

Anytime Coalition Structure Generation: An Average Case Study

Kate S. Larson and Tuomas W. Sandholm*

{ksl2, sandholm}@cs.wustl.edu

Department of Computer Science, Campus Box 1045
Washington University, One Brookings Drive
St. Louis, MO 63130-4899

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 theoretical results obtained in [13]. 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.

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, 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. 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 sav-

ings 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 limited time. Instead agents have to select a subset of coalition structures on which to focus their search. Sandholm et al [13] 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 presented 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 paper is organized as follows. Section 2 discusses coalition structure in characteristic function games. The following 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 discusses 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 [6, 17, 9, 22, 20, 19, 23, 7, 15], 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 [16].)

We assume that each coalition's value is nonnegative:

$$v_S \geq 0 \quad (1)$$

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 [6].

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

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Autonomous Agents '99 Seattle WA USA

Copyright ACM 1999 1-58113-066-x/99/05...\$5.00

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 the sum of the values of the coalitions in it:

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

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

$$CS^* = \arg \max_{CS \in M} V(CS) \quad (3)$$

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 [13].

The coalition structure search process can be viewed as search in a *coalition structure graph*, Figure 1. Finding the optimal coalition structure in this graph is infeasible due to computational complexity.

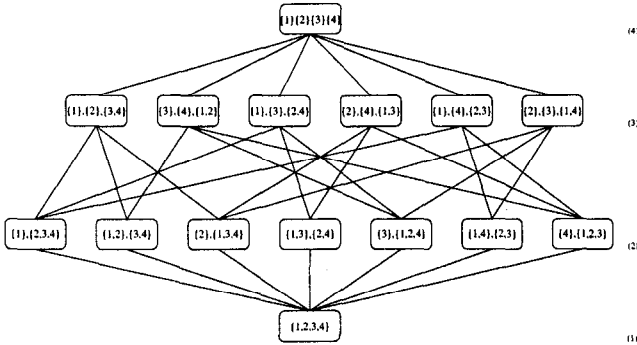


Figure 1: *Coalition structure graph for a 4 agent game. The nodes represent coalition structures. The arcs represent mergers of two coalitions when followed downward, and splits of a coalition into two when followed upward.*

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:

$$CS_N^* = \arg \max_{CS \in N} V(CS) \quad (4)$$

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

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

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

Sandholm et al [13] showed that the minimal number of nodes that must be searched before a bound, k , can be established is $n = 2^{a-1}$. 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 this point, the bound, in the worst case, is $k = a$ which occurs when all coalitions, S , with $|S| = 1$ have value $v_S = 1$ and all other coalitions have value 0.

3 Search Algorithms

In this paper we investigate three search algorithms for the coalition structure search problem. The three were selected for study for two reasons. First, we had worst case theoretical results for them. Second, they seemed like reasonable search algorithms for the problem.

- **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 [13].
- **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 necessarily help since at least part of level 2 needs to be searched as well: coalitions of size $a - 2$ only occur there.
- **Splitting Algorithm (SPLIT)** does a breadth first search from the bottom of the graph. This is identical to CSS1 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 CSS1. 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, $CS^* = \{\{1\}, \dots, \{a\}\}$, $V(CS^*) = a$, 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, $\frac{V(CS^*)}{V(CS_N^*)} = \frac{a}{l-1}$.¹ 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 CSS1 searches first.

¹The only exception comes when the algorithm completes the last (top) level, i.e. $l = a$. Then $\frac{V(CS^*)}{V(CS_N^*)} = 1$.

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 assigned to each coalition (and thus each coalition structure). The values were chosen using four different 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 $|S|$, where $|S|$ was the size of the coalition.
3. The coalition values were superadditive, i.e. $v_{S \cup T} \geq v_S + v_T$. The method of choosing the values is irrelevant.
4. The coalition values were subadditive, i.e. $v_{S \cup T} \leq v_S + v_T$. 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 the graph was searched exhaustively in order to find the optimal coalition structure. The search was then restarted using one of the algorithms: **CSS1**, **MERGE**, or **SPLIT**. 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 at each run. Once the one thousand runs were completed, the mean for the bound at each node was computed, along with 95 percent 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' behavior observed in the simulations.

