Feasible Coalition Sequences

Tabajara Krausburg
School of Technology, PUCRS
Porto Alegre, Brazil
Department of Informatics, TUC
Clausthal, Germany
tabajara.rodrigues@edu.pucrs.br

Jürgen Dix
Department of Informatics, TUC
Clausthal, Germany
dix@tu-clausthal.de

Rafael H. Bordini
School of Technology, PUCRS
Porto Alegre, Brazil
rafael.bordini@pucrs.br

ABSTRACT

We introduce the idea of a finite *sequence of coalition formation games* over a set of agents, and we call it Sequential Characteristic-Function Game (SCFG). We define the solution of such a game as a corresponding sequence of coalition structures that must be related by a given *feasibility relation*, so no coalition structure can be evaluated in isolation. A sequence satisfying this condition is called Feasible Coalition-Structure Sequence (FCSS). Such games can be a useful abstraction for modelling various scenarios, in particular those for real-world disaster management that we consider in this paper. We give an algorithm for computing an FCSS and evaluate it experimentally. Our results show that an SCFG can represent various classical variations of characteristic-function games, and our algorithm solves instances with a reasonable number of agents.

KEYWORDS

Coalition Structure Generation; Interdependence of Coalitional Games; Practical Applications of Games; Disaster Response

ACM Reference Format:
Tabajara Krausburg, Jürgen Dix, and Rafael H. Bordini. 2021. Feasible Coalition Sequences. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 9 pages.

1 INTRODUCTION

Coalition formation has long been an interesting topic of research, either for its practical applications or complexity issues. The classical coalition formation process is formalised by a Characteristic Function Game (CFG) and one important part of it is the generation of coalition structures: given a set of agents $A = \{1, \ldots, n\}$ as input, we aim to partition it in the best possible way and obtain a Coalition Structure (CS) CS: $A = \bigcup CS$ and $C, C' \in CS$ implies $C \cap C' = \emptyset$, C is called a coalition of agents. The optimisation problem is then to maximise $V(CS) = \sum_{C \in CS} v(C)$ (where $v : 2^A \to \mathbb{R}$ is the characteristic function of the game) [14]. Previous work in the literature has shown that the coalition structure generation problem is $F\Delta_2^P$-complete [6]. Exact algorithms (e.g., [11]) were proposed to solve this problem, but due to its complexity, heuristic quality-bounded algorithms are needed for larger problem instances (e.g., [2]), and for real-world scenarios (e.g., [1]).

However, not all problems can be efficiently solved using a single coalition structure. Consider for instance the Incident Command System (ICS) [8], a popular system for disaster management. A part of this system establishes a modular hierarchy that adjusts itself according to the demands of a given disaster response operation. For instance, just one aspect of it requires resources (e.g., experts and equipment) to be distributed to a *group hierarchy* (we detail this process later). A group hierarchy may be modelled by a sequence of CFGs, one per level. However, picking the optimal solution of each level may not lead to a feasible overall solution: optimal CSs may not be compatible with one another.

In this paper, we propose a novel game named Sequential Characteristic-Function Game (SCFG), and a solution for such a game is a Feasible Coalition-Structure Sequence (FCSS). It extends the ideas of our previous work [9] to generalise the relationships between subsequent coalition structures in a sequence. We are particularly interested in the *coalition structure generation* for such games. An SCFG $\mathcal{G}$ takes as input a sequence of CFGs and, based on a binary relation on coalition structures, has as solution an ordered (according to the input) FCSS $\boldsymbol{CS}$. We show that our game can be used to model a variety of extended CFGs, like constrained and task-based CFGs. Further, we provide a heuristic algorithm, named MC-Link, to solve SCFG instances. It is inspired by C-Link [2], a near-optimal algorithm for coalition structure generation problems in CFGs. We show that our algorithm can be applied to instances containing a reasonable number of agents (up to 100 in the experiments reported here).

This paper is structured as follows. In Section 2, we introduce a simple example to illustrate the need for our approach. Moreover, we formally define and analyse our framework for SCFG. Section 3 introduces a disaster response management problem and we discuss how to define an SCFG for such a complex domain. In Section 4, we propose an algorithm for solving SCFG instances. We evaluate this algorithm in Section 5 under a relation inspired by a real-world application. In Section 6, we discuss how SCFG compares to other coalition-formation games and other similar problems. Finally, we conclude our work and point out future directions in Section 7.

2 SOLUTIONS FOR SEQUENTIAL GAMES

The games we introduce here allow us to consider coalition structures that are somehow related to each other (see the example below) and therefore should not be evaluated in isolation.

2.1 Motivation and Main Notions

Here is a scenario to motivate our work. Andy (a), Bobby (b), and Carol (c) are planning to go shopping. To do so, they need to travel to a store and then buy their items. The first game models their preferences for travelling, where different coalitions travel to different stores. $v_1(\{a\}) = v_1(\{b\}) = v_1(\{c\}) = 1, v_1(\{a, b\}) = v_1(\{b, c\}) = 2,$

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.
© 2021 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

and $v_1(\{a,c\}) = v_1(\{a,b,c\}) = 4$. So Andy enjoys travelling with Carol, and even larger coalitions are also preferred because they will lower the costs. The second game models the shopping itself. Andy is hoping for Bobby's or Carol's help to decide the best item to buy, although Bobby knows much more about those items than Carol. However, if both Bobby and Carol are together, they only talk about life and are of no help to Andy. Thus, $v_2(\{a\}) = v_2(\{b,c\}) = v_2(\{a,b,c\}) = 0$, $v_2(\{b\}) = v_2(\{c\}) = v_2(\{a,c\}) = 1$, and $v_2(\{a,b\}) = 3$.

The optimal CS CS in game 1 (namely $\{\{a,c\},\{b\}\}$ with a value of 5) is not optimal in game 2 (where the optimal is $\{\{a,b\},\{c\}\}$, with a value of 4). If we take $\{\{a,c\},\{b\}\}$ (optimal in game 1) for both games, we get 5 in game 1 but 2 in game 2, so a total of 7. Similarly, if we consider the optimal structure in game 2 ($\{\{a,b\},\{c\}\}$) and pick it for both games we get also a total of 7. We may choose different structures for the two games and take the optimal in each round: $v_1(\{\{a,c\},\{b\}\}) + v_2(\{\{a,b\},\{c\}\}) = 9$. Both coalition structures are optimal as there is no other with a higher value in each game separately. It means that Andy and Carol travel together to the same store, and Bobby goes to another one (first round). In the second round, Andy and Bobby are supposed to buy articles together, but this is no longer *feasible* since they are at different stores.

To solve the problem, we model the interdependence between the games with a binary relation $\mathcal{R}$ on the set of all coalition structures $\boldsymbol{CS}^A$ to state that the agents buying items together must have travelled to the same place. This is important when there are restrictions on the coalition formation so that the sequence of individually optimal structures is not feasible. In the example, the optimal FCSS $\boldsymbol{CS}^*$ has a value of $v_1(\{\{a,b,c\}\}) + v_2(\{\{a,b\},\{c\}\}) = 8$; travelling together to the same store but buying at different departments.

