J. EXPT. THEOR. ARTIF. INTELL. 12(2000)23–42

Anytime coalition structure generation: an average case study

KATE S. LARSON and TUOMAS W. SANDHOLM

Department of Computer Science, Washington University, One Brookings Drive, St. Louis, MO 63130-4899, USA
email: {ksl2, sandholm}@cs.wustl.edu

Abstract. Coalition formation is a key topic in multiagent systems. One would 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 for exhaustive search for the optimal one. We present experimental results for three anytime algorithms that search the space of coalition structures. We show that, in the average case, all three algorithms do much better than the recently established theoretical worst case results in Sandholm *et al.* (1999a). We also show that no one algorithm is dominant. Each algorithm's performance is influenced by the particular instance distribution, with each algorithm outperforming the others for different instances. We present a possible explanation for the behaviour of the algorithms and support our hypothesis with data collected from a controlled experimental run.

Keywords: coalition structure, algorithm, multiagent systems

1. Introduction

Multiagent systems with self-interested agents are becoming increasingly important. One reason for this is the technology push of a growing standardized communication infrastructure—Internet, WWW, 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. 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. In many multiagent settings, self-interested agents representing real world parties can operate more effectively by forming coalitions and coordinating their activities and pooling resources within each coalition. This may allow tasks to be completed that otherwise would not have been possible. For example, there is 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 labour 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.

Journal of Experimental & Theoretical Artificial Intelligence ISSN 0952-813X print/ISSN 1362-3079 online

1.1. *The three activities of coalition formation*

In many domains, self-interested 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 is the formation of coalitions by the agents such that agents within each coalition coordinate their activities, but agents do not coordinate between coalitions. 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 this joint problem. A 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.

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 limited time. Instead, agents have to select a subset of coalition structures on which to focus their search. Sandholm *et al.* (1999a) studied 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 optimal coalition structure. In that paper they presented three algorithms for searching for the optimal coalition structure and determined worst case (or bad case) results for bounds. In this paper we conduct an empirical study of the three search algorithms in order to see how they behave on average.

The rest of the paper is organized as follows. Section 2 discusses coalition structure generation in characteristic function games. The next two sections describe the algorithms that were studied and the setup of the experiments. Section 5 discusses the results from the experiments and Section 6 proposes possible explanations for the results. The paper concludes with a description of related research and directions for future research.

2. Coalition structure generation in characteristic function games

Let A be the set of agents, and $a = |A|$. As is common practice (Sandholm and Lesser 1997, Shehory and Kraus 1996, Ketchpel 1994, Zlotkin and Rosenschein 1994, Kahan and Rapoport 1984, Raiffa 1982, Wu 1977, Stearns 1968, Shapley 1953) 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 its problem (Sandholm and Lesser 1997).)

Not all settings are CFGs. In CFGs, each coalition S has some value v_S, i.e. each coalition's value is independent of non-members' actions. However, in general the value of a coalition may depend on non-members' actions due to positive and negative externalities (interactions of the agents' solutions). Negative externalities between a coalition and non-members are often caused by shared resources. Once non-members 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, non-members may actually move the world further from the coalition's goal state(s) (Rosenschein and Zlotkin 1994). Positive externalities are often caused by partially overlapping goals. In satisfying their goals, non-members 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 non-members. General settings with possible externalities can be modelled as *normal form games* (NFGs). CFGs are a strict subset of NFGs. However, many real-world multiagent problems happen to be CFGs (Sandholm and Lesser 1997).

We assume that each coalition's value is non-negative:

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

This is not an unreasonable assumption since if some coalitions' values are negative but bounded from below, one can normalize the coalition values by subtracting $\min_{S \subset A} v_S$ from all coalition values v_S. This rescales the coalition values so that Equation 1 holds for all coalitions. The rescaled game is strategically equivalent to the original game (Kahan and Rapoport 1984).

A coalition structure, CS, is a partition of agents, A, into disjoint, exhaustive coalitions. 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 the sum of the values of the coalitions in it:

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

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*

The problem of finding the optimal coalition structure is computationally complex. First, the input of the problem is exponential in the number of agents. The input to a coalition structure generation algorithm consists of the values of the coalitions, v_S. One value is associated with each coalition and there are $2^a - 1$ coalitions. Secondly, the number of coalition structures grows rapidly as the number of agents increase. The number of coalition structures is $O(a^a)$ and $\omega(a^{a/2})$ as shown in Sandholm *et al.* (1999a). Exhaustive enumeration is not a viable method for searching for the

optimal coalition structure unless the number of agents is small (less than 15 or so in practice). If the input to the optimization problem includes only the coalitions with strictly positive value, no algorithm can find the optimal coalition structure in polynomial time in the size of the input because the problem is $\mathcal{NP}$-complete (Sandholm *et al.* 1999a).

2.2. *Lack of prior attention*

Coalition structure generation has not received much attention previously. Research has focused (Zlotkin and Rosenschein 1994, Lundgren *et al.* 1992, Van der Linden and Verbeek 1985, Kahan and Rapoport 1984, Raiffa 1982) on superadditive games, i.e. games where $v_{S\cup T} \geqslant v_S + v_T$ for all disjoint coalitions $S, T \subseteq A$. In such games, coalition structure generation is trivial because the agents are better off 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 better 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 would use if 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 (Sandholm and Lesser 1997). 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. $\{\{1\}, \{2\}, \ldots, \{a\}\}$ 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.

2.3. *Approximate coalition structure generation*

The coalition structure search process can be viewed as search in a *coalition structure graph*, see figure 1.

