Coalitional Structure Generation in Skill Games

Yoram Bachrach† and **Reshef Meir**‡ and **Kyomin Jung**§ and **Pushmeet Kohli**†
† Microsoft Research, Cambridge, UK
‡ Hebrew University Jerusalem, Israel
§ KAIST, Daejeon, Korea

Abstract

We consider optimizing the coalition structure in *Coalitional Skill Games (CSGs)*, a succinct representation of coalitional games (Bachrach and Rosenschein 2008). In CSGs, the value of a coalition depends on the tasks its members can achieve. The tasks require various skills to complete them, and agents may have different skill sets. The *optimal coalition structure* is a partition of the agents to coalitions, that maximizes the sum of utilities obtained by the coalitions. We show that CSGs can represent any characteristic function, and consider optimal coalition structure generation in this representation. We provide hardness results, showing that in general CSGs, as well as in very restricted versions of them, computing the optimal coalition structure is hard. On the positive side, we show that the problem can be reformulated as constraint satisfaction on a hyper graph, and present an algorithm that finds the optimal coalition structure in polynomial time for instances with bounded tree-width and number of tasks.

Introduction

Consider a network with outposts, each with several sensors covering a certain geographical area. Suppose the system must track objects, each traveling through a different path. By partitioning the outputs into teams, we wish to maximize the number of tracked objects, where each team must include an optical sensor and a motion sensor. Alternatively, consider rescuers in a disaster zone with people trapped in unknown places, which require different skills to get to. Each rescuer has some of these skills, but can only be in one place at a time. For example a firefighter, a medic and a stretcher-carrier have together the set of skills required to rescue a person from a burning room. What is the best partition into teams that maximizes the number of people rescued? Such problems can be modeled as optimizing the coalition structure in skill games. Skill games (Bachrach and Rosenschein 2008) can also model other domains, including task allocation, grid computation and robotics.

A key issue in multi-agent systems is agent cooperation. Such domains can be handled within the framework of *cooperative (or coalitional) games*. In such games agents form coalitions to pursue a joint cause. Much research focused on settings where only one coalition can be formed. However, many domains allow the formation of *several* mutually exclusive coalitions, where each coalition can secure its profits independently. Assuming no externalities between coalitions, an important goal is to partition the agents in a way that maximizes the total value gained by all coalitions, i.e. the social welfare. This problem is known as *finding the optimal coalitional structure*. Representation languages for cooperative games define the value generated by each coalition. Generally any coalition may have a different value, so a naïve representation requires space exponential in the number of agents. This emphasizes the need for a *succinct representation language*, with short descriptions even for many agents. Many representations were suggested. Some guarantee a polynomial description, but only represent restricted classes of games (for example (Peleg and Sudholter 2007; Deng and Papadimitriou 1994)). Others represent any game, but the description may be exponential in the worst case (Ieong and Shoham 2005). One interesting model is CSGs — Coalitional Skill Games (Bachrach and Rosenschein 2008). In CSGs agents cooperate to complete tasks. Each task requires a set of skills, and a coalition accomplishes the task if its members have all the required skills.

Our contribution We study the problem of optimal coalition structures in CSGs. We show that the general problem is computationally hard and remains so even under severe restrictions on the number of tasks or the number of skills they require. We then propose a Fixed Parameter Tractable (FPT) algorithm, which solves in polynomial time instances whose underlying structure has a *bounded tree-width*. In addition, we show that CSGs can represent *any* coalitional game, so our positive results provide a general algorithm for coalitional structure generation. We note however that representing and solving arbitrary games as CSGs would be inefficient, thus our method should be used in domains where the game has a simple CSG representation.

Preliminaries

A transferable utility (TU) coalitional game is composed of a set of n agents, I, and a characteristic function mapping any subset (coalition) of the agents to a rational value $v : 2^I \rightarrow \mathbb{Q}$, indicating the total utility these agents achieve

together. A coalitional game is *monotone* if for all coalitions $C' \subset C$ we have $v(C') \leq v(C)$. In some domains, each coalition can obtain its value, without being affected by the formation of additional coalitions. For such domains, we follow the *Coalition Structure (CS)* model introduced in (Aumann and Dreze 1974), where agents can split into several teams which work *simultaneously*. A coalition structure is simply a partition of the agents into disjoint coalitions. Formally, $CS = (C^1, \ldots, C^l)$ is a *coalition structure* over a set of agents I if $\cup_{i=1}^{l} C^i = I$ and $C^i \cap C^j = \emptyset$ for all $i \neq j$; The set $\mathcal{CS}(I)$ denotes all possible coalition structures on I. We overload the notation for values by denoting $v(CS) = \sum_{C^j \in CS} v(C^j)$. Given a game $\langle I, v \rangle$, the *optimal* coalition structure $CS^* \in \mathcal{CS}(I)$ is the partition that maximizes social welfare, i.e. for any other $CS \in \mathcal{CS}(I)$ we have $v(CS) \leq v(CS^*)$.