We now define sequences of games as a formalisation for scenarios of the type described above, where the set of agents is $\{a,b,c\}$, we have two characteristic function games ordered as Γ_1 and then Γ_2, defined by their characteristic functions v_1 and v_2. In that case, we have a sequence of length two. The outcome of that game should be a pair of coalition structures $\langle CS_1, CS_2\rangle$ that respects the relation $\mathcal{R}$ and results in the highest value of $v_1(CS_1) + v_2(CS_2)$.

Definition 2.1 (SCFG, FCSS). A Sequential Characteristic-Function Game (SCFG) $\mathcal{G}$ is a tuple $\langle A, \mathcal{H}, \mathcal{R}\rangle$ where:

- A is a set of agents $A = \{a_1, \ldots, a_n\}$;
- $\mathcal{H}$ is a totally ordered set of CFGs $\Gamma_i = \langle A, v_i\rangle$, $1 \le i \le h$ (we use h to denote the length of the game sequence);
- $\mathcal{R}$ is a binary relation on $\boldsymbol{CS}^A$ (all coalition structures over A).

A *solution* for an SCFG $\mathcal{G}$ is a sequence $\boldsymbol{CS} = \langle CS_1, \ldots, CS_h\rangle$ of coalition structures respecting relation $\mathcal{R}$: $CS_i \mathcal{R} CS_{i+1}$, $1 \le i < h$. We call this condition *feasibility*. We define the value $\mathcal{V}$ of $\langle CS_1, \ldots, CS_h\rangle$ as $\sum_{i=1}^{h} V_i(CS_i)$, where $V_i(CS_i) = \sum_{C\in CS_i} v_i(C)$. We call such a sequence a Feasible Coalition-Structure Sequence (FCSS). The goal is to find an optimal FCSS $\boldsymbol{CS}^*$: $\boldsymbol{CS}^* = \operatorname{argmax}_{\boldsymbol{CS}}\ \mathcal{V}(\boldsymbol{CS})$.

The main point is that we cannot just take the optimal CS of each game (see the previous example); we need to make sure that the sequence of CSs is compatible with $\mathcal{R}$, which is specific to each particular application. For example, it may be the case that once a coalition structure is formed in the first round, only subgroups of those coalitions can be formed for the next coalition structures (leading to a form of group hierarchy). If a sequence of coalition structures is not feasible, we omit the F and call it CSS.

2.2 SCFG and Coalition Formation Problems

Our framework is sufficiently general to express various other coalition formation problems, including discrete overlapping coalitions [18], combinatorial auctions [10], constrained coalition formation games, and task-based CFGs. We show the latter two results more formally below. In [13], a Constrained Coalition Formation Game (CCFG) is defined as a tuple $\langle A, \boldsymbol{CS}_{cst}, v\rangle$ where $A = \{a_1, \ldots, a_n\}$ is the set of agents; $\boldsymbol{CS}_{cst} \subseteq \boldsymbol{CS}^A$ is the set of feasible coalition structures; and v assigns a real value to each coalition. An optimal coalition structure is one with the highest value.

The other example is the Task-Based Characteristic Function Game [15] (TCFG), formally defined as $\langle A, T, v\rangle$. We are given a set of tasks T, and the characteristic function v assigns a value to each coalition and task (one coalition works on one task): $v : 2^A \times T \to \mathbb{R}$. An optimal CS CS^* is one where $\sum_{C\in CS} v(C, t_C)$ is maximal, where $t_C \in T$ are different tasks in T: $t_C \ne t_{C'}$ for $C \ne C'$.

Theorem 2.2. *Both constrained coalition formation games CCFG (as defined in [13]) as well as task-based characteristic function games TCFG (as defined in [15]) are special cases of SCFGs.*

All coalition structures CS for a CCFG Γ_C (resp. for a TCFG Γ_T) are in one-to-one correspondence to the feasible sequence of coalition structures $\boldsymbol{CS}$ for a corresponding SCFG (as constructed below) and the values are identical.

Proof. Let $\langle A, \boldsymbol{CS}_{cst}, v\rangle$ be a CCFG with h coalition structures $\boldsymbol{CS}_{cst} = \{CS_1, \ldots, CS_h\}$. We construct an SCFG $\mathcal{G}$ with h copies of the ordinary characteristic function game $\langle A, \frac{1}{h}v\rangle$ where all coalition structures not in $\boldsymbol{CS}_{cst}$ get a value $-\infty$. So we get Γ_i with $1 \le i \le h$ where $v_i(C) := \frac{v(C)}{h}$. This is our set $\mathcal{H}$. The relation $\mathcal{R}$ is defined by $CS\ \mathcal{R}\ CS'$ iff $CS, CS' \in \boldsymbol{CS}_{cst}$. So we essentially repeat the same game h times. The modified v ensures the right value.

Consider now a TCFG $\langle A, T, v\rangle$; it suffices to construct an equivalent CCFG $\Gamma_C = \langle A', \boldsymbol{CS}'_{cst}, v'\rangle$. Let $A' := A \cup T$ and let the feasible coalition structure $\boldsymbol{CS}'_{cst}$ consist of all coalition structures CS for A' that satisfy: (1) for all $C \in CS : |C \cap T| = 1$, and (2) $C \setminus T \ne \emptyset$. Thus, the feasible coalition structures for A' are exactly the coalition structures of the original A. This allows us to define $v' : A' \to \mathbb{R}$ by $v'(C') := v(C, t)$ where $C' = C \cup \{t\}$ (which is well defined because of our definition of feasible coalition structures in $\mathcal{R}$). Clearly, the optimal $\boldsymbol{CS}^*$ of an SCFG for Γ_C corresponds to the optimal CS^* for TCFG and has the same value. □

Clearly, the opposite direction is not true: SCFGs are more general than both CCFG and TCFG.

2.3 The Role of $\mathcal{R}$ in a Sequential Game

In the general form of the game, the relation $\mathcal{R}$ can be any binary relation. We introduce below three interesting relations that can be used to define particular types of SCFGs and will be used in the remainder of this work.

Definition 2.3 (Relations $\mathcal{R}_H$, $\mathcal{R}_S$, $\mathcal{R}_O$). For any subsequent $CS, CS' \in \boldsymbol{CS}$, we define the following three relations:

$CS\,\mathcal{R}_H\,CS'$**:** for all $a \in A, C \in CS, C' \in CS'$, s.t. $a \in C$, it holds that if $a \in C'$, then $C \cap C' = \{a\}$;

$CS\,\mathcal{R}_S\,CS'$**:** for all $C' \in CS'$ there is $C \in CS$ such that $C' \subseteq C$ and $|CS| < |CS'|$;

$CS\,\mathcal{R}_O\,CS'$**:** there is $C \in CS$ s.t. all other coalitions $C' \in CS$, $C' \neq C$, are also contained in CS', and CS' contains two additional coalitions C_1, C_2 with $C_1 \cup C_2 = C$.

