Forming Stable Coalitions in Large Systems with Self-Interested Agents

Pavel Janovsky and Scott A. DeLoach

Kansas State University, Department of Computer Science, KS 66506, USA,
{janovsky,sdeloach}@ksu.edu

Abstract. In coalition formation with self-interested agents both social welfare of the multi-agent system and stability of individual coalitions must be taken into account. However, in large-scale systems with thousands of agents, finding an optimal solution with respect to both metrics is infeasible.
In this paper we propose an approach for finding coalition structures with suboptimal social welfare and coalition stability in large-scale multi-agent systems. Our approach uses multi-agent simulation to model a dynamic coalition formation process. Agents increase coalition stability by deviating from unstable coalitions. Furthermore we present an approach for estimating coalition stability, which alleviates exponential complexity of coalition stability computation. This approach enables us to select a solution with high values of both social welfare and coalition stability.
We experimentally show that our approach causes a major increase in coalition stability compared to a baseline social welfare-maximizing algorithm, while maintaining a very small decrease in social welfare.

Keywords: coalition formation, coalition stability, multi-agent simulation

1 Introduction

Coalition formation is a process of grouping of agents into *coalitions* in order to increase the agents' cooperation. Examples of coalition formation include task allocation or collective purchasing. A goal of coalition formation is often to increase social welfare of the multi-agent system. However, such a goal can generate unrealistic solutions if the agents prefer their own profit to the global social welfare. These self-interested agents would deviate from the computed social welfare-maximizing coalitions. Consider the following example[1]. Three agents x, y, and z, can form coalitions with the following distribution of profit: $\{x = 2, y = 2, z = 3\}, \{x = 3, y = 3\}, \{x = 1, z = 1\}, \{y = 1, z = 1\}, \{x = 0\}, \{y = 0\}$, and $\{z = 0\}$. The first coalition yields the highest total social welfare of 7. However, agents x and y would jointly deviate from this coalition and form the second coalition in order to maximize their own profit.

[1] In this example we assume that social welfare is equal to sum of coalition values, which are in turn calculated by summing up agents' profits.

In coalition formation with self-interested agents, *stability* of the coalitions, which measures the coalition's ability to de-incentivize any sub-coalition of agents from leaving the coalition, must be addressed as a concept that along with the social welfare influences the coalition formation algorithms and solutions. Coalition formation is usually split into three sub-problems [17]: coalition structure generation, solving the optimization problem in each coalition, and division of the coalition's profit among its agents. Coalition stability is relevant to the profit division sub-problem, and is addressed in literature mainly through the concept of a *core*, which is a set of allocations to the agents in a coalition, such that these allocations cannot be improved upon by allocations to a subset of these agents. While the *core* is a strong concept, its computation in a setting where coalition values are generated by general polynomial-time functions requires an evaluation of all $2^{|C|}$ possible sub-coalitions of each coalition C containing $|C|$ agents. In this setting checking whether a solution is in the *core* is *co-NP-complete* [7], and determining whether the *core* is non-empty is $\Delta_2^P - complete$ [7]. This complexity makes the use of the *core* in large-scale systems with thousands of agents infeasible. Therefore instead of the *core* we approach coalition stability using multi-agent simulation. Instead of looking for stable distribution of the coalition value to the agents, we specify an allocation scheme beforehand and let the agents utilize this information to choose more stable coalitions.

Specifically, the contributions of this paper are the following:

1. *An algorithm for large-scale coalition formation of thousands of agents that uses deviations of the agents in order to increase coalition stability.* Our approach uses multi-agent simulation, in which agents make decisions about joining, leaving, and deviating from coalitions. We show the approach in Section 3, and we discuss a deviation strategy in Section 3.1. Finally, we evaluate our algorithm experimentally in Section 4.
2. *An approach for selecting sub-optimal solutions based on their social welfare and coalition stability.* We discuss the ways to select a solution out of a pool of solutions for which stability is unknown and expensive to compute in Section 3.2.

To the best of our knowledge our approach is the first that uses multi-agent simulation to find suboptimal coalition structures with respect to social welfare and coalition stability in large-scale multi-agent systems, in which coalition values are computed using arbitrary polynomial-time functions.

2 Problem Statement

We study the coalition formation problem, in which agents $a_1, a_2, \ldots, a_n \in A$ form coalitions C_i such that each agent belongs to exactly one coalition. We assume that the agents have full information about each others' states. A coalition structure CS is a set of all coalitions C_i that the agents formed. The task is to find a coalition structure that maximizes its social welfare as well as its stability.