Skill Games We briefly review the definitions of CSGs from (Bachrach and Rosenschein 2008). A *skill domain* is composed of a set of agents, $I = \{a_1, \ldots, a_n\}$, a set of tasks $T = \{t_1, \ldots, t_m\}$, and a set of skills $S = \{s_1, \ldots, s_k\}$. Each agent a_i has a set of skills $S_i \subset S$, and each task t_j requires a set of skills $S(t_j) \subset S$, where $S(t_j) \neq \emptyset$. We denote the set of skills a coalition C has by $S(C) = \cup_{a_i \in C} S_i$. We say a coalition $C \subset I$ can perform task t_j if every skill required to perform the task is owned by some agent in the coalition (i.e. $S(t_j) \subset S(C)$). We denote the set of tasks a coalition C can perform as $T(C) = \{t_j \in T | S(t_j) \subset S(C)\}$. We denote the set of skills required to perform a set of subtasks $T' \subset T$ by $S(T') = \cup_{t_j \in T} S(t_j)$. A *task value function* maps a subset of the tasks a coalition achieves to a rational value: $u : 2^T \rightarrow \mathbb{Q}$ [1]. We generally assume u is monotone, so if $T_1 \subset T_2 \subseteq T$, then $u(T_1) \leq u(T_2)$. Consider a skill domain; we define the skill game as follows:

Definition 1. *A* Coalitional Skill Game (CSG) *is the coalitional game $\langle I, v \rangle$, where $v(C) = u(T(C))$, i.e. the value of the tasks that coalition C can perform.*

In their most general form, without restrictions on u, the description length of a CSG is still exponential in the number of tasks. Thus we define a simpler class of games by restricting the function u. A concise representation allows tasks to have different weights, and the characteristic function is the weighted sum of all accomplished tasks.

Definition 2. *A* Weighted Task Skill Game (WTSG) *is a CSG with a weight function for tasks $w : T \rightarrow \mathbb{Q}$. The characteristic function u is defined as $\forall T' \subseteq T$, $u(T') = w(T')$, where $w(T') = \sum_{t \in T'} w(t)$. We denote by WTSG$_+$ games where all weights are positive.*

In a very restricted form of skill games, there is only one task t. In this case, u only needs to distinguish between coalitions that can perform t, and coalitions that cannot. Dropping all skills that are not required to perform t, a coalition C wins if it manages to cover *all* the skills.

Definition 3. *A* Single Task Skill Game (STSG) *is a WTSG where there is only one task t with unit weight, thus $v(C) = 1$ if $S(C) = S$ and $v(C) = 0$ otherwise.*

[1] We assume a binary encoding for numbers.

Finding the Optimal Coalitional Structure

The work of (Bachrach and Rosenschein 2008) which proposed the CSG model did not discuss what characteristic functions can be represented as CSGs. Consider WTSGs of Definition 2. It is easy to see that instances of WTSG$_+$ cannot express all games (e.g. non-monotonic games), but as the following theorem shows, if arbitrary weights are allowed then WTSG are highly expressive.

Theorem 1. *Any game $\langle I, v \rangle$ can be expressed as a WTSG.*

Proof. Consider a characteristic function $u : 2^I \rightarrow \mathbb{Q}$. We construct a WTSG representation as follows. For any agent $i \in I$ we create a single skill s_i, which only that agent has. For any singleton coalition $C = \{i\}$ we create a single task $t_{\{i\}}$, which requires s_i, and set its weight so that $u(\{i\}) = w(t_{\{i\}}) = v(\{i\})$. We now proceed from coalitions of size x to coalitions of size $x + 1$. For any coalition C we compute the sum of weights of all tasks that can be performed by subsets of C, and add a task t_C which requires C and whose weight is exactly the difference $w(t_C) = v(C) - \sum_{C' \subsetneq C} w(t_{C'})$. Thus we have by induction that $u(T(C)) = v(C)$ for all $C \subseteq I$. □

Theorem 1 states the WTSGs are fully expressive (when negative weights are possible). We later propose an algorithm for computing the optimal structure in CSGs. Since any game can be converted into such a WTSG, this algorithm is a *general algorithm* for coalition structure generation. However, the above construction can result in a very large CSG representation, with a number of tasks exponential in the number of the agents. Thus, our methods are tailored to domains with a succinct CSG representation.