The relation $\mathcal{R}_H$ states that agents work on different coalitions in which at most one member can be repeated. In other words, the agents will always work alone or with different agents in a subsequent coalition structure. $\mathcal{R}_S$ establishes the subset relation between coalitions in different levels and is useful in domains that require a *group hierarchy* between agents. Finally, the relation $\mathcal{R}_O$ states that in one step all but one coalition C remain, and C is split into exactly two coalitions.

2.4 A Training Session Application

Having defined an SCFG, we now demonstrate how it can be used to address an interesting problem. Assume a company is starting a new branch for a new product. A set of agents will work on the development of this product and to make the agents familiar to one another, the company proposes a training session. The session is divided into three interconnected projects in which a coalition structure is formed for each one of them. The first two projects address participants' technical skills, hence they use the same characteristic function. The last project addresses general skills and has a specific valuation.

When participating in a project, an agent eventually develops an acquaintanceship with all of its coalition partners. This is represented by a weighted undirected graph $G = (A, E)$ where A is the set of agents, and $E \subseteq A \times A$ is a set of edges. Each agent will build its own graph, adding new edges after all coalitions have been disbanded. Each edge is given a weight $w : E \longrightarrow \mathbb{N}$, so that $w(e_{ij})$ represents how well agents $i, j \in A$ worked together.

To maximise the chances of the agents establishing good relationships (i.e., maximising w), the company formulates the training session as an SCFG $\mathcal{G} = \langle A, \mathcal{H}, \mathcal{R}_H \rangle$, where:

A is the set of agents participating in the training session;
$\mathcal{H}$ is the ordered set of games (projects) with length $h = 3$;
$\mathcal{R}_H$ is as in Definition 2.3.

The company could choose the characteristic functions from a range of options. For instance, a function that determines that better coalitions are those with heterogeneous members, and larger coalitions are preferable. The relation $\mathcal{R}_H$ will ensure new edges are added to the agents' interaction graphs in the two technical projects. In the last project, the agents will not be aware of what others, in the same coalition, have done in the preceding (technical) project, although they might meet again agents they collaborated with in the first project. This way, the collaboration between them will be stimulated in the general skill session. It should be noted that $\mathcal{R}_H$ copes with subsequent CSs only. With $\mathcal{R}$ defined as a binary relation on the set of coalition structures we cannot express, for example, that two agents must not work together more than once throughout the whole sequence. Formally, given *any* $CS, CS' \in \mathbf{CS}$, we require for all $a \in A$, $C \in CS$, $C' \in CS'$: if $a \in C$ and $a \in C'$, then $C \cap C' = \{a\}$. We aim to address this in future work.

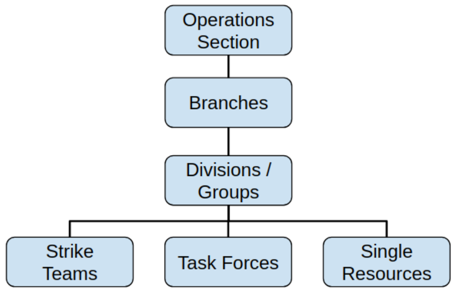


Figure 1: An illustration of the organisational structure of the Operations Section; adapted from [8].

3 A GAME FOR DISASTER RESPONSE

We now demonstrate the modelling of a real-world scenario as an SCFG. For this, we use a disaster response problem in which coalitions must respond to a disaster event. First, we introduce the ICS [8], a management system designed for disaster response operations. We then define a simplified instance of the problem of distributing agents in an ICS specification.

3.1 The Incident Command System

The ICS was initially developed to deal with a series of wildfire events occurring in southern California in 1970. Since then, it has become so popular that it is now adopted and used by FEMA (in the US) to respond to disaster events [4]. For our work, two main ICS characteristics are relevant:

Span of Control: this number is the ratio of command between supervisors and subordinate units acting upon the event: $\frac{1}{\lambda}$. For instance, ratio $\frac{1}{3}$ indicates 1 supervisor is responsible for 3 subordinate units.

Modular Organisation: the organisational structure is adjusted according to the complexity of the current event following the span-of-control guidance. For instance, if the event occurs over a large area, then the span of control for a supervisor may be reduced (e.g., from five to two or three subordinate units per supervisor). This ratio is closely related to the characteristics of a disaster event.

For the entire disaster response, five main sections are defined. However, we focus on the *Operations Section*. It acts upon the damaged area of the disaster event, and its responsibilities include achieving command objectives, tactical operations, contingency planning, among others. We depict in Figure 1 its hierarchical organisation. Branches are divided into divisions/groups, in which divisions are allocated to geographical areas of the event (e.g., because of different jurisdictions) and groups are not fixed to a specific site. Instead, they focus on specific functionalities (e.g., rescue group). The number of branches, divisions, and groups depends on the span-of-control ratio, which can make the units more manageable by their supervisors. As the size and complexity of the event increases, groups/divisions are further partitioned into task forces, strike teams, or even single resources. A resource can be a person or an individual piece of equipment [8]. Heterogeneous resources constitute task forces, whilst strike teams are homogeneous.

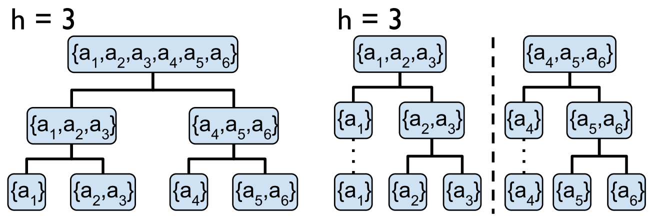


Figure 2: Hierarchies of same length but starting from different coalition structures.

3.2 SCFG Properties under $\mathcal{R}_S$

We are interested in the hierarchical structure of an ICS instance. When applying the relation $\mathcal{R}_S$ on an SCFG, the outcome will be a tree structure (see Figure 2). Using this hierarchical representation, the properties of SCFGs are as follows.

Sub-hierarchies: It is not always the case that the top level of the hierarchy (i.e., first game) will consist of the grand coalition (see Figure 2).

Shape of the Hierarchy: A coalition may either split itself or remain as it is in the next level of the hierarchy. This leads to well-known hierarchical patterns (e.g., balanced, unbalanced, or ragged) as discussed in the database literature [12].

Specialisation: As we move on to deeper levels of the hierarchy, the coalitions become further specialised.

3.3 The ICS as a Sequential Game

Theorem 3.1. *The Operations Section of the ICS above can be modelled by an SCFG $\mathcal{G}$.*

Proof. The proof is by using Theorem 2.2, so it suffices to model the ICS as a constrained coalition formation game.

As noted in Section 3.1, the set of agents appears through branches, division/groups, task forces, strike teams, and single resources defined according to the span of control. This structure forms a group hierarchy in which agents are allocated firstly to branches, then to divisions/groups, and finally to the entities at the last level. While these are only 3 levels, to model it as an SCFG we need more games in the sequence, as explained below.

