, 1996.

[29] H. Raiffa. *The Art and Science of Negotiation.* Harvard Univ. Press, Cambridge, Mass., 1982.

[30] J. S. Rosenschein and G. Zlotkin. *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers.* MIT Press, 1994.

[31] M. H. Rothkopf, A. Pekeč, and R. M. Harstad. Computationally manageable combinatorial auctions. *Management Science*, 44(8):1131–1147, 1998.

[32] A. Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica*, 50:97–109, 1982.

[33] T. W. Sandholm. An implementation of the contract net protocol based on marginal cost calculations. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 256–262, Washington, D.C., July 1993.

[34] T. W. Sandholm. *Negotiation among Self-Interested Computationally Limited Agents.* PhD thesis, University of Massachusetts, Amherst, 1996. Available at http:// www.cs.wustl.edu/ ~sandholm/ dissertation.ps.

[35] T. W. Sandholm. Unenforced E-commerce transactions. *IEEE Internet Computing*, 1(6):47–54, Nov–Dec 1997. Special issue on Electronic Commerce.

[36] T. W. Sandholm. An algorithm for optimal winner determination in combinatorial auctions. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, Stockholm, Sweden, 1999. Extended version: Washington University, Department of Computer Science technical report WUCS-99-01.

[37] T. W. Sandholm, K. S. Larson, M. R. Andersson, O. Shehory, and F. Tohmé. Anytime coalition structure generation with worst case guarantees. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 46–53, Madison, WI, July 1998.

[38] T. W. Sandholm and V. R. Lesser. Coalition formation among bounded rational agents. In *Proceedings of the Fourteenth International Joint Conference on*

Artificial Intelligence (IJCAI), pages 662–669, Montreal, Canada, Aug. 1995. Extended version appeared as University of Massachusetts at Amherst, Computer Science Department technical report 95-71.

[39] T. W. Sandholm and V. R. Lesser. Issues in automated negotiation and electronic commerce: Extending the contract net framework. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS)*, pages 328–335, San Francisco, CA, June 1995. Reprinted in *Readings in Agents*, Huhns and Singh, eds., pp. 66–73, 1997.

[40] T. W. Sandholm and V. R. Lesser. Advantages of a leveled commitment contracting protocol. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 126–133, Portland, OR, Aug. 1996. Extended version: University of Massachusetts at Amherst, Computer Science Department technical report 95-72.

[41] T. W. Sandholm and V. R. Lesser. Coalitions among computationally bounded agents. *Artificial Intelligence*, 94(1):99–137, 1997. Special issue on Economic Principles of Multiagent Systems. Early version appeared at the International Joint Conference on Artificial Intelligence, pages 662–669, 1995.

[42] L. S. Shapley. A value for n-person games. In H. W. Kuhn and A. W. Tucker, editors, *Contributions to the Theory of Games*, volume 2 of *Annals of Mathematics Studies, 28*, pages 307–317. Princeton University Press, 1953.

[43] O. Shehory and S. Kraus. Task allocation via coalition formation among autonomous agents. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 655–661, Montreal, Canada, Aug. 1995.

[44] O. Shehory and S. Kraus. A kernel-oriented model for coalition-formation in general environments: Implemetation and results. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 134–140, Portland, OR, Aug. 1996.

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

[46] R. E. Stearns. Convergent transfer schemes for n-person games. *Transactions of the American Mathematical Society*, 134:449–459, 1968.

[47] F. Tohmé and T. W. Sandholm. Coalition formation processes with belief revision among bounded rational self-interested agents. *Journal of Logic and Computation*, 9(97–63):1–23, 1999. An early version appeared in the IJCAI-97 Workshop on Social Interaction and Communityware, pp. 43–51, Nagoya, Japan.

[48] W. J. van der Linden and A. Verbeek. Coalition formation: A game-theoretic approach. In H. A. M. Wilke, editor, *Coalition Formation*, volume 24 of *Advances in Psychology*. North Holland, 1985.

[49] L. S. Wu. A dynamic theory for the class of games with nonempty cores. *SIAM Journal of Applied Mathematics*, 32:328–338, 1977.

[50] S. Zilberstein and S. Russell. Optimal composition of real-time systems. *Artificial Intelligence*, 82(1–2):181–213, 1996.

[51] G. Zlotkin and J. S. Rosenschein. Coalition, cryptography and stability: Mechanisms for coalition formation in task oriented domains. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 432–437, Seattle, WA, July 1994.