In order to measure the social welfare of the formed coalition structure, we first define $v(C)$ as a value of coalition C, and $v(CS)$ as a value of the coalition structure as

$$v(CS) = \sum_{C \in CS} v(C), \tag{1}$$

where $v(C)$ is assigned to the coalition C by a polynomial-time function. The social welfare is represented by a gain metric, which was defined in [8] as $g(CS) = \frac{v(CS)-v(CS_0)}{n}$, where CS_0 denotes the coalition structure of singleton coalitions. The gain shows how much on average an agent benefits from coalition formation. We use gain to measure the social welfare of a coalition structure.

Self-interested agents maximize their own profit, which we define for agent a_j participating in coalition C_i as

$$p_{C_i}(a_j) = v(C_i \cup \{a_j\}) - v(C_i), \tag{2}$$

where the coalition values $v(C_i \cup \{a_j\})$ and $v(C_i)$ are computed right after and right before the agent entered the coalition respectively. The profit reflects marginal contributions of agents to the coalitions, [3] describes games that use this profit sharing scheme as Labor Union games. This definition of profit guarantees that the allocation to the agents granted at the point of entry to the coalition will not change later regardless of further additions of agents to the coalition. We discuss other profit sharing schemes in Section 5.

In order to measure stability of coalition structure CS we need to determine stability of all coalitions $C_i \in CS$. Determining the coalition stability is computationally expensive, because it requires evaluation of all $2^{|C|}$ sub-coalitions. We therefore introduce $stability_\alpha$ to approximate the stability of coalition structures. We say that a coalition C is $\alpha-$stable if no sub-coalition D with $\langle 1, \alpha \rangle$ members can be formed in which some agents would benefit more and no agent would benefit less than in C. Formally,

$$\begin{aligned} &C \text{ is } \alpha-\text{stable iff } \nexists D \subset C, |D| \leq \alpha : \\ &\qquad \exists a_j \in D : p_D(a_j) > p_C(a_j) \wedge \ \forall a_j \in D : p_D(a_j) \geq p_C(a_j). \end{aligned} \tag{3}$$

We denote S_α as the set of $\alpha-$stable coalitions in CS, for which it holds that $\forall \alpha : S_{\alpha+1} \subseteq S_\alpha$. Finally we define $stability_\alpha$ of a coalition structure in terms of α as

$$stability_\alpha(CS) = \frac{|S_\alpha|}{|CS|} \tag{4}$$

where $|CS|$ denotes the number of coalitions in CS. It holds that

$$\lim_{\alpha \to max_{C_i \in CS}(|C_i|)} stability_\alpha(CS) = stability(CS), \tag{5}$$

where $stability(CS)$ is the true stability of CS, which we define as the ratio of stable coalitions in CS. Since $stability_\alpha$ is non-increasing with respect to α, it can serve as an upper estimate of the coalition structure stability.

Finally, we use the price of stability

$$PoS(CS_{sw}, CS_{sa}) = \frac{g(CS_{sw})}{g(CS_{sa})} \tag{6}$$

to show the ratio between the gain of social welfare maximizing solutions CS_{sw} and the gain of solutions reached by behavior of self-interested agents CS_{sa}.

3 Methodology

We find solutions to coalition formation using multi-agent simulation. We extend a multi-agent simulation framework for large-scale coalition formation proposed in [8], in which the agents maximize the social welfare. In that framework the agents use strategies to decide about leaving their coalitions and joining new coalitions. The coalitions are evaluated by a polynomial-time valuation function $f : C \to \mathbb{R}$. This process repeats in an iterative fashion, resulting in an agent-driven search of the state space of coalition structures. While [8] shows almost-optimal performance in small-scale scenarios and stable gain in large-scale scenarios, it does not consider stability of the solutions.

In order to increase stability of coalition structures we extend the algorithm from [8] by first allowing the agents to create more stable sub-coalitions within their coalition by the process of deviation, and second by selecting the best solution out of the pool of solutions generated by the simulation with respect to both social welfare and stability.

3.1 Deviation

Deviation allows agents to leave their current coalition along with other agents from the same coalition. We allow the agents to deviate from their coalitions in order to guide the search towards more stable coalition structures. There are two conditions that a sub-coalition of agents $D \subset C$ must satisfy in order to be able to deviate from a coalition C: 1) $\forall a_j \in D : p_D(a_j) \geq p_C(a_j)$, and 2) $\exists a_j \in D : p_D(a_j) > p_C(a_j)$. These conditions are satisfied by sub-coalitions in which no agent loses profit by deviation and at least one agent gains profit. If an agent finds a sub-coalition that satisfies these conditions, this sub-coalition will deviate from their current coalition C and form a new coalition, thus increasing the stability of the original coalition. Considering all $2^{|C|-1}$ possible sub-coalitions that an agent can be part of is infeasible, therefore agents use a heuristic to guide their search. Some possible heuristics are adding agents to the sub-coalition in order of increasing and decreasing profit, and in random order. Our experiments showed that most stable coalitions were found using the increasing profit heuristic. We therefore let the agents to form the sub-coalitions by adding other agents in order of increasing profit. An agent keeps adding other agents to the new sub-coalition as long as the above-mentioned conditions are met.