Now, how should such a graph be searched if there is not enough time to search it entirely? We would like to search through a subset $N \subseteq M$ of coalition structures,

pick the best coalition structure we have seen:

$$CS_N^* = \underset{CS \in N}{\operatorname{argmax}}\, V(CS) \tag{4}$$

and be guaranteed that this coalition structure is within a bound from optimal, i.e. that

$$k = \min\{\kappa\} \text{ where } \kappa \geqslant \frac{V(CS^*)}{V(CS_N^*)} \tag{5}$$

is finite, and as small as possible. We define n_{min} to be the smallest size of N that allows us to establish a finite bound k.

Sandholm *et al.* (1999a) showed that the minimal number of nodes that must be searched before a bound, k, can be established is $n_{min} = 2^{a-1}$. This is established by searching the lowest two levels in the coalition structure graph. After searching these two levels, all possible coalition values have been seen and this is the fastest way to see all of them. At that point, the bound, in the worst case, is $k = a$ which occurs when single agents have value 1, and all other coalitions have value 0, i.e. $v_S = 1$ if $|S| = 1$, and $v_S = 0$ otherwise.

3. Search algorithms

In this paper we investigate three search algorithms for the coalition structure search problem. The three algorithms are:

- **Merging Algorithm (MERGE)** does a breadth-first search from the top of the graph. In the worst case, 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

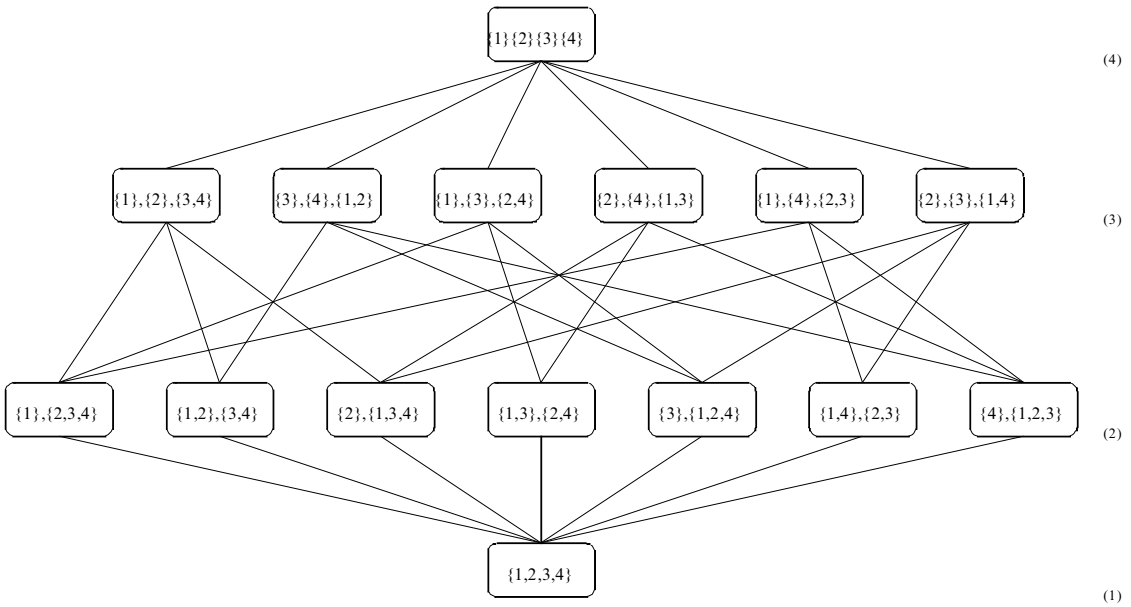

Figure 1. Coalition structure graph for a 4 agent game. The nodes represent coalition structures. The arcs represent mergers of two coalitions when followed downward, and splits of a coalition into two when followed upward.

necessarily help since at least part of level 2 needs to be searched as well: coalitions of size $a-1$ only occur there.

- **Splitting Algorithm (SPLIT)** does a breadth-first search from the bottom of the graph. After searching 2^{a-1} nodes, the bottom two levels of the graph, a bound $k = a$ is established. This is the optimal way of establishing a worst case bound. After that, the splitting algorithm reduces the bound slowly. This can be shown by constructing bad cases for the splitting algorithm. To construct a bad case, set $v_S = 1$ if $|S| = 1$, and $v_S = 0$ otherwise. Now, $CS^* = \{\{1\}, \ldots, \{a\}\}$, $V(CS^*) = a$, 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 found on level l 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 $\frac{V(CS^*)}{V(CS^*_N)} = 1$.) In other words the divisor increases by one every time a level is searched.
- **Coalition–Structure–Search–1 (CSS1)** searches the bottom two levels of the coalition structure graph and then begins a breadth-first search from the top of the graph which continues until time runs out. It returns the best coalition structure among those seen so far. After searching the bottom two levels, the bound is (in the worst case) $k = a$. After 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. The exact drop in the bound is formulated in Sandholm *et al.* (1999a). After searching the bottom two levels of the graph, **CSS1** decreases the bound at a faster rate than **SPLIT**. The levels that **CSS1** searches after the bottom two levels have fewer nodes than the levels that **SPLIT** searches.

The three were selected for study for two reasons. First, in an earlier paper (Sandholm *et al.* 1999a) we had derived worst case theoretical results for the algorithms. We believed that the bounds obtained theoretically were substantially higher than what we would find in the average case. We were also curious as to whether certain behaviour seen in the worst case analysis would appear experimentally. This behaviour included the importance of completing search through an entire level in the coalition structure graph in order to decrease the bound. Second, intuitively these three search algorithms seemed the most reasonable ones for the problem. They were anytime algorithms (algorithms with the property that their output quality improves over time), that could be used in settings where only the coalition structure values could be observed, or in settings where there was more information available, such as the values of individual coalitions.