One of the main characteristics in an ICS is the span of control $\frac{1}{\lambda}$ together with the number of branches b, each divided into divisions and groups, based on a set A of agents (see Figure 1). It must hold that $b \leq \lambda$, and each coalition can have size at most λ. These are the only restrictions, apart from the three levels, each of which contains a coalition structure.

It is, therefore, more appropriate to model this situation with an intermediate CCFG representation (which can then be represented as an SCFG). Before defining the SCFG, we introduce the intermediate CCFG representation. Given $\langle S_C, S_{CS} \rangle$, where $S_C \subseteq \{1, \ldots, |A|\}$ and $S_{CS} \subseteq \{1, \ldots, |A|\}$, we define $\langle A, \boldsymbol{CS}'_{cst}, v \rangle$ the associated constrained coalition formation game, where $\boldsymbol{CS}'_{cst}$ consists of all coalition structures of sizes in S_{CS} and coalitions C in them of sizes in S_C. No other coalition structures are feasible, the valuation is not constrained in any way.

For the simple hierarchy on the right-hand side of Figure 2, we need the following three associated CCFGs on the set of agents $A, |A| = 6$: Γ_{C1} determined by $\langle \{1, 2, 3, 4, 5, 6\}, \{2\} \rangle$, Γ_{C2} determined by $\langle \{1, 2, 3, 4, 5, 6\}, \{4\} \rangle$, and Γ_{C3} determined by $\langle \{1\}, \{6\} \rangle$. In the general case, given any group hierarchy defined by the ICS, it is clear that a sequence of CCFGs (as illustrated in the example above) can model the ICS (the limiting sizes S_C, S_{CS} are determined by λ and b). Of course, we also need to define a characteristic function for each game. This would be done by the experts responsible for the operation.

While Theorem 2.2 only states that one single CCFG can be expressed as an SCFG, the same construction applies to several CCFGs, with the relation $\mathcal{R}$ appropriately adapted. □

SCFG is a flexible framework that makes it possible to model applications in various ways. Above, we have used our framework to model the ICS as a sequence of CCFGs. One can also model it by adding in A a set of "distinguished agents": each agent represents a particular level in the hierarchy. A clever choice of $\mathcal{R}$ would limit the search space to exactly the CSs covered by any sequence of CCFGs. Then, we distinguish between CSs that belong to particular levels using the distinguished agents. We aim to show how to model the ICS in this particular way in future work.

There are many possibilities for representing domain knowledge within an SCFG to reduce the overall search space in the construction of an FCSS $\boldsymbol{CS}$. However, during the modelling of the ICS, we ignored several important aspects for forming coalitions in such a complex domain. For instance, the leader responsible for a group/-division could play an important role in the response operation. Many such aspects can be modelled by choosing the valuation function v appropriately. Alternatively, the games chosen to be in $\mathcal{H}$ can provide the features for describing the best way to place the agents in the group hierarchy. For instance, the concept of pivotal agents in valuation structures [6] can be investigated to represent leaders in this hierarchical problem.

3.4 A Second Disaster Response Example

As a second example in disaster response operations, consider a wildfire event. A fire brigade is called to act upon four big fires that are approaching a town. The entire fire brigade has enough experts and resources to fight off nine fires simultaneously. To be ready for any new fire event, the brigade supervisor decides to take all the resources and experts to the field. The brigade will be divided initially into four coalitions, and as soon as a new fire appears, one of these coalitions is split up to fight it off. This scenario can also be modelled as an SCFG.

As we are interested in forming a new CS as new fires are detected, we pick the relation $\mathcal{R}_O$ for this setting (see Definition 2.3). The sequence of games has length $h = 6$ to match the number of new fire events the brigade will be able to fight off (four initial ones plus up to five new episodes to reach 9 simultaneous fires). The first game is a CFG constraining the coalition structure size to exactly four. The remaining five games are CFGs without constraints. All six games receive as input the same characteristic function.

The outcome of this game will be a six-length sequence of coalition structures $\boldsymbol{CS}$. As relation $\mathcal{R}_O$ ensures that from the initial coalition structure only one coalition is allowed to split. As soon as a new fire requires effort to be extinguished, the fire brigade advances to a new coalition structure in $\boldsymbol{CS}$. The supervisor allocates all resources to $CS_1 \in \boldsymbol{CS}$ in advance. This way, the resources to be assigned to the next fire event to occur (i.e., the ones that will form a new coalition) are already positioned together.

4 AN ALGORITHM TO COMPUTE AN FCSS

Having explained the general idea of our approach, we now consider a heuristic algorithm that solves an SCFG instance. We take as inspiration the clustering-based algorithm named C-Link [2], and construct a new algorithm called MC-Link. We briefly explain C-Link before introducing our own algorithm[1].

4.1 The Multiple Coalition Linkage Algorithm

C-Link [2] is a heuristic algorithm (anytime under some conditions) with time complexity $O(n^3)$. It starts off from the coalition structure of singletons, and to produce a new coalition structure it evaluates all the moves (see Figure 3a) resulting of a merger of any two coalitions. It selects the move that produces the greatest gain, and moves cannot be undone. Gain is a clustering concept that, translated into the coalition formation problem, is described by Equation 1.

$$gain(C_i, C_j) = v(C_i \cup C_j) - v(C_i) - v(C_j) \tag{1}$$

It states that the gain of merging two coalitions is the difference between the coalition values with and without placing the agents in the same coalition. When all the moves are evaluated, the algorithm merges the two most suitable coalitions. This is determined through a *linkage function*; four such functions are proposed: (i) single-link; (ii) complete-link; (iii) average-link; and (iv) gain-link. Functions (i), (ii), and (iii) are based on pairwise relations between the agents that belong to the coalition, whilst (iv) considers the coalition itself and not individual members. For the remainder of this work, we consider only the gain-link (GL) defined in Equation 2, as it obtained the best results in the experiments reported in [2].

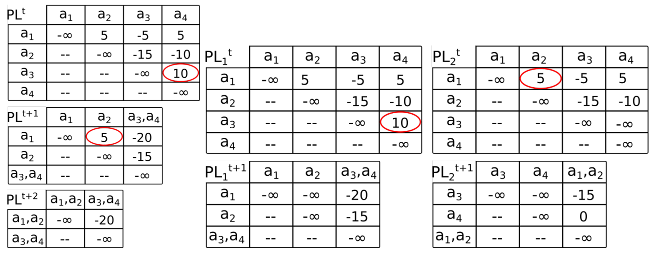
$$lf_{\text{GL}}(C_i, C_j) = gain(C_i, C_j) \tag{2}$$

The algorithm iteratively updates a partition linkage matrix PL (initially a $n \times n$ matrix) which is filled out with the value returned by the linkage function $lf(C_i, C_j)$ for entry (i, j) (note that the diagonal of the matrix is not relevant). The algorithm picks the argmax from a table PL and performs a merger. Once this is done, the matrix is updated with the new coalitions and their respective values from the linkage function. The algorithm stops when there is no advantage in merging any two coalitions—when the linkage function for all coalitions in the matrix has zero or a negative value.