Deviation is performed in our model after the agents decide on leaving and joining coalitions. Each iteration of the simulation therefore consists of two steps:

social welfare maximization by leaving and joining coalitions, and stability maximization by deviation. The agents deviate recursively i.e. they try to deviate from the new coalition created by their deviation.

3.2 Solution selection

An inherent advantage of using multi-agent simulation for coalition formation is the fact that it creates a pool of solutions by storing all coalition structures encountered during the search. At the end of the simulation, [8] selects from this pool a solution that maximizes the gain. We propose to select a solution based on both gain and stability metrics. However, computing stability of a coalition structure is computationally expensive, therefore we use $\mathit{stability}_\alpha$ to estimate the true stability of the solutions.

We compute $\mathit{stability}_\alpha$ in an iterative fashion for increasing $\alpha \in \langle 1, \alpha_{max} \rangle$. We only have to determine whether a coalition is α-stable if it is $(\alpha - 1)$-stable. We mark a coalition C α-stable if in all permutations of all combinations of α agents from C some agents lose or no agent gains profit[2]. We then calculate $\mathit{stability}_\alpha$ using Equation 4.

After $\mathit{stability}_\alpha$ of all coalition structures is computed, a multi-criteria optimization is used to select a best coalition structure based on its gain and $\mathit{stability}_\alpha$. Common approaches of multi-criteria optimization are finding Pareto optimal solutions and designing a fitness function. In our experiments we used a simple fitness function that allows us to give preference to any of the criteria:

$$f(CS, \alpha) = w_g \cdot g_{norm}(CS) + w_s \cdot \mathit{stability}_\alpha(CS), \tag{7}$$

where $g_{norm}(CS) \in \langle 0, 1 \rangle$ is a normalized gain of CS, $\alpha \in \langle 1, n \rangle$ is an input parameter that represents the trade-off between quality of solution stability estimate and computation time, and w_g and w_s are weights assigned to the two criteria[3]. Finally, the best coalition structure is returned, such that

$$CS_{best} = \operatorname*{argmax}_{CS} f(CS, \alpha). \tag{8}$$

Figure 1 shows the effect of deviation and solution selection. A combination of both of these approaches yields solutions with higher stability while only sacrificing a small fraction of the gain.

4 Experimental Analysis

We tested our algorithm in two coalition formation scenarios: collective energy purchasing and resource sharing.

[2] All permutations must be considered because the order in which agents join coalitions determines their profit.

[3] Given the values of $g_{norm}(CS)$ and $\mathit{stability}_\alpha(CS)$ for each CS, Pareto optimal solutions can also easily be found.

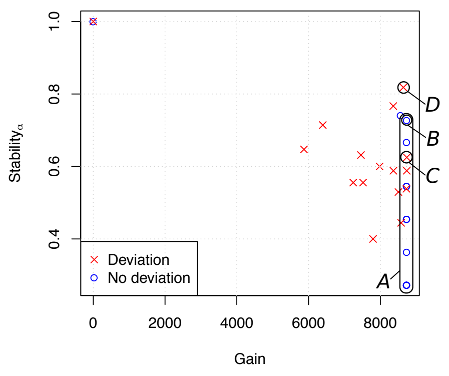


Fig. 1: Gain and $stability_\alpha$ of coalition structures generated by a single simulation with 100 agents, 15 iterations, and $\alpha = 4$. With no deviation and no solution selection, a coalition structure is selected randomly from A, since only the gain is maximized. Solution selection without deviation returns B. Deviation without solution selection returns C, and deviation used along with solution selection returns D.

The collective energy purchasing scenario, proposed in [20], models agents as households that buy electricity based on their requested daily energy profiles. Electricity can be bought at spot and forward markets. The spot market provides amounts of energy based on the current demand, while the forward market provides cheaper electricity which has to be bought ahead of time. Agents form coalitions in order to make their aggregate energy profiles more predictable so they could exploit the reduced prices of the forward market. The valuation function which represents the payment of a coalition was proposed in [20] as

$$v(C) = \sum_{t=1}^{T} q_S^t(C) \cdot p_S + T \cdot q_F(C) \cdot p_F + \kappa(C), \tag{9}$$

where p_S and p_F represent unit prices at the spot and forward markets, respectively, $q_S^t(C)$ represents the amount of energy to be bought at the spot market at time t, and $T \cdot q_F(C)$ represents the total amount of energy to be bought at the forward market for time interval T (in our experiments, $T = 24$ represents a length of a daily energy profile). $\kappa(C) = -|C|^\gamma$ was proposed in [6] to represent the penalty for the coalition size. An algorithm given in [20] computes optimal energy amounts for a coalition given the coalition's aggregate energy profile. Using this algorithm, we obtain energy amounts $q_S^t(C)$ and $q_F(C)$ that we use to compute the coalition value $v(C)$. For this scenario we used a dataset of daily energy profiles of households in Portugal [11]. For each household we averaged daily energy profiles of all days in January 2014 into a single average daily energy profile. The unit prices were set to $p_S = -80$ and $p_F = -70$, as suggested in [20]. We use negative values because the coalition value $v(C)$ is maximized. Following [6] we set $\gamma = 1.1$.