Definition 4 (OPT-CS-CSG). *We are given a CSG $\langle I, v \rangle$ and a rational number r, and are asked whether the value of the optimal coalitional structure for this game is at least r, i.e. if there is $CS \in \mathcal{CS}(I)$ such that $v(CS) \geq r$.*

We use the class name as a suffix (e.g. "OPT-CS-WTSG" is finding the optimal structure in WTSG). In WTSGs and STSGs, the description is polynomial in the number of tasks, skills and players. Computing the value of any structure can be done in polynomial time, so OPT-CS-CSG is *in* $\mathcal{NP}$.

Hardness Results

We now show that computing the optimal coalition structure is hard, even for very restricted classes of the problem.

Single Task We show that OPT-CS is $\mathcal{NP}$-hard even for single task skill games (STSGs), using a reduction from SET-SPLITTING, which is a known $\mathcal{NP}$-complete problem (Garey and Johnson 1979): We are given a set of elements U and a family F of subsets $S_1, \ldots, S_m$, such that each $S_i \subseteq U$, and are asked if there is a partition of U into C_A, C_B such that each S_i is split between C_A and C_B (i.e. for any i, $C_A \cap S_i \neq \emptyset$ and $C_B \cap S_i \neq \emptyset$).

Theorem 2. *OPT-CS-STSG is $\mathcal{NP}$-complete.*

Proof. For any set $S_j \in F$ in the original SET-SPLITTING instance, we define a skill s_j that all the agents in S_j own, and set $r = 2$. A partition (C_A, C_B) of U splits all sets in F iff each of C_A, C_B has all skills, so both C_A, C_B are coalitions that can perform the task. Thus the optimal CS has a utility of at least 2 iff there is a set-splitting. □

Thus, merely testing whether there exists a structure with utility higher than 1 is hard. Observe that CSGs has a graphical interpretation as a hypergraph when replacing the term "elements" (or agents) with "vertices" and "subsets of elements" with "hyperedges", where every hyperedge corresponds to a skill. This hypergraph is the *skill graph* of the CSG problem and we use it in the positive results section.

Tasks that require only two skills Theorem 2 considers a single task requiring any number of skills. We now show that OPT-CS is $\mathcal{NP}$-complete even when tasks require at most two skills, through a reduction from NEGATIVE-CUT, another known $\mathcal{NP}$-complete problem (Deng and Papadimitriou 1994). In NEGATIVE-CUT, we are given an undirected weighted graph $G = \langle V, E, g\rangle$, where $g : E \to \mathbb{Q}$ is a weight function, and are asked if there is a cut in the graph (i.e. a partition of V to two distinct sets (V_1, V_2)), such that $\sum_{i\in V_1, j\in V_2} g(i,j) \leq 0$.[2]

Theorem 3. *OPT-CS-WTSG$_+$ is $\mathcal{NP}$-complete even if the input is limited to tasks that require at most two skills, and skills that are owned by at most two agents.*

Proof. Let $G = \langle V, E, g\rangle$ be an undirected graph with weight function $g : E \to \mathbb{Q}$. For any edge set $E' \subseteq E$, denote by $g(E') = \sum_{e\in E'} g(e)$ the weight of the subset. We construct our WTSG as follows: the set of agents I is the set of vertices V. We denote by $P \subseteq E$ the set of all positive edges, i.e. $P = \{e \in E : g(e) > 0\}$, and its complement by $N = E \setminus P$. For every positive edge $e = (i, i') \in P$, define a task t_e and two skills $s_{e,i}, s_{e,i'}$. The task t_e requires the two skills, and each of them is owned exclusively by one agent (either i or i'). Set the value to $w(t_e) = g(e) > 0$. For every negative edge $f = (j, j') \in N$, define a task t_f and a single skill s_f, owned by both agents j, j'. Set the value to $w(t_f) = -g(f) > 0$. Finally, set $r = \sum_{e\in E} |g(e)|$.