The MC-Link algorithm follows the same general idea adopted for C-Link; it starts off with the coalition structure of singletons and merges two coalitions based on a function that measures the suitability of such a merger[2]. The underlying intuition is that each game Γ_i has a corresponding table PL_i. We perform the most suitable movement for a table PL_i (i.e., a merger), then advance to the next table PL_{i+1} and repeat the process (see Figure 3b). Doing so, we are able to construct the tables PL (one at time) whilst enforcing the relation $\mathcal{R}$, which is also given as input.

[1]https://github.com/smart-pucrs/SCFG

[2]A promising technique for SCFG is divisive clustering algorithms [16, Chapter 13]. Depending on the heuristic chosen to partition a cluster, it could lead to more efficient algorithms than the naive agglomerative approach, which is $O(n^3)$.



PL^t	a_1	a_2	a_3	a_4
a_1	$-\infty$	5	-5	5
a_2	--	$-\infty$	-15	-10
a_3	--	--	$-\infty$	10
a_4	--	--	--	$-\infty$

PL^{t+1}	a_1	a_2	a_3,a_4
a_1	$-\infty$	5	-20
a_2	--	$-\infty$	-15
a_3,a_4	--	--	$-\infty$

PL^{t+2}	a_1,a_2	a_3,a_4
a_1,a_2	$-\infty$	-20
a_3,a_4	--	$-\infty$

(a)

$PL_1{}^t$	a_1	a_2	a_3	a_4
a_1	$-\infty$	5	-5	5
a_2	--	$-\infty$	-15	-10
a_3	--	--	$-\infty$	10
a_4	--	--	--	$-\infty$

$PL_2{}^t$	a_1	a_2	a_3	a_4
a_1	$-\infty$	5	-5	5
a_2	--	$-\infty$	-15	-10
a_3	--	--	$-\infty$	$-\infty$
a_4	--	--	--	$-\infty$

$PL_1{}^{t+1}$	a_1	a_2	a_3,a_4
a_1	$-\infty$	$-\infty$	-20
a_2	--	$-\infty$	-15
a_3,a_4	--	--	$-\infty$

$PL_2{}^{t+1}$	a_3	a_4	a_1,a_2
a_3	$-\infty$	$-\infty$	-15
a_4	--	$-\infty$	0
a_1,a_2	--	--	$-\infty$

(b)

Figure 3: An example run with four agents for C-Link in 3a and for MC-Link in 3b. The red ovals represent the selected column and row to be merged, and t represents an iteration. MC-Link received as input $\mathcal{R}_H$, and a sequence of two CFGs; the games use the same characteristic function.

An algorithm for SCFG needs to take into account that:

(1) A CSS $\boldsymbol{CS}$ may not be feasible right from the beginning. In MC-Link, all CSs start off with singleton coalitions which may not be acceptable by $\mathcal{R}$ (e.g., $\mathcal{R}_S$). In fact, a CSS may become feasible only after several iterations of MC-Link.
(2) To calculate a value for an FCSS, we must consider the sequence as a whole. For instance, consider $\mathcal{R}_S$ and $\mathcal{R}_H$. $\mathcal{R}_S$ generates constraints being applied in only one direction (left to right). On the other hand, $\mathcal{R}_H$ leads to constraints being imposed in both directions.

Condition 1 indicates that we need to look for an FCSS even if that would mean choosing a non-suitable merger (i.e., choosing zero or negative values). Condition 2 indicates that the search is conducted in rounds, and at each round only *one merger per table* is carried out.

We show in Algorithm 1 the pseudocode for MC-Link. It starts by initialising a CSS $\boldsymbol{CS}$ with as many coalition structures of singletons as there are games in $\mathcal{H}$. Then, we start looking for a CSS that is feasible according to $\mathcal{R}$ (lines 3-16).

We check the feasibility condition using the Boolean function Feasible (e.g., line 3). It checks whether all pairs of subsequent CSs in a candidate solution are in $\mathcal{R}$. In case an index is provided (e.g., line 34), we check if the $\widehat{CS}$ at that index is feasible in relation to both the preceding and succeeding positions in the sequence.

In the beginning, if the sequence is not feasible, then we go through the tables PL merging two coalitions even if there is no suitable merger to carry out (e.g., all have a negative value). In this phase, a merger is considered impossible if it is constrained by relation $\mathcal{R}$ (i.e., the respective entry in table PL receives $-\infty$). In that case, we go back to the prior position in the sequence, construct the table, and repeat the procedure. If a merger is possible, then it receives the following value:

$$slf(v, C_i, C_j) = v(C_i \cup C_j) - v(C_i) - v(C_j) \tag{3}$$

This is necessary to pick the correct characteristic function according to the game we are evaluating at the moment. If the table for the first game has all of its moves constrained (i.e., equal to $-\infty$), that means MC-Link could not find an FCSS.

Algorithm 1 MCLINK

```
Input:
    $A$, a set of agents
    $\mathcal{H}$, a sequence of CFGs
    $\mathcal{R}$, a binary relation on $\boldsymbol{CS}^A$
Output: $\boldsymbol{CS}$, an FCSS
 1: $\boldsymbol{CS} \leftarrow \langle\{\{a_1\},\dots,\{a_n\}\}_1,\dots,\{\{a_1\},\dots,\{a_n\}\}_h\rangle$
 2: $\hat{M} \leftarrow \langle\infty_1,\dots,\infty_h\rangle$
 3: if $\neg$ Feasible($\boldsymbol{CS}$, $\mathcal{R}$) then
 4:     $i \leftarrow 1$
 5:     while $i \leq h-1$ do
 6:         $PL \leftarrow$ FillTable($i, \boldsymbol{CS}, v_i$)
 7:         $\hat{M}[i] \leftarrow \max_{j,k} PL(j,k)$
 8:         if $\hat{M}[i] > -\infty$ then                ▶ merger is available
 9:             $\hat{j},\hat{k} \leftarrow \text{argmax}_{j,k} PL(j,k)$
10:             $\boldsymbol{CS}[i] \leftarrow (\boldsymbol{CS}[i] \setminus \{C_{\hat{j}}\} \setminus \{C_{\hat{k}}\}) \cup \{C_{\hat{j}} \cup C_{\hat{k}}\}$
11:             $i \leftarrow i+1$
12:         else
13:             if $i = 1$ then
14:                 return $\emptyset$        ▶ No FCSS could be found for $\mathcal{R}$
15:             else
16:                 $i \leftarrow i-1$
17: $i \leftarrow 1$
18: while $\max_{\hat{M}} > 0$ do
19:     $PL \leftarrow$ FillTable($i, \boldsymbol{CS}, v_i$)
20:     $\hat{M}[i] \leftarrow \max_{j,k} PL(j,k)$
21:     if $\hat{M}[i] > 0$ then     ▶ a suitable merger can be carried out
22:         $\hat{j},\hat{k} \leftarrow \text{argmax}_{j,k} PL(j,k)$
23:         $\boldsymbol{CS}[i] \leftarrow (\boldsymbol{CS}[i] \setminus \{C_{\hat{j}}\} \setminus \{C_{\hat{k}}\}) \cup \{C_{\hat{j}} \cup C_{\hat{k}}\}$
24:     $i \leftarrow i+1$
25:     if $i > h$ then
26:         $i \leftarrow 1$
27: return $\boldsymbol{CS}$
28: procedure FillTable($i, \boldsymbol{CS}, v$)
29:     $s \leftarrow |\boldsymbol{CS}[i]|$
30:     let $PL$ be a $s \times s$ matrix initialised with $-\infty$
31:     for $j \leftarrow 1$ to $s$ do
32:         for $k \leftarrow j+1$ to $s$ do
33:             $\widehat{CS} \leftarrow (\boldsymbol{CS}[i] \setminus \{C_j\} \setminus \{C_k\}) \cup \{C_j \cup C_k\}$
34:             if Feasible($i, \widehat{CS}, \boldsymbol{CS}, \mathcal{R}$) then
35:                 $PL(j,k) \leftarrow slf(v, C_j, C_k)$             ▶ Equation 3
36:     return $PL$
```