The resource sharing scenario, proposed in [8], models a market in which cooperation is rewarded. Agents operate with resources, each agent can either have a surplus or shortage of each resource. Agents within coalitions are able to transfer their resource surpluses to agents with shortages. The coalition value depends on the amount of resources transferred. The valuation function was proposed in [8] for k resources as

$$v(C) = \sum_{l=1}^{k} min(b_C^+[l], b_C^-[l]) + \kappa(C), \tag{10}$$

where $b_C^+[l]$ is the positive balance for resource l, which is the sum of surpluses of resource l over all agents in coalition C, and $b_C^-[l]$ is an absolute value of the negative balance computed with the shortages, respectively. $\kappa(C) = -|C|^\gamma$ [6] captures the penalty for the coalition size. We used an international trade dataset provided by the World Trade Organization [12], which stores import and export amounts in US dollars between 167 countries in 17 commodity types. The amount of each resource of each agent was computed as the difference between export and import amounts of the given country in the year 2014. Positive and negative values of the resulting resource amounts denote surplus and shortage respectively. Similarly as in [8], we set $\gamma = 2$ for the resource sharing scenario to prevent the grand coalition from being the trivial gain-maximizing solution.

Because the use of κ as a coalition size penalty causes agents to form small coalitions, we instead define κ as $\kappa = min(-|C|+\mu, 0)^\gamma$, which effectively allowed us to increase the average coalition size and thus make the problem harder to compute due to its exponential complexity. In our experiments we set $\mu = 10$.

Several agent strategies were studied in [8]. In our experiments we use the *local search* strategy [8], in which the agents perform a best response move to new coalitions i.e. the agents select coalitions which grant them maximal marginal profit. If the search reaches a local optimum for all agents, a random jump is applied by all agents in order to escape this optimum.

We used two values of α for evaluation of $stability_\alpha$. For the solution selection algorithm, we set $\alpha_{ss} = 3$ to allow the algorithm to quickly compute $stability_\alpha$ of multiple solutions, and for the final stability verification we set $\alpha = 4$ to obtain a better final stability estimate. In order to give equal preference to both gain and stability we set the weights $w_g = w_s = 1$.

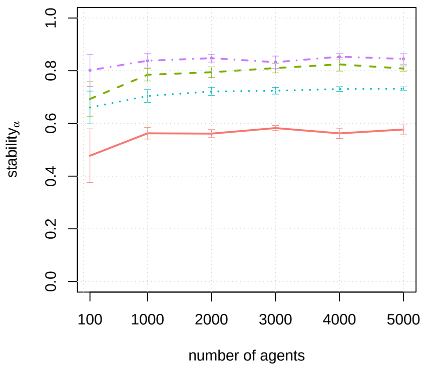
In order to achieve reasonable run-times of our algorithm, we used the following number of iterations N in our experiments. For instances with number of agents $n < 100$ we set $N = 100$ and for instances with $n > 100$ we set $N = 10$.

We ran our Java implementation of the proposed algorithms on 2.7 GHz Intel Xeon E5 CPU with 2 GB of memory. We averaged our results over 10 random runs [4].

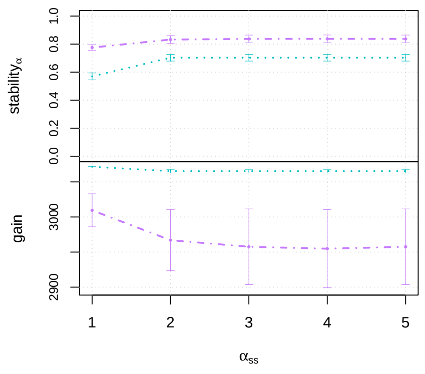
4.1 Experiment Results

We compared results of our algorithms with the baseline multi-agent simulation algorithm for coalition formation [8] using the $stability_\alpha$ and *price of stability* metrics. Average values of $stability_\alpha$ and *price of stability* are shown in Table 1. The first row of Table 1 shows results of the baseline algorithm. The following rows show how the average $stability_\alpha$ increases when we plug in the proposed stability-increasing methods. As expected, the average *price of stability* is increasing with the increase in $stability_\alpha$, but the increase in *price of stability* is very low compared to the significant improvement in $stability_\alpha$. Table 1 therefore shows that our algorithms find solutions with much higher stability while only sacrificing a fraction of the social welfare.