The set of coalitional structures $\mathcal{CS}(I)$ is the set of multicuts of G (i.e. partitions of $I = V$). The notation CS is used in both contexts. For any non overlapping coalitions (vertex subsets $C, C' \subseteq I$), denote by $g(C, C') = \sum_{i\in C, j\in C'} g(i,j)$ the weight of the partial cut (C, C'). For any multicut $CS = (C_1, \ldots, C_c)$ of I, denote by $g(CS) = \sum_{C,C'\in CS} g(C, C')$ the weight of the multicut. This is the total weight of all edges crossing the multicut. Figure 1 shows an example of a graph and a multicut. To complete the reduction, we need the following two lemmas.

[2] In the original version of the problem the requirement was for a strictly negative cut. However, the weak inequality can be proved by replacing the term 0 with a very small $\epsilon > 0$. The term only has to be smaller than the difference between any two sums of edge weights. Assuming a binary number encoding, it is sufficient to set $0 < \epsilon < 2^{-2n\cdot|x|}$, where $|x|$ is the length of the original problem. Note edges may not have a zero weight.

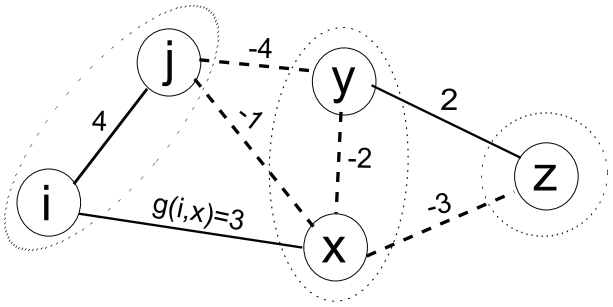

Figure 1: In the example graph $r = \sum_{e\in E} |g(e)| = 19$, and $CS = (\{i, j\}, \{x, y\}, \{z\})$. Viewing CS as a multicut, its weight is $g(CS) = -3$. Solid-line tasks need both agents, and dashed lines tasks can be carried by each agent. Thus the value of CS is $v(CS) = 4 + 2(1 + 4 + 3) + 2 = 22$, which indeed equals $r - g(CS)$ as lemma 2 states.

Lemma 1. *G has a non-positive multicut iff it has a non-positive cut.*

Proof. Suppose there is a multicut CS of the vertices such that $g(CS) \leq 0$, where $c = |CS| \geq 3$. Let $C', C'' \in CS$ be two coalitions that maximize $g(C', C'')$. We merge the C' and C'' and get a new structure CS' of size $c - 1$, and weight $g(CS') = g(CS) - g(C', C'')$. If *all* partial cuts in CS were negative, then clearly $g(CS') < 0$, since it is a sum of negative terms. Otherwise, for the maximal partial cut $g(C', C'') \geq 0$, so $g(CS') \leq g(CS) \leq 0$. In any case, we get a new negative cut of size $c-1$. We repeat the process until we have a negative cut (C_1, C_2). The other direction is trivial, as every cut is a multicut. □

Lemma 2. *For any structure CS, $v(CS) = r - g(CS)$.*

Proof. Each positive task t_e is performed (once) iff its corresponding edge $e = (i, i')$ does not cross the multicut (so both required agents are in the same coalition). Each negative task t_f is performed at least once, and twice if the two agents capable of performing it belong to different coalitions, i.e. the corresponding edge $f = (j, j')$ crosses the multicut. We denote by $E(CS) \subseteq E$ the set of all edges crossing the multicut CS. Summing payments over performed tasks we get:

$$\begin{aligned} v(CS) &= \sum_{e\in P\setminus E(CS)} w(t_e) + \sum_{f\in N} w(t_f) + \sum_{f\in N\cap E(CS)} w(t_f) \\ &= \sum_{e\in P\setminus E(CS)} g(e) + \sum_{f\in N} -g(f) - \sum_{f\in N\cap E(CS)} g(f) \\ &= \sum_{e\in P} g(e) - \sum_{e\in P\cap E(CS)} g(e) + \sum_{f\in N} -g(f) - \sum_{f\in N\cap E(CS)} g(f) \\ &= \sum_{e\in E} |g(e)| - \sum_{e\in E(CS)} g(e) = r - g(CS) \ . \end{aligned}$$ □

Due to Lemma 2, the value of a coalition structure CS reaches the threshold r iff the weight of CS as a multicut is non-positive. Thus, a partition CS for which $v(CS) \geq r$ exists iff G contains a non-positive multicut, which by Lemma 1 happens iff G contains a non-positive cut. □

Theorems 2 and 3 indicate that OPT-CS in general CSGs is very hard computationally, as it is hard to solve even highly restricted classes of WTSGs.