5.1 Number of Agents

The number of agents did not qualitatively affect the overall behavior of the algorithms. All algorithms behaved consistently. The calculated bound was dependent on the section of the graph that had been searched, not on the absolute 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. For **CSS1**, after searching the bottom two levels and the top node, the bound was well under $k = a$ for a . In fact, for all a , $k \leq \frac{a}{3}$ after the bottom levels had been searched. **SPLIT**, which searched the bottom two levels first like **CSS1** had similar results. **MERGE** had very low bounds, within $k \leq a/5$, 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**

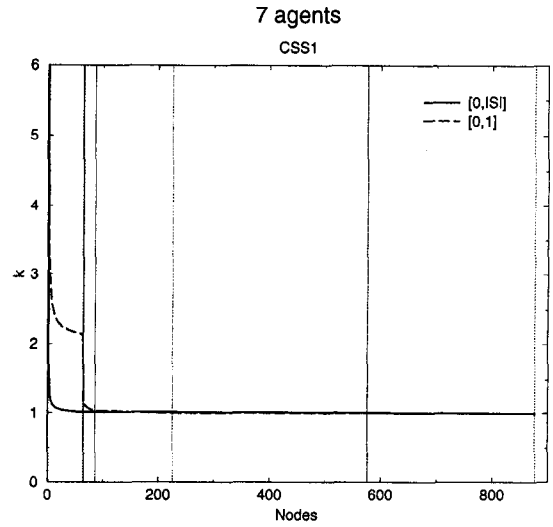


Figure 2: 7 agent setting where **CSS1** was used to find the optimal coalition structure. The bounds from the two different coalition value distributions are shown. The vertical lines show where the algorithm has completed searching a level in the graph

searches last contain coalitions that are only found once in the entire graph.

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 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

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 most of the bottom two levels had been 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 $k = 1.0$, Figure 2. On the other hand, under the uniform distribution on $[0, |S|]$ there were no nonconvexities observed in the graph, Figure 2. 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.

For **MERGE**, when the coalition values were from the interval $[0, 1]$, there were no large reductions in the bound since the bound first calculated was already low. However, the bound decreased the most during the beginning of the

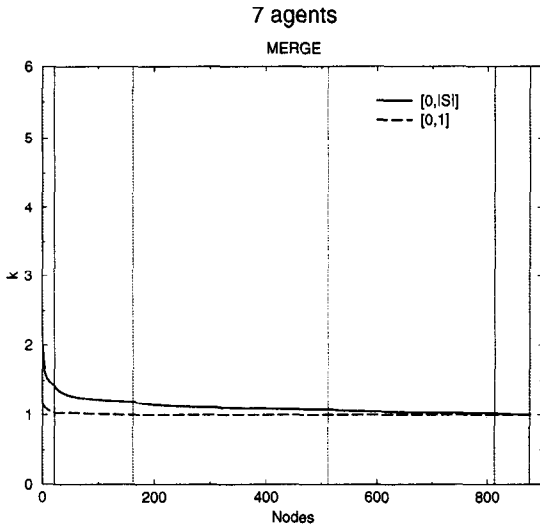


Figure 3: 7 agent setting where the **MERGE** algorithm was used to find the optimal coalition structure. The bounds from the two different coalition value distributions are shown.

search and the reduction slowed as further search was performed, Figure 3.

In the case where the coalition values were weighted by the number of members in the coalition, nonconvexities were observed as the bound dropped (Figure 3). 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.

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, Figure 4. 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.

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 but had to search the entire graph before it located the optimal coalition structure in a subadditive domain.

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 nodes which is linear in the size of the input, Figure 6. 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, Figure 5.