4. Setup of the experiments

The set-up of the simulations is as follows. For problem instances of six to ten agents, a coalition structure graph was generated and values were assigned to each coalition (and thus to each coalition structure). The values were chosen using four alternative instance distributions:

1. Each coalition's value was picked independently from a uniform distribution between 0 and 1.

2. Each coalition's value was picked independently from a uniform distribution between 0 and the size of the coalition, $|S|$.
3. The coalition values were superadditive, i.e. $v_{S\cup T} \geqslant v_S + v_T$. The method of choosing the values is irrelevant. We selected one method of assigning values, but any other method that lead to the above inequality holding would result in the same behaviour for each algorithm.
4. The coalition values were subadditive, i.e. $v_{S\cup T} < v_S + v_T$. The method of choosing the values is irrelevant.

The search algorithms were only able to see the values of the coalition structures, not the values of the individual coalitions. In the experiments each graph was searched exhaustively in order to find the optimal coalition structure which was used as a baseline against which the search algorithms were compared. The search was then restarted using one of the algorithms: **MERGE**, **SPLIT**, or **CSS1**. After each node was searched, the bound, k, was calculated. This procedure was repeated one thousand times for each distribution and algorithm, with new values being assigned to the coalitions for each run. Once the one thousand runs were completed, the mean for the bound at each node was computed, along with 95% confidence intervals. Since the confidence intervals were so small, they are not included in any of the figures in this paper. The three algorithms were all executed on the same problem instances.

5. Results

In this section we discuss the results and the algorithms' behaviour observed during the simulations.

5.1. *Number of agents*

The number of agents did not qualitatively affect the overall behaviour of the algorithms. All algorithms behaved consistently. The experimental bound depended on the section of the graph that had been searched, not on the absolute number of nodes. It was more significant that search along a level had been completed as opposed to searching a specific number of nodes. This means that the three algorithms scaled up well.

5.2. *Average results as compared to Worst Case Analysis*

All three algorithms produced results that were substantially better than the worst case theoretical results. **MERGE** had very low bounds, within $k \leqslant a/5$, after searching the same number of nodes as found in the bottom two levels of the coalition structure graph. This was surprising since, theoretically, **MERGE** could not guarantee that any node it returned was within a finite bound from optimal if the entire coalition structure graph had not been searched since the nodes that **MERGE** searches last contain coalitions that are only found once in the entire graph. For both **SPLIT** and **CSS1**, after searching the bottom two levels the bound was well under $k = a$. In fact, for all a, the average bound, k, taken over all 1000 problem instances was less than or equal to $\frac{a}{3}$ after the bottom two levels had been searched.

Even though the bound obtained by **MERGE** was better than **SPLIT** and **CSS1** after searching a small subset of nodes, it did not mean that **MERGE** always

dominated the other two algorithms. Each algorithm outperformed the other two for some problem instance distribution and some amount of search. All three algorithms rapidly decreased the bound early on, with diminishing returns as the search progressed. This is a very desirable feature in an anytime algorithm.

5.3. *Algorithm performance under different problem instance distributions*

5.3.1. *MERGE.* When the coalition values were taken from the interval $[0, 1]$, **MERGE** did not cause any large reductions in the bound since the bound first calculated was already close to 1.00. However, the bound decreased the most during the beginning of the search and the reduction slowed as further search was performed, see figure 2.

In the case where the coalition values were weighted by the number of members in the coalition, nonconvexities were observed as the bound dropped, see figure 2. The nonconvexities corresponded to completion of search through a level in the coalition structure graph.

In subadditive domains, **MERGE** found the optimal coalition structure immediately, but in superadditive domains the algorithm had to search the entire graph before locating the optimal coalition structure.

5.3.2. *SPLIT.* When the coalition values were drawn uniformly from $[0, 1]$, the **SPLIT** algorithm reduced the bound early in the search, but the reduction slowed as more nodes were seen, see figure 3. After completing search of a level in the coalition structure graph, the bound dropped sharply, creating nonconvexities in the graph. These nonconvexities occurred exactly when a level had been exhaustively searched and the first node of a new level was observed.

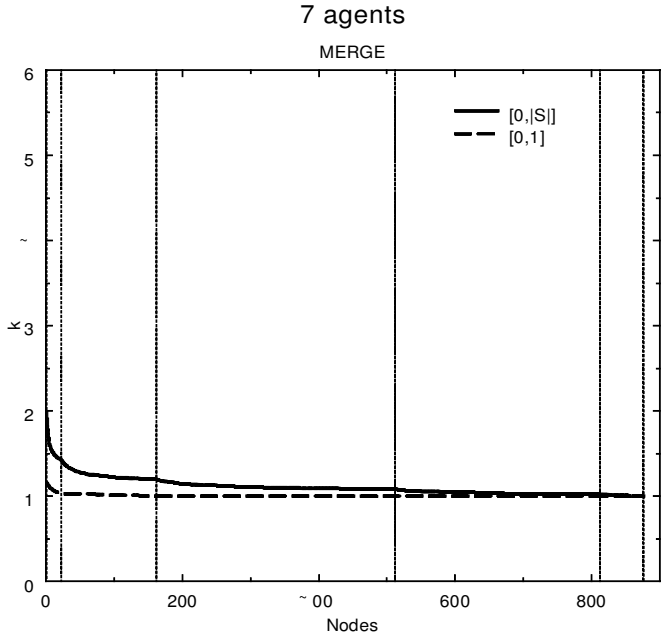

Figure 2. 7 agent setting where the **MERGE** algorithm was used to find the optimal coalition structure. The bounds from the two different coalition value distributions are shown. The vertical lines show where the algorithm has completed searching a level in the graph.