Positive Results

Definition 5. *Given a CSG, its corresponding* Skill Graph *is a hypergraph $G = \langle V, E \rangle$ defined as follows. The agent form the vertex set, i.e. $V = I$, and the set of skills forms the hyperedge set. That is, each skill s defines a single hyperedge $e_s \in E$ that connects all vertices corresponding to the agents owning the skill s. We denote by $d = max_{s \in S}|e_s|$ the largest number of agents sharing a single skill.*

A simple case: STSG We first consider STSGs, and further assume that for each skill, there are at most two agents having that skill (i.e. $d = 2$). We show that a simple algorithm would work in this case.

Theorem 4. *Given a STSG with $d = 2$, the maximal value of a coalition structure, $\max_{CS \in \mathcal{CS}(I)} v(CS)$, is 2 if and only if G is a bipartite graph and 1 otherwise. Further, if G is bipartite then the bi-partition of the vertices of G forms the optimal coalition structure CS^*. If G contains singletons or is not bipartite, then $CS^* = I$.*

Proof. If G contains singletons, then only the grand coalition can do the task and $v(CS^*) = v(I) = 1$. Otherwise, all hyperedges connect exactly two vertices (and thus are edges), so G is a graph. Since each skill is owned by exactly two agents, for any coalitional structure, at most two coalitions in it can cover all the skills. Thus, to find the optimal coalitional structure, CS^*, we only need to consider structures with two coalitions, thus $v(CS) \in \{1, 2\}$.

Each of the two coalitions covers all skills iff each edge of G connects two agents from the two different coalitions, since then each side of the partition has an agent who has the skill that edge represents. This occurs when G is a bipartite, and the two coalitions form the vertex bi-partition. □

Bounded tree-width graph The tree-width of a hypergraph G generalizes of the tree-width of graphs. The following definitions are from (Gottlob, Leone, and Scarcello 2002), and we refer the reader there for a detailed introduction. We use $V(G)$ to denote the vertex set of G, and $E(G)$ for the hyperedge set of G.

Definition 6. *Let $G = \langle V, E \rangle$ be a hypergraph. A* tree decomposition *of G is a tuple $(R, \mathbf{B})$, where T is a tree and $\mathbf{B} = (B_t)_{t \in V(R)}$ is a family of subsets of $V(G)$ (each such B_t is called a bag) such that: (a) for each $e \in E(G)$, there is a node $t \in V(R)$ such that $e \in B_t$; (b) for each $v \in V(G)$ the set $\{t \in V(R) | v \in B_t\}$ is connected in T.*

Definition 7. *Let $(R, \mathbf{B})$ be a tree decomposition of a hypergraph G. The* tree-width *of $(R, \mathbf{B})$ is $\max_{t \in V(R)} |B_t|$. The tree-width of G is the minimum tree-width of $(R, \mathbf{B})$ over all tree decompositions $(R, \mathbf{B})$ of G. We denote the tree-width of a hypergraph G by $w(G)$.*

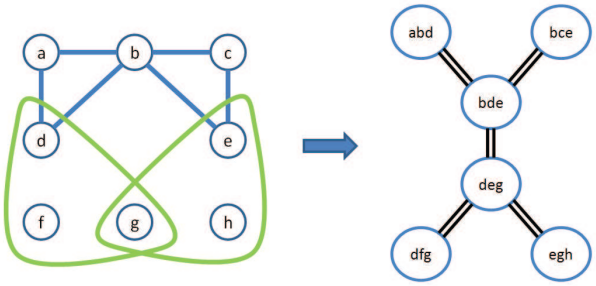

Figure 2: A tree decomposition of a hypergraph. The tree-width of the tree decomposition is 3.

Figure 2 illustrates an example of tree decomposition of a hypergraph, and the tree-width of the decomposition.

Let CSG(m, w) be the class of all CSGs with the restriction that the number of tasks is at most m, and the tree-width of the corresponding skill graph is at most w. Note that when the number of tasks $|T|$ is bounded by a constant, the value function $u : 2^T \to \mathbb{Q}$ can also be described by a fixed size table. As before, we denote the number of agents by n, and the number of skills by k. Thus the description length of any instance of CSG(m, w) is $O(nk)$. We now show a *fixed parameter tractable* (FPT) algorithm[3] that computes the optimal coalitional structure for CSG(m, w), using the tree decomposition of Definition 6.