Once a first FCSS is found, we go to the next phase, improving it (lines 18-26). We iterate over the sequence, performing one merger per table, but now we consider only suitable mergers (i.e., values greater than zero). If no such value is available in any table, then the algorithm returns the best FCSS up to that point (line 27).

4.2 MC-Link Analysis

Having described how MC-Link operates, we turn our attention to properties that it can be shown to have.

Convergence: MC-Link converges to a solution. This is due to the fact that at most $n-1$ mergers can be done per table PL. As we need to evaluate h tables, our algorithm provides an outcome after at most $n \times h$ mergers.

Anytime: As soon as MC-Link finds a CSS that satisfies relation $\mathcal{R}$, it becomes an anytime algorithm, since its solution from that moment on will only be improved at each iteration until no more mergers between coalitions are feasible.

By Condition 1, we know a CSS $\boldsymbol{CS}$ may not be feasible from the beginning of an execution, and its feasibility will depend on $\mathcal{R}$. However, we show that, if such FCSS $\boldsymbol{CS}$ exists, MC-Link will be able to find it for the relations introduced in this paper, namely $\mathcal{R}_H$, $\mathcal{R}_S$, and $\mathcal{R}_O$. We assume any $\mathcal{R}$ to be given in an algorithmic form and let α and β denote its time and space complexity, respectively.

Theorem 4.1. *Given $\mathcal{G} = \langle A, \mathcal{H}, \mathcal{R}\rangle$, in which $\mathcal{R} \in \{\mathcal{R}_H, \mathcal{R}_S, \mathcal{R}_O\}$, MC-Link eventually outputs an FCSS $\boldsymbol{CS}$, if one exists. The time complexity of MC-Link is $O(h^2n^3\alpha)$, and the space complexity is $O(n^2 + hn + \beta)$.*

Proof. For relation $\mathcal{R}_H$ this result immediately follows: the CSS containing only coalition structures of singletons is feasible.

For $\mathcal{R}_S$, we start off from a CSS $\boldsymbol{CS}$ containing only singleton coalitions. The $CS_1 \in \boldsymbol{CS}$ will dictate how CS_2 is to be constructed. That is, it is not constrained by any other coalition structure in $\boldsymbol{CS}$ and hence for its table PL_1 the condition $\max_{j,k} PL_1(j,k) > -\infty$ holds; thus, a merger is carried out. In fact, for each table PL_i, $1 \leq i \leq h$, at most $n-i$ mergers can be carried out (recall mergers of negative or zero gain are allowed). As $h \leq n$, under $\mathcal{R}_S$ an FCSS $\boldsymbol{CS}$ is eventually found. For $\mathcal{R}_O$, we have $\mathcal{R}_O \subset \mathcal{R}_S$.

Regarding the time complexity, MC-Link has two phases that share the same CSS $\boldsymbol{CS}$. Each $CS \in \boldsymbol{CS}$ has a corresponding table PL that can be constructed, when required, in $n^2 \times \alpha$ steps. To make $\boldsymbol{CS}$ feasible, MC-Link needs at most time $O(hn^3\alpha)$, as we need at most $(h-1) \times (n-1)$ steps to find an FCSS. To improve an FCSS $\boldsymbol{CS}$, in the worst case, assume the CSS of singletons is feasible, so no merger was carried out. In the worst case, $\mathcal{R}$ makes it possible only one merger per iteration (from 1 to h). That is, we reach the grand coalition in a single table in $h \times (n-1)$ iterations. As we have h tables, the time complexity of MC-Link is $O(h^2n^3\alpha)$. However, we only need space $O(n^2 + hn + \beta)$, as we evaluate one table at time and can construct it directly from a $CS \in \boldsymbol{CS}$. Although the final complexity depends on α and β, it is expected that MC-Link is typically polynomial. □

The reader should bear in mind that MC-Link will not be able to output an FCSS or improve on an initial solution depending on the given $\mathcal{R}$. In fact, MC-Link will not find an FCSS in all cases where a CS of singletons does not appear as a second element of any pair in $\mathcal{R}$ (i.e., to be a feasible end of a sequence). To address that, one can change the backtracking mechanism in the first while. Even if a feasible sequence is found, under some circumstances, no improvement can be done. For instance, consider a relation in which a pair is feasible iff the CSs have the same size: $|CS| = |CS'|$. MC-Link gets stuck at the CS of singletons as no pair in $\mathcal{R}$ allows an increase in size. Again, MC-Link needs to be altered for that particular application. However, MC-Link is useful for various relations, as the ones in Sections 2.4, 3.2, and 3.4.

5 EXPERIMENTS

In this section we report on experiments carried out to evaluate the performance of our algorithm. As it is the first algorithm for solving SCFGs, we cannot compare it to others. Instead, we developed a brute-force algorithm, specific for $\mathcal{R}_S$, to compare the quality of a solution provide by MC-Link and to evaluate its running-time performance.

All the experiments were performed on a virtual machine containing 32 GB of RAM and a CPU with four single cores of 2095 MHz each; the algorithms were implemented in Python 3.8.5.

5.1 Quality of the Solution

Our first experiment aims to compare the quality of outcomes computed by our algorithm against the optimal result. We evaluate it with different characteristic functions to understand how they affect the performance. Following the literature in coalition structure generation, we experiment with the following distributions that are defined in [11]: uniform, modified uniform, agent-based uniform, normal, modified normal, agent-based normal, NDCS, exponential, beta, and gamma. Before each experiment, we draw a value for each coalition $C \in 2^A$ from the selected distribution and store it in a table. For each game $\Gamma_i \in \mathcal{H}$, we always sample new values for the coalitions; this means all valuations v_i are different.

We chose $\mathcal{R}_S$ (see Definition 2.3) for this experiment, and implemented a brute-force algorithm especially designed for it. It searches exhaustively the search space determining a value for every FCSS. As the search space is exponential in the number of games and agents, we can only use a small set of agents, namely $n = 9$. We set the number of games to $h = 3$ (doing so, the length of the lines in the chart do not get too short). Based on this setting we experiment with all distributions mentioned above.