In the setting where the coalition values were selected uniformly from the interval $[0, 1]$, **MERGE** outperformed **CSS1** and **SPLIT** (Figure 7). However, when the coalition values were weighted by the size of the coalition,

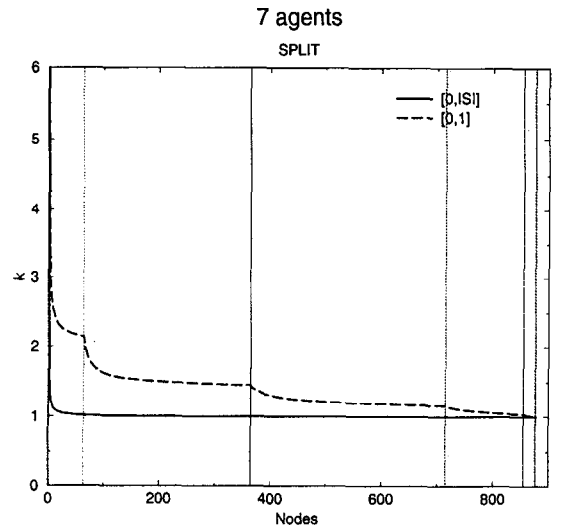


Figure 4: 7 agent setting where the **SPLIT** algorithm was used to find the optimal coalition structure. The bounds from the two different coalition value distributions are shown.

SPLIT was the best algorithm with **CSS1** behaving similarly. **MERGE** did not do nearly as well as the other two (Figure 8).

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

$$V(CS) = \sum v_S \quad (6)$$

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

$$E[V(CS)] = \sum E[v_S] \quad (7)$$

and the variance is

$$\text{Var}[V(CS)] = \sum \text{Var}[v_S] \quad (8)$$

since we assume that the v_S values are independent.

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

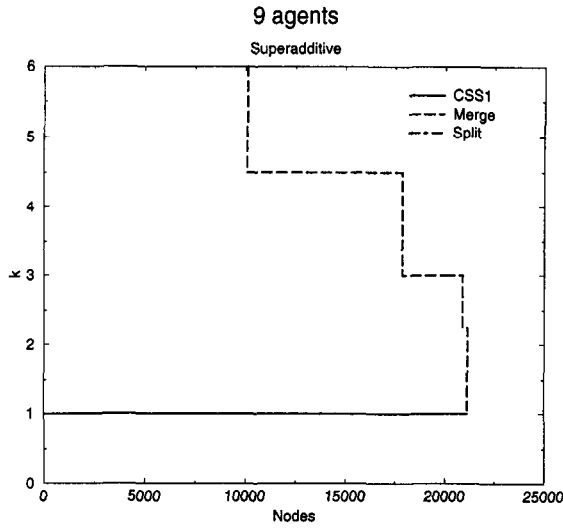


Figure 5: 9 agent setting where coalition values were superadditive. **SPLIT** led to the same graph as **CSS1**.

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 non-convexities occur whenever **SPLIT** began 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 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 begin searching level a where the expected value is $a/2$. After searching the top node the bound was close to $k = 1.00$ and it was difficult to observe any further reduction, 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 the optimal coalition structure quickly. After searching only a few nodes, the bound was so close to $k = 1.00$ that any further decrease was hard to 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 $19/12$, while the coalition structure $\{\{1\}, \{2\}, \{3,4\}, \{5,6,7\}\}$ had variance $15/12$. These two coalition

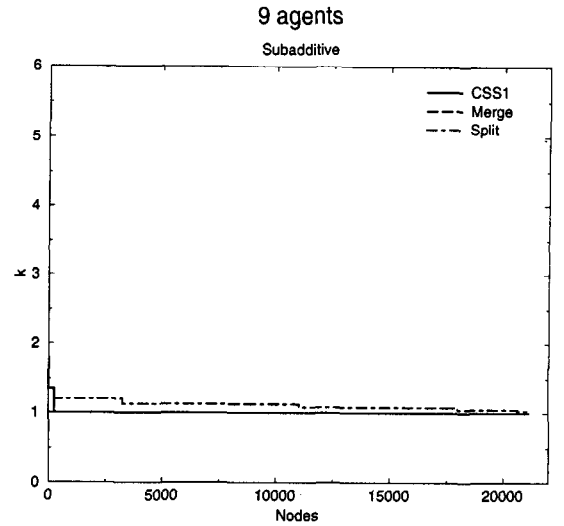


Figure 6: 9 agent setting where coalition values were subadditive.