When coalition values were drawn uniformly from the interval $[0, |S|]$, **SPLIT** almost immediately reduced the bound to 1.00. No nonconvexities were observed during the bound reduction.

In a superadditive domain, **SPLIT** found the optimal coalition structure immediately. In a subadditive domain it had to search the entire graph before it located the optimal coalition structure.

5.3.3. *CSS1.* When the coalition values were drawn uniformly from $[0, 1]$, **CSS1** decreased the bound rapidly after seeing the first few nodes, but the decrease slowed as more nodes on the bottom two levels were searched. Once the two bottom levels were completed there was a sharp drop in the bound as the top node in the coalition structure graph was observed. As search continued the bound rapidly approached $k = 1.00$, see figure 4. On the other hand, under the uniform distribution on $[0, |S|]$ there were no nonconvexities observed in the graph, see figure 4. The bound decreased rapidly with only a few nodes searched. After searching the second level from the bottom, the results were already very close to optimal. For example, the bound for seven agents was $k = 1.0151$. The reduction of the bound slowed as more nodes were searched until, eventually, the optimal coalition structure was found.

In a superadditive domain, **CSS1** found the optimal coalition structure immediately, while in a subadditive domain it had to search 2^a nodes (i.e. the bottom two levels and the top node) before the optimal solution was found.

5.4. *Selective superiority of algorithms*

The three algorithms performed differently in different settings. In subadditive domains **MERGE** was the best algorithm since it found the optimal coalition structure immediately while **SPLIT** had to search the entire graph. **CSS1**, while not finding the optimal coalition structure immediately, did find it after searching 2^a

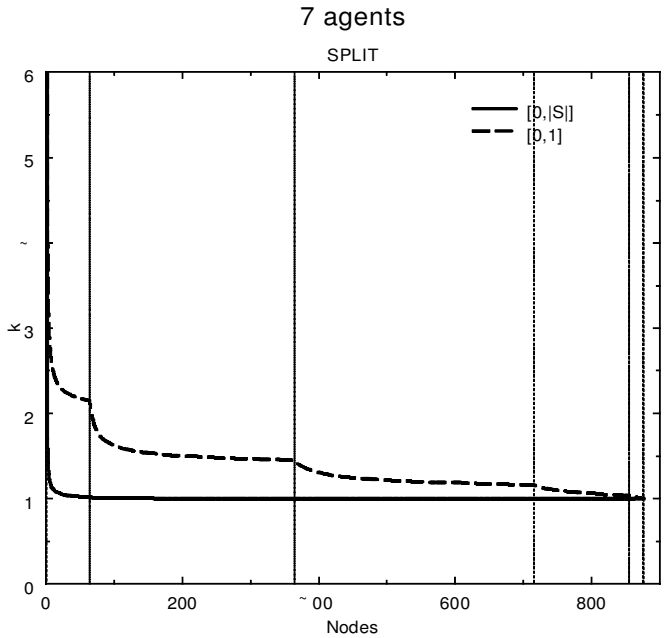

Figure 3. 7 agent setting where the **SPLIT** algorithm was used to find the optimal coalition structure. The bounds from the two different coalition value distributions are shown.

nodes which is linear in the size of the input, see figure 5. For superadditive domains, the outcome changed. Both **SPLIT** and **CSS1** found the optimal coalition structure immediately while **MERGE** had to search the entire search space before finding it, see figure 6.

In the setting where the coalition values were selected uniformly from the interval $[0, 1]$, **MERGE** outperformed **CSS1** and **SPLIT**, see figure 7. However, when the

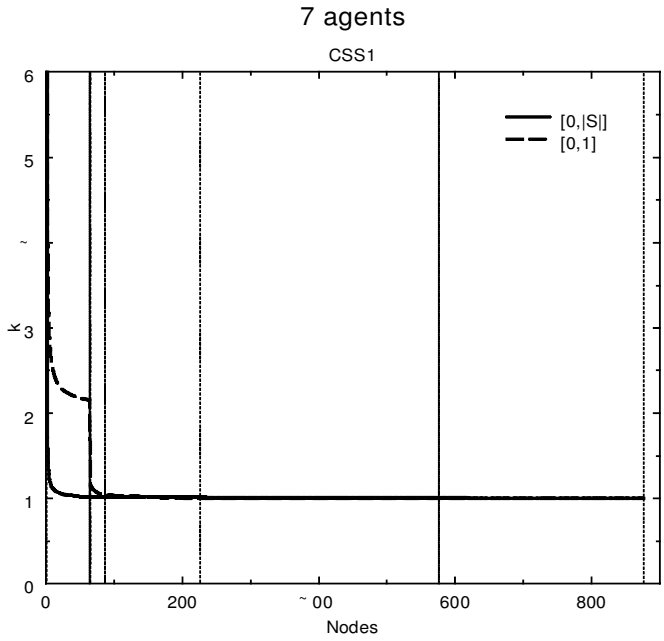

Figure 4. 7 agent setting where **CSS1** was used to find the optimal coalition structure. The bounds from the two different coalition value distributions are shown.