In the results shown in Figure 4 we see that, in general, the solutions found by MC-Link are close to the optimum. The most significant difference is regarding the *exponential*, *beta*, and *gamma* distributions. In those settings, the size of a given coalition plays a significant role and as MC-Link depends on gains over small-sized coalitions to decide on a merger, it stays far from the optimal results. On the other hand, for the normal-based distributions, in which the values are concentrated around the mean, MC-Link is able to find solutions that are, or are close to, the optimal one.

5.2 Time Analysis

Our next experiment shows how our algorithm performs when we vary and scale up the number of agents. As storing the values of a characteristic function in a table is no longer feasible, we set every game to be evaluated according to $v(C) = |C|^2$. We pick this v based on the properties shown in Section 4.2. We know MC-Link converges and we want to evaluate it in the worst-case scenario, which is when all feasible movements of each table are performed. Hence, we need v to be super-additive. We pick relation $\mathcal{R}_S$ and compare MC-Link against the brute-force algorithm developed for that relation. However, we expect the results to hold for brute-force algorithms for $\mathcal{R}_O$ and $\mathcal{R}_H$ as well. We aim to confirm that in future work.

We first compare the running time of both algorithms. In the previous section, we introduced the comparison between the quality of the solution considering at most nine agents. Now, we vary the number of games $1 \leq h \leq 9$ for different numbers of agents. For the brute-force algorithm, we pick sizes $n \in \{8, 9\}$ and for MC-Link we set $n \in \{8, 9, 29, 49, 69\}$.

The results in Figure 5 show, as expected, that MC-Link is much faster than the brute-force algorithm. The discontinuous line `Brute-Force_|A|=9` means a timeout of one hour was reached, therefore no solution was recorded. The line `Brute-Force_|A|=8` shows an interesting behaviour. This is due to the fact that the number of solutions to be evaluated—which is a combination of coalition structures from which we want to select h coalition structures—will first increase and then decrease as h approaches n.

We also experiment with MC-Link alone, varying the number of agents up to $n = 100$. We show the results in Figure 6. Each line represents a different number of games, and appears in the figure when the number of agents is sufficient (if $h > n$ there exists no hierarchy for $\mathcal{R}_S$). We can see that even with a heuristic approach, the running time increases significantly. However, even real-world problems may not require a large number of different games (e.g., the problem described in Section 3.4). In addition, the characteristic function may limit positive gains of mergers to a few coalitions, therefore less operations would be required. We aim to improve the algorithm in future work to allow it to deal with hundreds of agents while keeping a quality bound on the optimal solution.

6 RELATED WORK

Our SCFG is a new concept in coalitional games and we have shown that it can be applied to interesting domains and problems. As we were not able to find in the literature any algorithms that output a sequence of interrelated coalition structures, we cannot do any direct comparisons. We therefore discuss in this section how SCFG relates to various approaches in the literature.

Many variations of coalitional games have been proposed, for instance, considering tasks [15], constraints [13], etc. An SCFG extends the idea of coalition structure in the literature [14] to a total order of such structures in which the interdependence between them is established by a binary relation. In fact, some coalitional games are a specialisation of SCFG as shown in Section 2.2.

The idea of interdependence between CFGs are reminiscent of combinatorial auctions [3, 10] and overlapping coalition formation [17]. Combinatorial auctions address the problem where, given a set of items, a set of buyers place bids for a subset of such items (each buyer may evaluate differently each item) and the aim is to maximise the overall buyers' social welfare given a partition of items [10]. On the other hand, coalition formation with overlaps drops the constraint of disjoint coalitions in a given coalition structure [17]. A coalition becomes a vector (of length $|A|$) and each agent establishes a desired contribution to it; a contribution of 0 means the agent does not participate in that coalition. It can be noted that both approaches output a single coalition structure and therefore address a different problem than ours.

A related field to coalition formation is clustering. It plays an important role, especially in dealing with databases, when we aim to group objects based on given criteria. Different approaches for this problem have been proposed and some of them are related to ours, for instance, *meta-clustering* [5]. In that setting, the aim

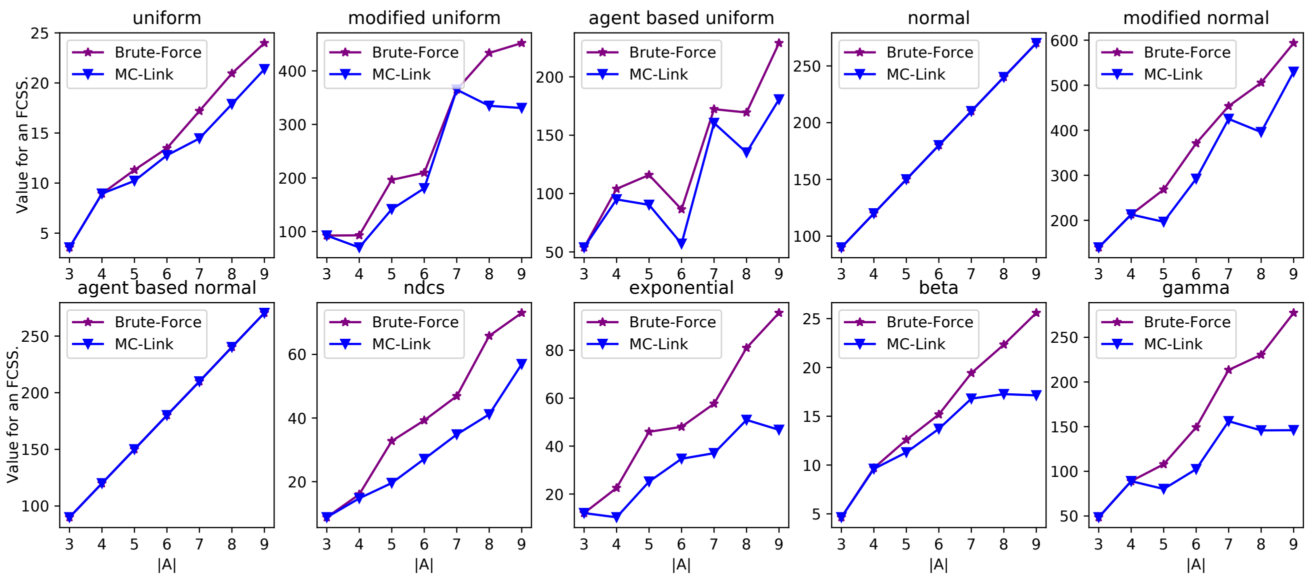


Figure 4: Quality of the solution found by MC-Link compared to a brute-force algorithm.

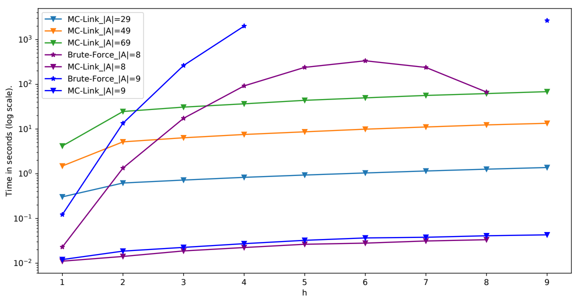


Figure 5: Comparison between the time required to provide a solution using MC-Link and a brute-force algorithm.