Theorem 5. *CS-OPT-CSG(m, w) is in $\mathcal{P}$. Further, there is a (m, w)-FPT algorithm that checks for a given CSG $\langle I, v \rangle$ whether it belongs to the restricted set CSG(m, w) and if so, returns the optimal coalition structure for CSG.*

Proof. Consider checking if a coalition C achieves a given task subset $T_0 \subseteq T$, i.e. whether $T_0 \subseteq T(C)$. By the definition of CSGs, C achieves T_0 iff for any skill s required by at least one task in T_0, at least one agent in C owns s.

Denote the maximal number of agents sharing a skill as $d' = \max_{s \in S} |\{a_i \in I | s \in S_i\}|$, and $d = d' \cdot |T|$. A coalition structure can achieve at most d tasks, as a task that requires skill a s which only x agents share can be achieved at most x times.

A *candidate task solution* is a set $\mathbf{T} = \{T_i\}_{i=1}^{d}$, where each $T_i \subset T$ is a subset of tasks. For any coalitional structure $CS = \{C_i\}_{i=1}^{d}$, we say CS *achieves* $\mathbf{T}$ if for all $1 \leq i \leq d$, C_i achieves T_i. If CS achieves $\mathbf{T}$ then the following holds: $v(CS) \geq \sum_{i=1}^{d} u(T_i)$ (C_i achieves T_i, but could also achieve *more* tasks). Further, for the optimal

[3] Fixed parameter tractability with parameter α (α-FPT) means the running time is $f(\alpha) \cdot p(n)$ where $f(\alpha)$ may be any function (e.g. an exponential function), and $p(n)$ is a polynomial function in the input length n. In our case, the parameters are the number of tasks and the tree-width. In contrast, an algorithm that runs in time $p(n)^{f(\alpha)}$ is still polynomial when the parameter α is bounded, but is not α-FPT (and is probably intractable in practice).

structure CS^*, there is a set of task subsets $\mathbf{T}^* = \{T_i^*\}_{i=1}^d$ for which the inequality is an equality. This occurs when $T_i^* = T(C_i^*)$ for all $C_i^* \in CS^*$. We denote the set of candidate task solutions that are achievable by some coalition structure of size d as $\mathcal{T}(d)$. The value of the optimal coalition structure is the maximal value of an achievable candidate task solution $v(CS^*) = \max_{\mathbf{T} \in \mathcal{T}(d)} \sum_{i=1}^d u(T_i)$.

Similarly, if for each fixed candidate task solution $\mathbf{T}$ we compute a CS that achieves it (or find that one does not exist), then we can compute CS^*.

We first show that for a fixed candidate task solution $\mathbf{T}$, we can efficiently verify whether it can be achieved (i.e. if $\mathbf{T} \in \mathcal{T}(d)$). A key observation is that CS achieves $\mathbf{T}$ iff the following holds for *each skill* s:

(a) For each $1 \leq i \leq d$, if s is required for at least one task in T_i, then at least one agent in C_i owns s.

We reformulate the problem as a constraint satisfaction problem (CSP). Consider our skill-hypergraph G, where each vertex is an agent. A coalition structure is a *coloring* of V in d colors, where all agents with the same color form a coalition. The constraints are represented by the hyperedges in G, where (a) is enforced separately for each skill (hyperedge). Since the size of each such hyperedge is at most d, a naïve representation of the domain requires d^d possible assignments to the vertices of the hyperedge e_s. We can check each such partial assignment in $d^2 \cdot k$ operations (if for every T_i that requires s there is at least one vertex with color i, we keep the assignment, and otherwise we remove it). Thus it takes $O(d^{d+2} \cdot k)$ to compute the domain of a single constraint, and $O(d^{d+2} \cdot k^2)$ to compute all the domains [4]

This only *defines* the CSP – we still need to solve it. Since the hypergraph G underlying our CSP has a bounded tree-width w, there are known algorithms that solve it efficiently. We briefly describe one method, which allows us to reuse many computations. We first take G as input, check if its tree-width is bounded by w, and if so return a tree decomposition $(R, \mathbf{B})$ of G (see (Gottlob, Leone, and Scarcello 2002; Bodlaender 1993)). The algorithm is w-FPT and runs in $f(w) \cdot p(n,k)$ time (for an exponential f and some low degree polynomial p). The *dual* problem of our original CSP is also a CSP, whose underlying graph is the tree T. The new problem has two types of constraints: the ones induced by (a) are enforced separately on each bag (see in figure 2 how every hyperedge in G is embedded in a bag in R). Also, each edge of T enforces consistency, i.e. prevents an agent from getting different colors in two neighboring bags (due to the connectivity property of $(R, \mathbf{B})$, the agent must get the same color in *all* buckets). Rhe underlying graph of the new CSP is a tree, and can be solved in $O(n \cdot k \cdot (d^d)^2)$ (linear in the size of the tree and quadratic in the size of the bags' domains. See e.g. (Russell and Norvig 2003).). A solution of either the primal or the dual CSP is a valid coloring of the vertices of G, that is, a coalition structure of size d that achieves $\mathbf{T}$. If there is no solution, we know that $\mathbf{T} \notin \mathcal{T}(d)$.