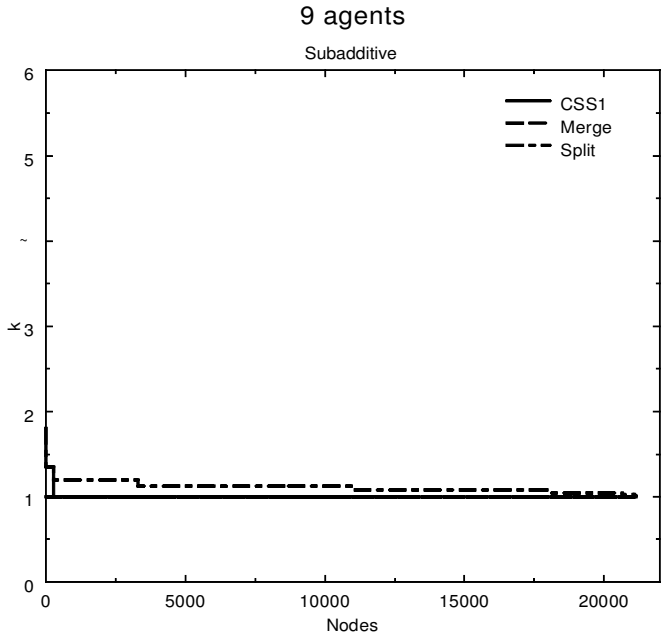

Figure 5. 9 agent setting where coalition values were subadditive. The first node that **MERGE** searches is the optimal coalition structure.

coalition values were weighted by the size of the coalition, **SPLIT** was the best algorithm with **CSS1** behaving in a similar fashion. **MERGE** did not do nearly as well as the other two, see figure 8. Table 1 summarizes the results.

6. Explaining the observed results

The question of interest to us is why do the coalition structure search algorithms behave differently in the various settings. A possible answer lies in the distributions

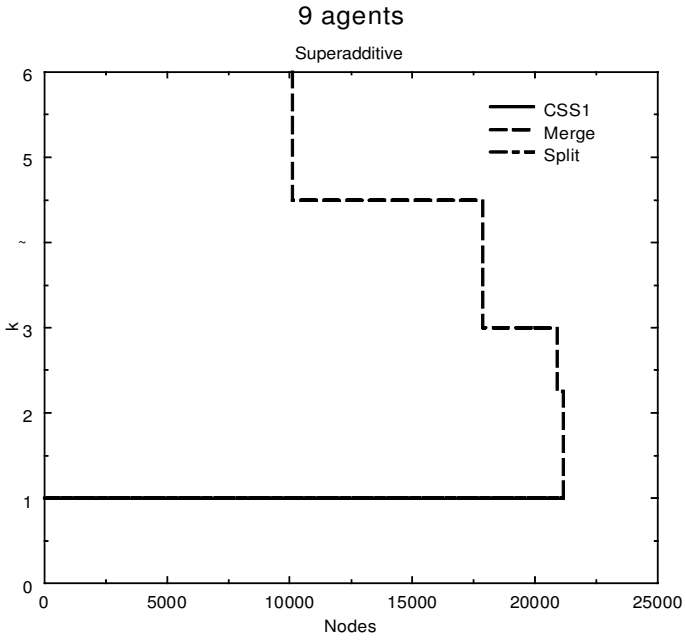

Figure 6. 9 agent setting where coalition values were superadditive. **SPLIT** led to the same graph as **CSS1**.

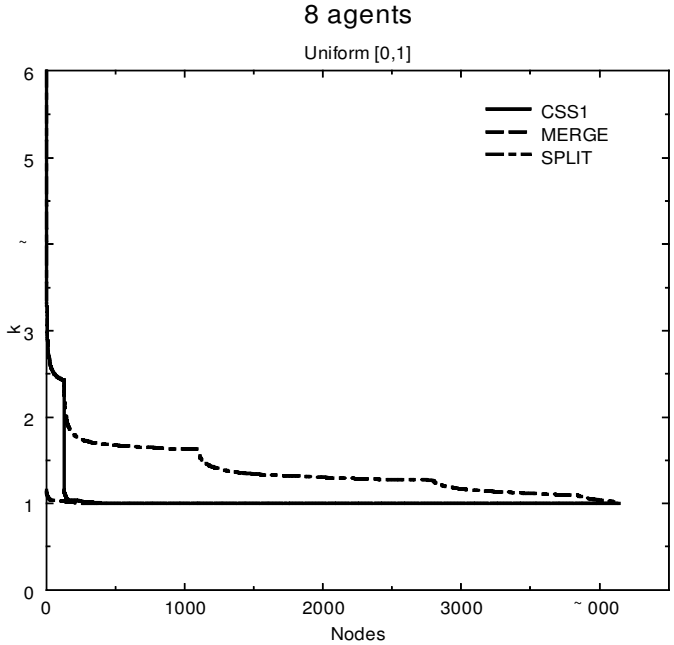

Figure 7. 8 agent setting with coalition values chosen uniformly from [0,1].

from where the coalition values (and hence coalition structure values) are drawn. Recall that the value of a coalition structure is

$$V(CS) = \sum v_S$$

where $S \in CS$. The expected value of the coalition structure, CS, is

$$E[V(CS)] = \sum E[v_S]$$

and the variance is

$$\mathrm{Var}[V(CS)] = \sum \mathrm{Var}[v_S]$$

since we pick the coalition values, v_S, independently. All values, v_S, were drawn uniformly from an interval $[0, b]$ where $b = 1$ or $b = |S|$, except for the superadditive and subadditive cases. Therefore