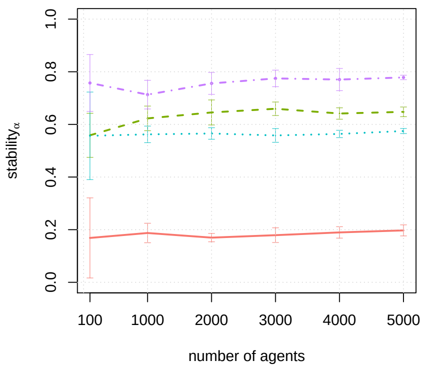
[4] Random runs are necessary because agents make decisions in random order.



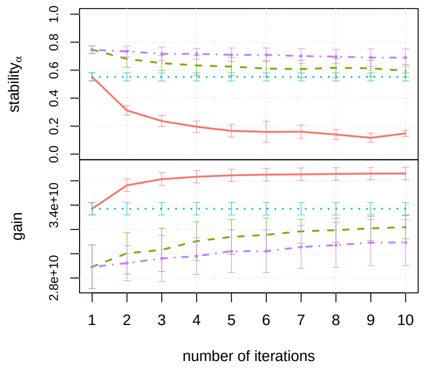
(a) *Stability*$_\alpha$ in collective energy purchasing scenario with 100 to 5000 agents



(b) Effect of α_{ss} on *stability*$_\alpha$ and gain in collective energy purchasing scenario with 1000 agents and $\alpha = 5$



(c) *Stability*$_\alpha$ in resource sharing scenario with 100 to 5000 agents



(d) Effect of number of iterations N on *stability*$_\alpha$ and gain in resource sharing scenario with 1000 agents

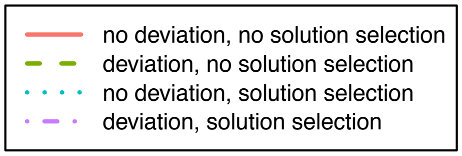

no deviation, no solution selection
deviation, no solution selection
no deviation, solution selection
deviation, solution selection

Fig. 2: 2a, 2c) Stability achieved by our algorithms and the baseline algorithm [8] in the collective energy purchasing and resource sharing scenarios: combination of deviation and solution selection algorithms yields highest stability. 2b) Effect of α_{ss} on *stability*$_\alpha$ and gain: higher α_{ss} yields better stability estimate and therefore increases stability of the selected solution. 2d) Effect of number of iterations N on *stability*$_\alpha$ and gain: higher N yields higher gain because the agents have more opportunity to form coalitions, which naturally leads to a decrease in stability. Decrease in *stability*$_\alpha$ achieved by our algorithms is significantly lower than the decrease achieved by the baseline algorithm. Error bars show standard deviation of aggregated variables.

Table 1: Trade-off between average stability and average price of stability achieved by our algorithms with $\alpha = 4$ and $n = \langle 20, 5000\rangle$.

Algorithm		**Results**	
Deviation	**Solution selection**	**Average** $stability_\alpha$	**Average** PoS
No	No	0.3914	-
Yes	No	0.6299	1.0308
No	Yes	0.6665	1.0210
Yes	Yes	0.8185	1.0629

Stability of our solutions is depicted in Figure 2a in collective energy purchasing scenario and in Figure 2c in resource sharing scenario. The use of solution selection algorithm never decreases the stability of the solutions, therefore the solutions generated by the solution selection algorithm always dominate the baseline algorithm with respect to stability. This dominance is not guaranteed by the deviation algorithm. However, in most instances the deviation algorithm achieves higher $stability_\alpha$ than the baseline algorithm. Finally, the highest increase in $stability_\alpha$ is achieved in majority of instances when the deviation and the solution selection algorithms are used together. As shown in Table 1, the average $stability_\alpha$ increases from 39% achieved by the baseline algorithm to 82% achieved by the combination of deviation and solution selection algorithms.

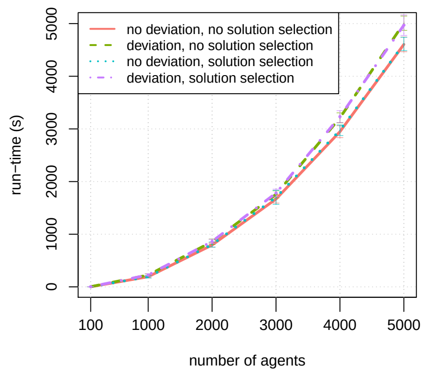


Fig. 3: Run-time of our algorithms and the baseline algorithm [8] in collective energy purchasing scenario with number of iterations $N = 10$.