[4] The concrete CSP constrains are simple restrictions of variable domains, or constraints regarding neighbouring bags in the tree decomposition.

To find the optimal coalition structure over all of $\mathcal{CS}(I)$, we iterate over candidate task solutions $\mathbf{T} \in \mathcal{T}(d)$, and pick the one that maximizes the utility and return its corresponding coalition structure CS^*. There are $2^{m \cdot d}$ possible $\mathbf{T}$'s, so the complexity of naïve loop would be:

$$O(2^{md}(f(w)p(n,k) + n \cdot k^2 \cdot d^{2d})) = f'(w,m,d) \cdot p'(n,k) \ .$$

Note that this is a FPT algorithm for the parameters m, w (as $d \leq w$). However, we can do better. since in all the iterations we essentially use the same skill graph, the decomposition can be performed once. The constraints will then be enforced directly on the dual for each $\mathbf{T}$, as we explained.

If we reuse the dual as suggested, the complexity is significantly reduced to $f(w)p(n,k) + O\left(2^{md} \cdot n \cdot k^2 \cdot d^{2d}\right)$. □

The proposed algorithm relies on the tree decomposition and therefore on the low tree-width of G. However, even if we cannot bound w, we may still try other algorithms designed for constraint satisfaction. These can be applied on the primal or dual CSP based on domain specific knowledge, graph properties other than tree-width, or heuristics.

Related Work

A key factor in designing algorithms for cooperative multi-agent systems is representing synergies and cooperation. Indeed, several papers describe representations based on combinatorial structures such as CSGs. For examples see (Deng and Papadimitriou 1994; Ieong and Shoham 2005) and a survey in (Bilbao 2000). In particular, generation of the optimal coalition structure and its applications, such as vehicle routing (Sandhlom and Lesser 1997) and multi-sensor networks (Dang et al. 2006) received much attention.

An early approach by (Shehory and Kraus 1998), focused on overlapping coalitions and gave a loose approximation algorithm. Another early approach (Sandholm et al. 1999) has a worst case complexity of $O(n^n)$, whereas dynamic programming approaches have a worst case guarantee of $O(3^n)$. Such algorithms were examined empirically in (Larson and Sandholm 2000).

Arguably, the current state of the art method is presented in (Rahwan and Jennings 2008). They suggest an algorithm with a worst case runtime of $O(n^n)$ and does not provide polynomial guarantees even for restricted classes, but in practice it is faster than the above mentioned methods. These approaches assume a black-box that computes the value of a coalition, whereas we rely on a specific representation. The approach of (Dang et al. 2006), which suggests a polynomial algorithm for a restricted case of multi-sensor networks, is more similar to our method as it is also a task based model. However, their task model is very different from CSGs. Other work of similar nature include (Elkind, Chalkiadakis, and Jennings 2008), which handles Weighted Voting Games (WVGs), and (Ohta et al. 2009) which uses CSP techniques for Synergy Coalition Groups, MC-nets and Multi-Issue Domains. However, WVG are an extremely limited representation, unlike CSGs which can represent any game, and the results in (Ohta et al. 2009) are hardness results (although the problems are reduced to Mixed Integer Programs). In contrast to the above approaches, our paper

provides both a *general* algorithm and a sufficient condition that guarantees a *polynomial* running time.

Conclusion

We considered computing the optimal coalitional structure in CSGs. We showed that this problem is $\mathcal{NP}$-hard, even in very restricted cases, but that it can be solved efficiently when the skill hypergraph exhibits certain topological properties. Specifically, the problem is fixed parameter tractable in the tree-width of the skill graph and the number of tasks.

CSGs are a general representation of coalitional games, therefore our algorithm can be viewed as a general algorithm for coalitional structure generation. However, since the CSG representation of a game may be extremely long, our algorithm is tractable only in domains which have a short CSG representation, with relatively few tasks and a low tree-width skill graph. We believe this is likely to be the case for many domains, especially domains such as the examples given in the beginning of the paper, where the tree-width is limited due to geographical constraints. In addition, the CSP formulation of the problem can be used by other algorithms that may work better even when the tree-width is high.