$$E[v_S] = \frac{b}{2}$$

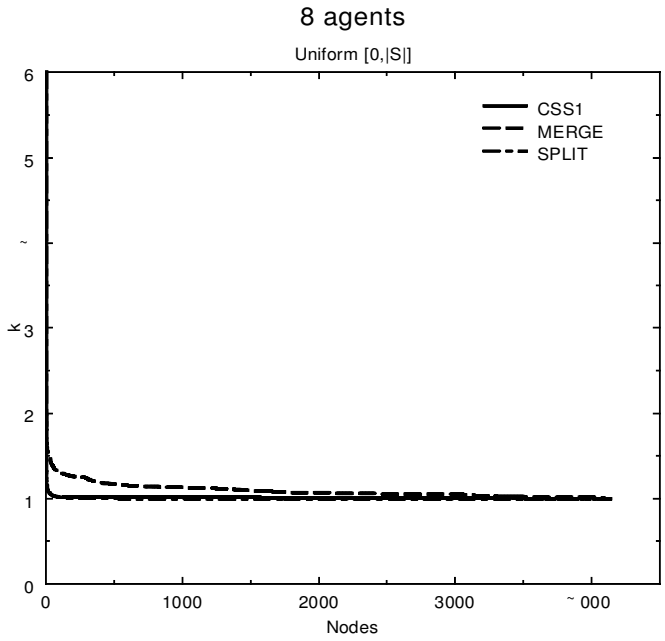

Figure 8. 8 agent setting with coalition values chosen uniformly from $[0, |S|]$.

Table 1. A comparison of the three different algorithms in different settings. The algorithms are ranked: 1 indicates that the algorithm was the best in the setting while a 3 means that it performed the worst compared with the others.

	*CSS*1	*MERGE*	*SPLIT*
Superadditive	1	2	1
Subadditive	2	1	3
$[0, 1]$	2	1	3
$0, |S|]$	2	3	1

and

$$\mathrm{Var}[v_S] = \frac{(b-0)^2}{12} = \frac{b^2}{12}.$$

6.1. *Uniform from Interval* $[0, 1]$

When the coalition values are drawn uniformly from the interval $[0, 1]$, the expected value of a coalition structure depends on which level of the coalition structure graph it is found. For example, the expected value of the node on the bottom level (the grand coalition) is $1/2$, while the expected value of the node at the top of the graph is $a/2$. In general, if a node is found at level l in the coalition structure graph, then its expected value is $l/2$. As one searches lower levels of the graph the expected value of the nodes decreases. The variance of a node in the graph also depends on the level on which the node is found. The bottom node in the graph has the least variance ($1/12$), while the top node has highest variance ($a/12$). In general, a node at level l will have variance $l/12$. As one searches lower levels of the graph, the variance per node decreases. Within a level, however, both expected value and variance are constant.

In figure 7 **SPLIT** performed poorly compared to the other two algorithms. This is because it conducts a breadth first search, starting at the bottom of the graph. It does not search the nodes with highest expected value until the end. One also observes that the plot for **SPLIT** contains long plateaus where the bound barely decreases, followed by a sharp drop (nonconvexity) in the bound. These nonconvexities occur whenever **SPLIT** begins to search a new level. The expected value of the nodes searched on the new level is higher than that on the previous level (i.e. the new node has expected value $l/2$ while the previous node had expected value $(l-1)/2$) which causes the sharp drop in the bound. There is relatively little improvement in the bound when searching within a level since the expected value is the same for all nodes.

CSS1 also caused the bound to sharply drop once the top node in the coalition structure graph had been observed. This happens since it had previously been searching on level 2 where the expected value of a node is 1, and then began searching level a where the expected value is $a/2$. After searching the top node the bound was close to $k = 1.00$ and further reductions were negligible (see figure 7).

MERGE outperformed the other algorithms in this setting. It searched the nodes with the highest expected value first and so was likely to find either the optimal coalition structure or coalition structures with values close to optimal quickly. After searching only a few nodes, the bound was so close to $k = 1.00$ that any further decrease was hard to observe (see figure 7).

6.2. *Uniform from Interval* $[0, |S|]$

When the coalition values were weighted by the size of the coalition (i.e. v_S was drawn from $[0, |S|]$) the expected value of every node in the coalition structure graph was $a/2$. The variance differed between nodes, even nodes that were found on the same level of the graph. For example, in the 7 agent case, the coalition structure $\{\{1\}, \{2\}, \{3\}, \{4\}, \{5,6,7\}\}$ had variance $\frac{1}{12} + \frac{1}{12} + \frac{1}{12} + \frac{1}{12} + \frac{9}{12} = \frac{13}{12}$, while the coalition structure $\{\{1\}, \{2\}, \{3,4\}, \{5,6,7\}\}$ had variance $\frac{1}{12} + \frac{1}{12} + \frac{4}{12} + \frac{9}{12} = \frac{15}{12}$.

These two coalition structures are found on level 4 of the coalition structure graph for 7 agents. However, as the level increases, the variance of a given node from the level decreases. The node with highest variance on level l had variance that was lower than the node with highest variance on level $l-1$. In fact, in some cases the node with highest variance on level l had variance that was lower (or equal) to the node with lowest variance in level $l-1$. Out of any single coalition structure searched, the bottom node in the coalition structure graph was most likely to have value close to (or equal to) that of the optimal coalition structure since the variance was large compared to all other nodes in the graph.

The algorithms **SPLIT** and **CSS1** did well in this setting. Both algorithms searched the bottom two levels of the graph first. While the mean is the same for each node, the variance, or dispersion, of the values is the highest in the bottom two levels. Therefore, this is where, out of any single coalition structure searched, it is most likely to find the optimal coalition structure, or at least coalition structures that have values close to optimal. **SPLIT** did better than **CSS1** since it continued searching the bottom sections of the graph while **CSS1** moved to the top of the graph.

MERGE did not perform as well as the other algorithms. It searched the top part of the graph first and only reached the area where higher valued coalition structures were most likely found at the end of its search. There were small decreases in the bound as **MERGE** began to search each new level. These can be explained by the difference in variance between levels. Coalition structures with greater value were more likely to be found on each new level.