The solution selection algorithm evaluates $stability_\alpha$ of all coalition structures for given α_{ss}. Figure 2b shows $stability_\alpha$ and gain for varying values of α_{ss}, where $\alpha = 5$. As expected, the $stability_\alpha$ of the selected solution is increasing with increasing α_{ss}, since higher α_{ss} provides a better stability estimate. However, due to the inherent trade-off between coalition stability and social welfare, the gain decreases with increasing α_{ss}. Figure 2b only shows algorithms that include solution selection and are therefore affected by changing α_{ss}.

The number of iterations N affects the quality of the resulting coalition structure. We show $stability_\alpha$ and gain of our algorithm for varying numbers of iterations N in Figure 2d. With the increasing number of iterations the agents have more opportunity to cooperate by creating coalitions, which leads to an increase in gain. However, higher social welfare might result in lower stability of the coalitions. This effect is most obvious in the results of the baseline algorithm, in which due to the increase in gain the $stability_\alpha$ drops significantly. However, when we plug in the stability-increasing approaches proposed in this paper, the decrease in $stability_\alpha$ is much slower.

In practice the run-time of an algorithm is an important factor. Figure 3 shows the run-time of our algorithm for increasing numbers of agents. Interestingly, the run-time does not change significantly when we plug in the proposed stability-increasing algorithms. Deviation of the agents has a higher impact on run-time than solution selection, because it is executed by all agents in each iteration. Run-time of our algorithm can be decreased by decreasing the number of iterations N, however such an approach might yield solutions of lower quality, as shown in Figure 2d.

We also experimented with the state-of-the-art algorithm for coalition formation C-Link [6] in order to determine its ability to create stable coalitions. C-Link, like our approach, can also be used with arbitrary valuation functions, and social welfare of its solutions is comparable with results of the baseline algorithm [8]. Even though C-Link was not designed for use with self-interested agents, the algorithm might still inherently create stable coalitions. However, our experiments showed that the only stable coalitions in solutions generated by C-Link in the collective energy purchasing and resource sharing scenarios are singleton coalitions, which by definition in Equation 3 are always stable. This result shows that multi-agent simulation, especially along with the stability-increasing methods proposed in this paper, is better suited for coalition formation of self-interested agents than other state-of-the-art coalition formation algorithms.

5 Discussion

We will now discuss some design choices that have to be made when designing a multi-agent system for coalition formation of self-interested agents. We will discuss various profit sharing schemes, definitions of stability, and behaviors of self-interested agents. Then we will analyze time complexity and convergence of our algorithms. Finally we will discuss practical usefulness of our approach.

Several profit sharing schemes have been proposed in the literature. Equal sharing [14] divides the coalition value equally among all its agents, fair value sharing [3] defines agents' payoff as marginal contribution to the coalition, labor union sharing [3], which we use in our experiments, defines agents' payoff as agents' marginal contribution to the coalition at the time of entry, and Shapley sharing [3] assigns payoffs based on agents' Shapley values. Among these sharing schemes, labor union is the only one in which consequent additions to the coalition do not affect agent's payoff assigned at the time of coalition entry. It is the only sharing scheme that models marginal contribution and at the same time is reasonably computationally efficient, which is why we used it in our experiments.

Several concepts have been used to describe coalition stability. Nash equilibrium describes a state in which no agent has an incentive to unilaterally deviate.

A stronger concept is a core, which is a set of profit assignments to agents, such that no subset of agents in a coalition has an incentive to jointly deviate from the coalition. Our definition of stability in Equation 3 is following the concept of the core. One might also consider a stricter version of the definition, in which all deviating agents must benefit by the deviation, i.e. $\forall a_j \in D : p_D(a_j) > p_C(a_j)$. However, this strict definition of stability yields high stability values for arbitrary coalitions, and therefore renders the problem less interesting.

Our algorithm searches the state space of coalition structures using three actions of the agents: leave a coalition, join a coalition, and deviate from a coalition. We designed these actions in order to search for coalition structures with high values of both social welfare and coalition stability. However, the space of possible agents' actions is not limited to actions used in this paper. For example, agents from multiple coalitions could jointly deviate, agents could decide whether to allow other agents to enter their coalition, agents could force other agents in their coalition to leave, etc. Adding new actions to the agents' action space will lead to new behavior of the multi-agent system. When designing agents' actions we must take into account the specific problem that is being solved, the effect of the actions on agents' behavior, and the computational complexity of the designed actions.