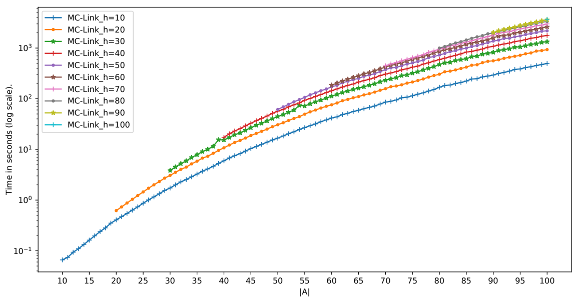


Figure 6: MC-Link running time for different lengths of $\mathcal{H}$ and varying the number of agents.

is to group different clustering solutions for a given database. A solution is a partition of objects into clusters; a meta-clustering algorithm works over solutions of clustering algorithms. Another interesting problem is *top-k* clustering [7]. In that problem, given a three-tuple graph having a set of vertices, edges, and attributes, the goal is, from a set of candidate solutions, to select k partitions that maintain quality and are dissimilar from each other. It is not hard to see that this problem could be formulated as an SCFG, by setting $k = h$ and choosing an appropriate $\mathcal{R}$.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we formally defined a sequential coalitional game SCFG and introduced an algorithm for computing an FCSS as its solution. A sequence of coalition structures (of the same length as the number of games) is generated in which the coalition structures are constrained by a binary relation. Our setting is expressive enough to represent some games previously proposed in the literature (see Theorem 2.2). We also showed that a real-world application can be naturally modelled as an SCFG (see Theorem 3.1). Moreover, we formally analysed our algorithm and showed experimentally that it can handle instances with a reasonable number of agents.

This new form of games is clearly challenging and opens various new research directions, for instance on payoff distribution, another important aspect of the coalition formation process. We would like to investigate also the expressiveness of relation $\mathcal{R}$ and to determine how typical properties of binary relations affect the game and its solutions. Both directions require further complexity analysis. In terms of practical applications, we aim to use the formulation of an ICS structure for applications in disaster management systems and experiment with real-world data. Finally, we also would like to improve the algorithm proposed here to provide a quality bound on the optimal solution.

ACKNOWLEDGMENTS

The first author acknowledges funding by CAPES—Brasil—Finance Code 001. The last author gratefully acknowledges the support from CNPq and CAPES.

REFERENCES

[1] Filippo Bistaffa, Alessandro Farinelli, Georgios Chalkiadakis, and Sarvapali D. Ramchurn. 2017. A Cooperative Game-theoretic Approach to the Social Ridesharing Problem. *Artificial Intelligence* 246 (May 2017), 86–117.

[2] Alessandro Farinelli, Manuele Bicego, Filippo Bistaffa, and Sarvapali D. Ramchurn. 2016. A Hierarchical Clustering Approach to Large-scale Near-optimal Coalition Formation with Quality Guarantees. *Engineering Applications of Artificial Intelligence* 59 (Dec. 2016), 170–185.

[3] Michal Feldman, Nick Gravin, and Brendan Lucier. 2015. Combinatorial Auctions via Posted Prices. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, USA, 123–135.

[4] FEMA. 2017. *National Incident Management System* (3 ed.). Independently Published.

[5] Alessio Ferone and Antonio Maratea. 2020. Decoy Meta–Clustering Through Rough Graded Possibilistic C-Medoids. In *2020 IEEE Conference on Evolving and Adaptive Intelligent Systems*. IEEE, 1–7.

[6] Gianluigi Greco and Antonella Guzzo. 2017. Constrained Coalition Formation on Valuation Structures: Formal framework, applications, and islands of tractability. *Artificial Intelligence* 249 (Aug. 2017), 19–46.

[7] Gustavo Paiva Guedes, Eduardo Ogasawara, Eduardo Bezerra, and Geraldo Xexeo. 2016. Discovering Top-k Non-Redundant Clusterings in Attributed Graphs. *Neurocomputing* 210 (Oct. 2016), 45–54.

[8] Robert L. Irwin. 1989. *Disaster response: Principles of preparation and coordination.* C.V. Mosby, St. Louis, MO, Chapter The Incident Command System (ICS), 303.

[9] Tabajara Krausburg. 2021. Hierarchical Coalition Formation in Multi-agent Systems. In *Proceedings of 17th International Conference on Distributed Computing and Artificial Intelligence, Special Sessions*. Springer International Publishing, Cham, 210–214.

[10] Piotr Krysta and Carmine Ventre. 2015. Combinatorial auctions with verification are tractable. *Theoretical Computer Science* 571 (March 2015), 21–35.

[11] Tomasz Michalak, Talal Rahwan, Edith Elkind, Michael Wooldridge, and Nicholas R. Jennings. 2015. A Hybrid Exact Algorithm for Complete Set Partitioning. *Artificial Intelligence* 230 (Oct. 2015), 14–50.

[12] Tapio Niemi, Jyrki Nummenmaa, and Peter Thanisch. 2001. Logical Multidimensional Database Design for Ragged and Unbalanced Aggregation. In *Proceedings of the 3rd International Workshop on Design and Management of Data Warehouses (CEUR Workshop Proceedings, Vol. 39)*. CEUR-WS, 7.

[13] Talal Rahwan, Tomasz P. Michalak, Edith Elkind, Piotr Faliszewski, Jacek Sroka, Michael Wooldridge, and Nicholas R. Jennings. 2011. Constrained Coalition Formation. In *Proceedings of the 25th International Conference on Artificial Intelligence.* AAAI Press, 719–725.

[14] Talal Rahwan, Tomasz P. Michalak, Michael Wooldridge, and Nicholas R. Jennings. 2015. Coalition Structure Generation: a Survey. *Artificial Intelligence* 229 (Dec. 2015), 139–174.

[15] T. Rahwan, T. Nguyen, T. P. Michalak, M. Polukarov, M. Croitoru, and N. R. Jennings. 2013. Coalitional Games via Network Flows. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence.* IJCAI/AAAI, 324–331.

[16] Sergios Theodoridis and Konstantinos Koutroumbas. 2008. *Pattern Recognition, Fourth Edition* (4th ed.). Academic Press, Inc., USA.

[17] Yair Zick, G. Chalkiadakis, and E. Elkind. 2012. Overlapping coalition formation games: Charting the tractability frontier. *Proceedings of 11th International Conference on Autonomous Agents and Multiagent Systems 2012, AAMAS 2012: Innovative Applications Track* 1 (2012), 176–183.

[18] Yair Zick, Georgios Chalkiadakis, Edith Elkind, and Evangelos Markakis. 2019. Cooperative games with overlapping coalitions: Charting the tractability frontier. *Artificial Intelligence* 271 (June 2019), 74–97.