structures are found on level 4 of the coalition structure graph for 7 agents. However, as the level increased, the variance of a given node from the level decreased. The node with highest variance on level l had variance that was lower (or equal) to the node with lowest variance on level $l-1$. Out of any single coalition structure searched, the bottom node in the coalition structure graph was most likely to be the optimal coalition structure.

Both **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. **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 the optimal coalition structure was 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. Nodes 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' behavior 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 behavior. 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

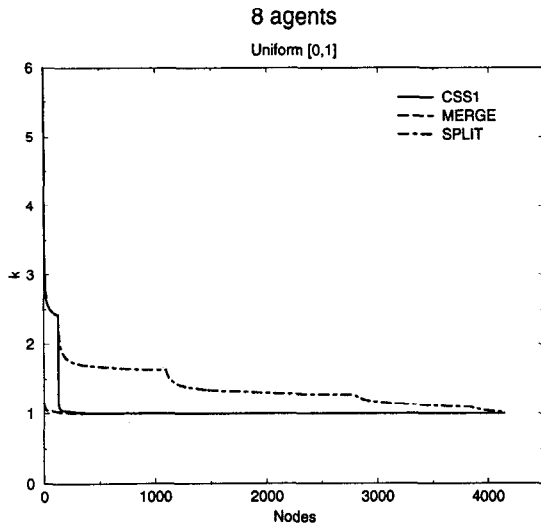


Figure 7: 8 agents with coalition values chosen uniformly from $[0, 1]$.

between 0 and 1. Thus, the value of each coalition structure was the sum of a values

$$V(CS) = \sum_a x_i \quad (9)$$

where each x_i was picked independently from a uniform distribution on $[0, 1]$. We then ran all three algorithms and plotted the results (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 behavior: other factors are insignificant.

7 Related Research

Coalition formation has been widely studied in game theory [6, 2, 1]. They mainly address the question of how to divide $V(CS^*)$ among agents so as to achieve stability of the payoff configuration. Coalitional bargaining addresses both coalition formation and payoff distribution [3, 8, 5]. Coalitional bargaining is seen as a generalization of the Rubinstein alternating offer bargaining model [10]. 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 should 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. There has been work on efficiency properties in this setting and how different protocols for determining the order in which agents make proposals affect the final outcome of the bargaining. However, to the best of our knowledge, most of the work has not taken into account the computational limitations involved. This section reviews some of the work that has.

Recently Deb et al [4] bounded the maximal number of payoff configurations that must be searched to guarantee stability. Unlike our work, they neither address a bound on solution quality nor provide methods for coalition structure generation.

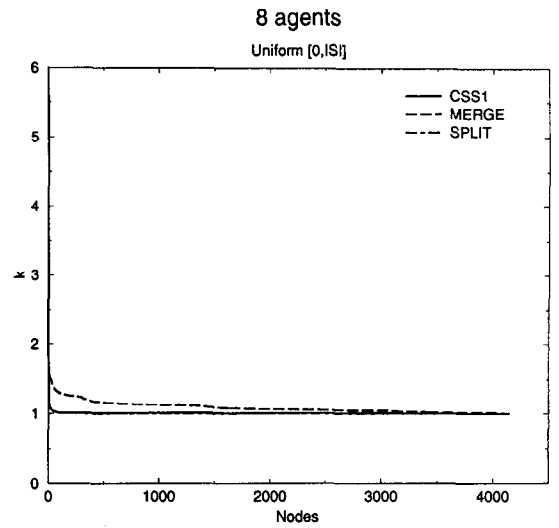


Figure 8: 8 agents with coalition values chosen uniformly from $[0, |S|]$.