Searching the exponential state space of coalition structures can lead to exponential worst-case time complexity of the search algorithms. However, we show a polynomial time complexity of both deviation and solution selection algorithms with respect to number of agents n and a constant α_{ss} bound. Deviation of a single agent requires sorting the agents in the coalition by marginal profit. The agents can deviate recursively, therefore the complexity of deviation of a single agent is $O(\sum_{i=1}^{n}(i \cdot log(i)))$, for which an upper bound is $O(n^2 log(n))$. Solution selection searches through all permutations of all combinations of size $\langle 1, \alpha_{ss} \rangle$ of agents in a coalition. Evaluating a single coalition therefore requires $O(\sum_{i=1}^{\alpha_{ss}}(i! \cdot \binom{n}{i}))$ steps. For $\alpha_{ss} << n$ it holds that

$$\sum_{i=1}^{\alpha_{ss}} \left(i! \cdot \binom{n}{i} \right) \leq \alpha_{ss} \cdot \alpha_{ss}! \cdot \binom{n}{\alpha_{ss}} \leq \alpha_{ss} \cdot n^{\alpha_{ss}}, \tag{11}$$

therefore the worst time complexity of finding $stability_\alpha$ for a single coalition with $\alpha = \alpha_{ss}$ is $O(n^{\alpha_{ss}})$. The worst-case time complexity of the solution selection algorithm is therefore $O(N \cdot n^{\alpha_{ss}+1})$, given the input of N coalition structures, each containing at most n coalitions. Worst-case time complexity of the baseline coalition formation algorithm is $O(N \cdot n^2)$, as was shown in [8].

Building on the complexity analysis above, Table 2 shows worst-case time complexity of our algorithms. Since we treat α_{ss} as a small constant, we get a polynomial complexity for all proposed algorithms. The analysis in Table 2 is very conservative, since we assume that each coalition structure contains n coalitions, each coalition is composed of n agents, and each sub-coalition deviating from coalition C is of size $|C| - 1$. The analysis does not include complexity of the valuation functions, as these are given as an input to the simulation. However, both collective energy purchasing and resource sharing valuation functions require constant time with respect to the number of agents n, and therefore do not affect the complexity analysis. Our algorithm is centralized, therefore we do not assume any additional cost of communication between agents, as would be the case with algorithms that distribute the computation among the agents.

Table 2: Worst-case time complexity of our algorithms and the baseline [8] as a function of number of iterations N, number of agents n, and a small constant α_{ss}. In our experiments we set $\alpha_{ss} = 3$.

Algorithm		**Worst-case time complexity**
Deviation	**Solution selection**	
No	No	$O(N \cdot n^2)$
Yes	No	$O(N \cdot n^3 \cdot log(n))$
No	Yes	$O(N \cdot n^{max(\alpha_{ss}+1,2)})$
Yes	Yes	$O(N \cdot n^{max(\alpha_{ss}+1,3)} \cdot log(n))$

An important aspect of an algorithm is its convergence behavior. The baseline algorithm converges if the random jump is not used [8]. Similarly, if the agents were only allowed to deviate, the simulation would converge from any initial state because each deviation splits coalitions and decreases the average coalition size. Therefore separate maximization of social welfare, as well as separate maximization of coalition stability, is guaranteed to converge. However, combining these steps in order to maximize both metrics does not guarantee convergence, because social welfare and coalition stability are somewhat contradictory goals.

Our algorithm can be used in real-world scenarios where coalition stability has to be considered. Solutions with high stability are more realistic, because they reflect decision making of self-interested agents. Such decision making might lower the social welfare of the solution, but as we showed in Table 1, the *price of stability* of our solutions only slightly increases with major increase in stability. Furthermore, the weights w_g and w_s in Equation 7 can be adjusted to give preference to either the social welfare or stability.

6 Related Work

Many algorithms have been proposed that search for a coalition structure with optimal or sub-optimal social welfare without considering coalition stability. Among them we highlight dynamic programming approaches [22, 15, 5], hierarchical clustering for large numbers of agents [6], and aproaches that use multi-agent simulation [8, 10, 13]. We refer the reader to a recent comprehensive survey on coalition structure generation in [16].

Theoretical properties of stability in coalition formation have been studied extensively. [18] provides an overview of social-welfare and stability in the coalition formation setting. [14] studies the existence of core stable coalition structures with respect to profit sharing rules and agents' preferences over coalitions.

Algorithms have been proposed to find stable coalitions in coalition formation games. [3] proposes algorithms to find Nash equilibria for various profit sharing schemes. Unlike [3], which assumes deviations of single agents only, we look for coalitions that are stable with respect to deviations of groups of agents. [1] computes profit sharing between agents that grants stability of the solutions for subadditive games only. Again, we do not require such restrictions on the valuation functions. An iterative approach for finding core-stable coalitions was proposed in [2]. While the approach in [2] is similar to ours, it can only be used in small scale scenarios due to its high complexity, as was shown in [4], where the algorithm from [2] was improved and empirically tested. An auction-based algorithm for creation of stable coalitions in large scale e-marketplaces is proposed in [21]. Coalition stability in a request for proposal domain is studied in [9], in which a negotiation protocol for coalition formation is introduced. There stability is demonstrated by showing that allowing agents to deviate from pure strategy profiles is not beneficial. It is unclear whether algorithms proposed in [21] and [9] can be modified for use with general polynomial-time valuation functions. Finally, [19] proposed algorithms that maximize social welfare and find stable payoff division among agents. However, the algorithms restrict the allowed size of coalitions, which renders the algorithms unusable in large-scale problem instances where large coalitions might occur.