Several directions remain open for future research. First, in cases where finding an optimal solution is intractable, we should design algorithms that find an approximately optimal solution based on structural or other properties of the CSG.

Second, while our motivation comes from coalitional game theory, we only addressed social welfare, and ignored the question of coalition stability. When agents are selfish, and only care about their own utility, further game theoretic analysis is appropriate. It would be interesting to examine questions relating to solution concepts such as the core, the Shapley value and nucleolus. It is known for example, that according to some notions of approximated core, the coalition structure that is the easiest to stabilize is the one that maximizes the social welfare (Bachrach et al. 2009).

Finally, could we find new structural properties that allow finding the optimal structure efficiently? Could we find optimal coalition structures for other representation languages, such as (Bilbao 2000; Ieong and Shoham 2005; Bachrach and Rosenschein 2007). Although there are relatively few works considering coalition structures in restricted forms of coalitional games, we believe many problems that arise in practice could be solved using techniques tailored to these restricted cases. Algorithms such as the one proposed in this paper are a step in this direction. Also, the approximation quality of many algorithms could be improved when dealing with restricted game classes.

Acknowledgement

Kyomin Jung was supported by the Engineering Research Center of Excellence Program of Korea Ministry of Education, Science and Technology(MEST) / National Research Foundation of Korea(NRF) (Grant 2009-0063242).

References

Aumann, R., and Dreze, J. 1974. Cooperative games with coalition structures. *International Journal of Game Theory* 3(4):217–237.

Bachrach, Y., and Rosenschein, J. S. 2007. Computing the Banzhaf power index in network flow games. In *AAMAS 2007*, 323–329.

Bachrach, Y., and Rosenschein, J. S. 2008. Coalitional skill games. In *AAMAS 2008*, 1023–1030.

Bachrach, Y.; Elkind, E.; Meir, R.; Pasechnik, D.; Zuckerman, M.; Rothe, J.; and Rosenschein, J. S. 2009. The cost of stability in coalitional games. In *SAGT 2009*, 122–134.

Bilbao, J. M. 2000. *Cooperative Games on Combinatorial Structures*. Kluwer Publishers.

Bodlaender, H. 1993. A linear time algorithm for finding tree-decompositions of small treewidth. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, 226–234. ACM New York, NY, USA.

Dang, V.; Dash, R.; Rogers, A.; and Jennings, N. 2006. Overlapping coalition formation for efficient data fusion in multi-sensor networks. In *AAAI-2006*, volume 21, 635.

Deng, X., and Papadimitriou, C. H. 1994. On the complexity of cooperative solution concepts. *Math. Oper. Res.* 19(2):257–266.

Elkind, E.; Chalkiadakis, G.; and Jennings, N. 2008. Coalition structures in weighted voting games. In *ECAI-2008*, 393.

Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company.

Gottlob, G.; Leone, N.; and Scarcello, F. 2002. Hypertree decompositions and tractable queries. *Journal of Computer and System Sciences* 64(3):579–627.

Ieong, S., and Shoham, Y. 2005. Marginal contribution nets: A compact representation scheme for coalitional games. In *Proceedings of the 6th ACM Conference on Electronic Commerce*, 193–202.

Larson, K., and Sandholm, T. 2000. Anytime coalition structure generation: average case study. *J. Experimental & Theoretical Artificial Intelligence* 12(1):23–42.

Ohta, N.; Conitzer, V.; Ichimura, R.; Sakurai, Y.; Iwasaki, A.; and Yokoo, M. 2009. Coalition Structure Generation Utilizing Compact Characteristic Function Representations. In *CP-09*. Springer.

Peleg, B., and Sudholter, P. 2007. *Introduction to the theory of cooperative games*. Springer.

Rahwan, T., and Jennings, N. 2008. An improved dynamic programming algorithm for coalition structure generation. In *AAMAS-2008*, 1417–1420.

Russell, S., and Norvig. 2003. *Artificial intelligence: a modern approach*. Prentice hall, Upper Saddle River, N.J.

Sandhlom, T., and Lesser, V. 1997. Coalitions among computationally bounded agents. *Artificial Intelligence* 94(1-2):99–137.

Sandholm, T.; Larson, K.; Andersson, M.; Shehory, O.; and Tohmé, F. 1999. Coalition structure generation with worst case guarantees. *Artificial Intelligence* 111(1):209–238.

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