6.3. *Controlling for the mean and variance*

While our mean and variance based explanations of the algorithms' behaviour are intuitively appealing, they do not completely capture the situation. They are exactly correct only when searching one node. When the search sees multiple nodes, a more detailed analysis would take into account the fact that the coalition structure values are not independent because the coalition structures share coalitions. It is conceivable that the simple model does not actually account for the observed behaviour. Therefore, we performed a controlled run. We assigned values to the coalitions so that the mean and variance was the same for each node. This was done by setting the value of coalition S to be the sum of $|S|$ values picked independently from a uniform distribution between 0 and 1. Thus, the value of each coalition structure was the sum of a values

$$V(CS) = \sum_{a} x_i$$

where each x_i was picked independently from a uniform distribution on $[0,1]$. This meant that

$$E[V(CS)] = \frac{a}{2} \quad \text{and} \quad \text{Var}[V(CS)] = \frac{a}{12}$$

for each coalition structure.

We then ran all three algorithms and plotted the results, see figure 9. All three algorithms produced almost identical results with no one algorithm outperforming the others. This suggests that the mean and variance indeed explain the observed behaviour: other factors are insignificant.

7. Related research

Coalition formation has been widely studied in game theory (Bernheim *et al.* 1987, Kahan and Rapoport 1984, Aumann 1959). They mainly address the question of how to divide $V(CS^*)$ among agents so as to achieve stability of the payoff configuration. However, most of that work has not taken into account the computational limitations involved. This section reviews some of the work that has been done on the computational aspects.

Early work in the field includes Friend's program (Kahan and Rapoport 1984, Friend 1973) that simulates a 3-agent coalition formation situation where agents can make offers, acceptances, and rejections to each other regarding coalitions and payoffs. 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 myopically consider only current proposals: they are assumed to do no forward thinking 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.

Deb *et al.* (1996) bound the maximal number of payoff configurations (i.e. coalition structure–payoff division pairs) that must be searched to guarantee stability of the payoff configuration. Unlike our work, they neither address a bound on solution quality nor provide methods for coalition structure generation.

Transfer schemes represent a dynamic approach to the payoff division activity of coalition formation in CFGs (Kahan and Rapoport 1984, Stearns 1968). 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 local strategy: by using some other strategy,

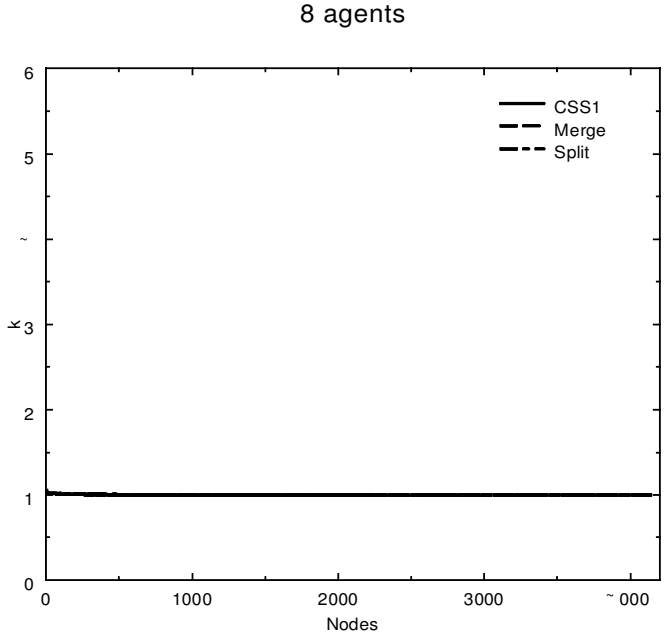

Figure 9. 8 agent setting where the expected value and variance is the same for all nodes.

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 (Wu 1977). 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 non-empty, i.e. if such a stable payoff vector exist.

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.

Zlotkin and Rosenschein (1994) analyse 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 (Shapley 1953). 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 (Zlotkin and Rosenschein 1994).

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 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 algorithms 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 could be randomized. This would give each agent an expected payoff equal to its Shapley value.

Ketchpel (1994) 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.

Shehory and Kraus (1996) analyse 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.

Shehory and Kraus (1995) also present an algorithm for coalition structure generation among agents that are not self–interested, but cooperative, i.e. social welfare maximizing. 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 (Shehory and Kraus 1998). 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.

The 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.

Coalitional bargaining addresses both coalition formation and payoff distribution. Coalitional bargaining is seen as a generalization of the Rubinstein alternating offer bargaining model (Rubinstein 1982). A typical model has agents 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 members. Unanimous agreement among the members of the proposed coalition lead to it being formed, otherwise no coalition is formed and another proposal is made. Some of the work in this area is reviewed here. To the best of our knowledge most of the work in this area has not taken into account computational limitations.

Chatterjee *et al.* (1993) investigate efficiency properties of stationary equilibria of strictly superadditive games where the first agent to reject a proposal becomes the next proposer. 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 equilibrium since it grants different agents different amounts of power.

Okada (1996) studies a similar setting, but is interested how efficiency of agreement is affected by the bargaining procedure, in particular by the rule governing the selection of proposers. Unlike Chatterjee *et al.* (1993), the proposer is chosen randomly with equal probability from among the group of remaining agents. In this

setting, 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 (1997) investigates a setting where agents compete for the right to make a proposal by investing resources. The agent that invests the most in a round wins the right to make the proposal for that round. 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.

Sandholm and Lesser (1997) 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 algorithms 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 that coalition would strike the optimal trade-off between quality of the optimization solution and the cost of that computation. Any one of our three algorithms 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 redecide its membership if its original coalition receives a worse optimization solution than expected.