7 Conclusion

Algorithms that find stable coalition structures are often proposed for settings that restrict the properties of the valuation functions. Practical aspects of the high complexity of finding stable coalitions for large-scale multi-agent systems are often not considered.

In this work we proposed an approach for increasing coalition stability in large-scale coalition formation with self-interested agents and arbitrary valuation functions. We modeled agent behavior using multi-agent simulation, in which we let agents to choose profitable coalitions and deviate from unstable coalitions. At the end of the simulation, we selected a solution out of a pool of generated coalition structures based on its social welfare and stability. We experimentally showed that our approach is able to increase the stability of the solutions in two real-world scenarios. We also showed that the necessary price for this increase in stability that our algorithm incurs to the social welfare is very low.

Some open questions and areas of further research, which we plan to investigate, include coalition formation of agents with limited information, distributed asynchronous simulation of coalition formation, and coalition formation with dynamically changing valuation functions.

8 Acknowledgements

This work was supported by the US National Science Foundation via Award No. CNS-1544705.

Bibliography

[1] Anshelevich E, Sekar S (2015) Computing stable coalitions: Approximation algorithms for reward sharing. Web and Internet Economics: 11th International Conference, WINE 2015

[2] Arnold T, Schwalbe U (2002) Dynamic coalition formation and the core. Journal of Economic Behavior & Organization

[3] Augustine J, Chen N, Elkind E, Fanelli A, Gravin N, Shiryaev D (2011) Dynamics of profit-sharing games. IJCAI'11: 21st International Joint Conference on Artificial Intelligence

[4] Bistaffa F, Farinelli A (2013) A fast approach to form core-stable coalitions based on a dynamic model. 2013 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2013

[5] Cruz-Mencía F, Cerquides J, Espinosa A (2013) Optimizing performance for coalition structure generation problems' idp algorithm. 2013 International Conference on Parallel and Distributed Processing Techniques and Applications

[6] Farinelli A, Bicego M, Ramchurn S, Zucchelli M (2013) C-link: A hierarchical clustering approach to large-scale near-optimal coalition formation. Twenty-Third International Joint Conference on Artificial Intelligence

[7] Greco G, Malizia E, Palopoli L, Scarcello F (2011) On the complexity of the core over coalition structures. Twenty-Second International Joint Conference on Artificial Intelligence

[8] Janovsky P, DeLoach SA (2016) Multi-agent simulation framework for large-scale coalition formation. 2016 IEEE/WIC/ACM International Conference on Web Intelligence

[9] Kraus S, Shehory O, Taase G (2003) Coalition formation with uncertain heterogeneous information. Proceedings of the second international joint conference on Autonomous agents and multiagent systems

[10] Lerman K, Shehory O (2000) Coalition formation for large-scale electronic markets. Fourth International Conference on MultiAgent Systems

[11] Lichman M (2013) UCI machine learning repository. Available at `https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014`

[12] World Trade Organization (n.d.) Available at `http://stat.wto.org/StatisticalProgram/WSDBStatProgramSeries.aspx`, Accessed: 2016-03-03

[13] Merida-Campos C, Willmott S (2004) Modelling coalition formation over time for iterative coalition games. Third International Joint Conference on Autonomous Agents and Multiagent Systems AAMAS 2004

[14] Pycia M (2012) Stability and preference alignment in matching and coalition formation. Econometrica

[15] Rahwan T, Jennings NR (2008) An improved dynamic programming algorithm for coalition structure generation. 7th International Conference on Autonomous Agents and Multiagent Systems

[16] Rahwan T, Michalak TP, Wooldridge M, Jennings NR (2015) Coalition structure generation: A survey. Artificial Intelligence

[17] Sandholm T, Larson K, Andersson M, Shehory O, Tohmé F (1999) Coalition structure generation with worst case guarantees. Artificial Intelligence

[18] Sandholm TW, Lesser VR (1997) Coalitions among computationally bounded agents. Artificial Intelligence

[19] Shehory O, Kraus S (1999) Feasible formation of coalitions among autonomous agents in nonsuperadditive environments. Computational Intelligence

[20] Vinyals M, Bistaffa F, Farinelli A, Rogers A (2012) Coalitional energy purchasing in the smart grid. 2012 IEEE International Energy Conference and Exhibition, ENERGYCON 2012

[21] Yamamoto J, Sycara K (2001) A stable and efficient buyer coalition formation scheme for e-marketplaces. AGENTS '01 Proceedings of the fifth international conference on Autonomous agents

[22] Yun Yeh D (1986) A dynamic programming approach to the complete set partitioning problem. BIT Numerical Mathematics