Ketchpel [7] 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 guarantees neither a bound from optimum nor stability of the coalition structure. There is no mechanism for motivating self-interested agents to follow his algorithm.

Shehory and Kraus [19] 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 (K-stability) is met. Their other protocol guarantees a weaker form of stability (polynomial K-stability), but only requires searching a polynomial number of coalition structures. Their algorithm is an anytime algorithm, but does not guarantee a bound from optimum.

Shehory and Kraus [18] also present an algorithm for coalition structure generation among cooperative agents. 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. Also, they address a more specialized setting where the v_S values have special structure.

Sandholm and Lesser [16] 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 that coalition would strike the optimal

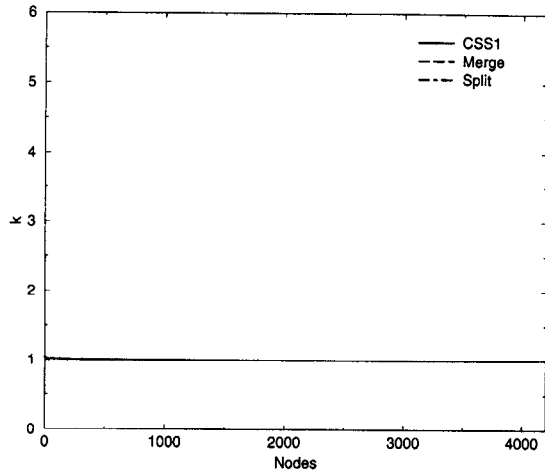


Figure 9: 8 agent setting where the expected value and variance is the same for all nodes.

tradeoff 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, had the advantage that after searching $2^a - 1$ nodes, the best coalition structure seen had value that was within a 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.

Future research includes studying design-to-time algo-

rithms and on-line search control policies 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 [12]. We are interested in seeing how our results apply to this new domain, and whether we can use results from combinatorial auction for coalition structure generation. We are also analyzing the interplay of dynamic coalition formation and belief revision among bounded rational agents [21]. 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 [14, 11] 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.

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] K. Chatterjee, B. Dutta, D. Ray, and K. Sengupta. A noncooperative theory of coalitional bargaining. *Review of Economic Studies*, 60:463–477, 1993.
- [4] 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.
- [5] R. Evans. Coalitional bargaining with competition to make offers. *Games and Economic Behavior*, 19:211–220, 1997.
- [6] J. P. Kahan and A. Rapoport. *Theories of Coalition Formation*. Lawrence Erlbaum Associates Publishers, 1984.
- [7] 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.
- [8] A. Okada. A noncooperative coalitional bargaining game with random proposers. *Games and Economic Behavior*, 16:97–108, 1996.
- [9] H. Raiffa. *The Art and Science of Negotiation*. Harvard Univ. Press, Cambridge, Mass., 1982.
- [10] A. Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica*, 50:97–109, 1982.
- [11] 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>.

- [12] T. W. Sandholm. An algorithm for optimal winner determination in combinatorial auctions. Technical Report WUCS-99-01, Washington University, Department of Computer Science, 1999.
- [13] 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.
- [14] 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.
- [15] 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.
- [16] 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 at IJCAI-95.
- [17] 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.
- [18] 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.
- [19] O. Shehory and S. Kraus. A kernel-oriented model for coalition-formation in general environments: Implementation and results. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 134–140, Portland, OR, Aug. 1996.
- [20] R. E. Stearns. Convergent transfer schemes for n -person games. *Transactions of the American Mathematical Society*, 134:449–459, 1968.
- [21] F. Tohmé and T. W. Sandholm. Coalition formation processes with belief revision among bounded rational self-interested agents. In *IJCAI Workshop on Social Interaction and Communityware*, pages 43–51, Nagoya, Japan, Aug. 1997.
- [22] L. S. Wu. A dynamic theory for the class of games with nonempty cores. *SIAM Journal of Applied Mathematics*, 32:328–338, 1977.
- [23] 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.