8. Conclusions and future research

Coalition formation is a key topic in multiagent systems. One would 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.

This paper presented results from an empirical study where three coalition structure search algorithms, **CSS1**, **MERGE**, and **SPLIT**, were tested. We showed that there was no 'best' algorithm among the three. In superadditive domains **SPLIT** and **CSS1** performed best while in subadditive domains **MERGE** was the best. When coalition values were drawn uniformly from the interval $[0, 1]$, **MERGE** outperformed the others because it searched the nodes that had the highest expected value first. On the other hand, when the coalition values were weighted by the size of the coalition (i.e. drawn from the interval $[0, |S|]$) **SPLIT** was the best and **MERGE** performed poorly. This was because **SPLIT** searched nodes with highest variance first.

CSS1, while never being the best algorithm in any of the domains studied, has the advantage that after searching $2^a - 1$ nodes, the best coalition structure seen has value that is within a worst case bound $k = a$ from optimal. **CSS1** also took advantage of both distributions by searching early areas of the coalition structure graph where the optimal coalition structure was likely to be found. This led to fairly consistent performance among the value distributions studied: **CSS1** was always close to the best algorithm.

The long term goal for future research is to construct normative methods that reduce the complexity for all three activities of coalition formation simultaneously: coalition structure generation, optimization and payoff division. Since the activities are interdependent, a complete theory of coalition formation much address all three. Future research includes studying the possibility of directing the search process as we observe values of coalition structures (online-search control policies), or of being able to design algorithms that return the best possible solution given a specified amount of time (design-to-time algorithms) for coalition structure generation. We are also interested in studying cases where the coalition values are known, instead of only knowing the coalition structure values. In these situations, coalition structure generation becomes very similar to winner determination in combinatorial auctions (Sandholm 1999, Rothkopf *et al.* 1998). We are interested in seeing how our results apply to this new domain, and whether we can use results from combinatorial auctions for coalition structure generation. We are also analysing the interplay of dynamic coalition formation and belief revision among bounded rational agents (Tohme and Sandholm 1999). 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 nonmanipulable leveled commitment contracts (Sandholm and Zhou 1999, Sandholm *et al.* 1999b, Sandholm 1996, Sandholm and Lesser 1996) to coalition formation deals as one possible way of implementing backtracking in this setting. We are also interested in the effect of computation on the process of coalitional bargaining.

Acknowledgements

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

References

Aumann, R., 1959, Acceptable points in general cooperative n-person games. In Volume IV of *Contributions to the Theory of Games* (Princeton University Press).

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

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

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

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

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

Kahan, J. and Rapoport, A., 1984, *Theories of Coalition Formation* (Lawrence Erlbaum Associates Publishers).

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

Larson, K. and Sandholm, T., 1999, Anytime coalition structure generation: An average case study. In *Proceedings of the Third International Conference on Autonomous Agents*, Seattle, WA, pp. 40–47.

Lundgren, M. G., Jörnsten, K. and Värbrand, P., 1992, On the nucleolus of the basic vehicle routing game. Technical Report 1992–96, Linköping University, Department of Mathematics, Sweden.

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

Raiffa, H., 1982, *The Art and Science of Negotiation* (Cambridge MA: Harvard University Press).

Rosenschein, J. and Zlotkin, G., 1994, *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers* (MIT Press).

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

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

Sandholm, T. W., 1996, Negotiation among self-interested computationally limited agents. PhD thesis, University of Massachusetts, Amherst.

Sandholm, T. W., 1999, An algorithm for optimal winner determinism in combinatorial auctions, In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, pp. 542–547. Extended version: Washington University, Department of Computer Science technical report WUCS-99-01.

Sandholm , T. W., and Lesser, V. R., 1996, Advantages of a leveled commitment contracting protocol. In *Proceedings of the National Conference on Artificial Intelligence* (*AAAI*), Portland, OR., pp. 126–133.

Sandholm. T. W., and Lesser, V. R., 1997, Coalitions among computationally bounded agents. *Artificial Intelligence*, **94**(1): 99–137.

Sandholm, T. W., Larson, K. S., Andersson, M. R., Shehory, O., and Tohmé, F., 1999a, Anytime coalition structure generation with worst case guarantees. *Artificial Intelligence*, **111**: 209–238.

Sandholm T. W., Sikka, S., and Norden, S., 1999b, Algorithms for optimizing leveled commitment contracts, In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence* (*IJCAI*), Stockholm, Sweden, pp. 535–540.

Sandholm, T., and Zhou, Y., 1999, Revenue equivalence of leveled commitment contracts. Technical report WUCS-99-03, Washington University, Department of Computer Science.

Shapley, L. S., 1953, A value for n-person games. In H. W. Kuhn and A. W. Tucker (eds) *Contributions to the Theory of Games*, volume 2 of *Annals of Mathematical Studies, 28* (Princeton University Press), pp. 307–317.

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

Shehory O. and Kraus, S., 1996, A kernel-oriented model for coalition formation in general environments: Implementation and Results. In *Proceedings of the National Conference on Artificial Intelligence* (*AAAI*), Portland, OR, pp. 134–140.

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

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

Tohmé F. and Sandholm, T., 1999, Coalition formation processes with belief revision among bounded rational self-interested agents. *Journal of Logic and Computation*, **9**: 1–23.

van der Linden, W. and Verbeek, A., 1985, Coalition formation: A game theoretic approach. In H. A. M. Wilke (ed), *Coalition Formation*, volume 24 of *Advances in Psychology*, (North Holland).

Wu, L. S., 1977, A dynamic theory for the class of games with non-empty cores. *SIAM Journal of Applied Mathematics*, **32**: 328–338.

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