Artificial Intelligence 174 (2010) 20–50

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Solving coalitional resource games

Paul E. Dunne[b], Sarit Kraus[a,c,*], Efrat Manisterski[a], Michael Wooldridge[b]

[a] *Department of Computer Science, Bar-Ilan University, Ramat Gan, 52900, Israel*
[b] *Department of Computer Science, University of Liverpool, Liverpool L69 7ZF, UK*
[c] *Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, USA*

ARTICLE INFO

Article history:
Received 10 April 2008
Received in revised form 18 September 2009
Accepted 22 September 2009
Available online 25 September 2009

Keywords:
Coalitional games
NTU games
Solution concepts
The core
Bargaining
Algorithms
Complexity

ABSTRACT

Coalitional Resource Games (CRGS) are a form of Non-Transferable Utility (NTU) game, which provide a natural formal framework for modelling scenarios in which agents must pool scarce resources in order to achieve mutually satisfying sets of goals. Although a number of computational questions surrounding CRGS have been studied, there has to date been no attempt to develop solution concepts for CRGS, or techniques for constructing solutions. In this paper, we rectify this omission. Following a review of the CRG framework and a discussion of related work, we formalise notions of coalition structures and the core for CRGS, and investigate the complexity of questions such as determining nonemptiness of the core. We show that, while such questions are in general computationally hard, it is possible to check the stability of a coalition structure in time exponential in the number of goals in the system, but polynomial in the number of agents and resources. As a consequence, checking stability is feasible for systems with small or bounded numbers of goals. We then consider constructive approaches to generating coalition structures. We present a negotiation protocol for CRGS, give an associated negotiation strategy, and prove that this strategy forms a subgame perfect equilibrium. We then show that coalition structures produced by the protocol satisfy several desirable properties: Pareto optimality, dummy player, and pseudo-symmetry.

1. Introduction

There is currently much interest in the possibility of delegating complex tasks to semi-autonomous software agents [34,37]. A highly desirable requirement for such domains is that the agents should be able to reach agreements with one-another on matters of common interest. For example, in order to accomplish the goals that they have been delegated, agents may need to share a scarce resource, work together to achieve a common goal, or come to a common understanding about a disputed domain of discourse. This requirement has motivated work in automated negotiation [22,28], online auctions [11] and computational social choice theory [15].

In this paper, our interest lies in domains with the following characteristics:

We desire some goal to be achieved, and delegate the goal to an agent, along with some bundle of resources, which may be expended by the agent in order to accomplish the goal. Accomplishing the goal may require resources not possessed by the agent, prompting the need for cooperation. A group of agents will thus pool resources to accomplish a set of goals to the satisfaction of all contributors. Our primary aim is for the agent to satisfy our goal, and to this end, if it is necessary to expend all the resources we endow it with,

* Corresponding author.
E-mail addresses: ped@csc.liv.ac.uk (P.E. Dunne), sarit@macs.biu.ac.il (S. Kraus), mjw@csc.liv.ac.uk (M. Wooldridge).

0004-3702/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.artint.2009.09.005

then this is acceptable. However, if there are multiple ways of satisfying the goal, we desire that the agent should minimize resource usage.

The formal model we use to capture such scenarios is a refinement of the *Coalitional Resource Games* (CRGs) framework, which was introduced in [39]. In a CRG, each agent has a set of goals, and is endowed with some quantity of resources; each goal requires a specified quantity of each resource in order to accomplish it. The main change to the basic CRG model that we make here is to introduce the idea of preferring outcomes that minimize resource consumption. To capture these preferences, we introduce costs: the cost to an agent of a particular scenario is simply the sum of the resource quantities it contributes in this scenario.

Many important real world scenarios seem to fit within this model. Consider agents that support arranging carpool schemes.[1]

The idea in a carpool scheme is to encourage people who live and work close to each other to share cars to and from work, rather than each driving their own car. We model car pooling in our framework as follows. A possible goal of an agent is arranging transportation for all necessary days and times. An agent may have more than one possible goal when the transportation can be on different possible days (e.g., a worker might visit the main office on either Monday or Wednesday). The resources are places in cars for specific days, origin, destination, and time. An agent would prefer that another agent provides a ride, if possible, even if he is available on that day and has the car, since this will save his resources.

One characteristic of our domains is that utility is *non-transferable*. In cooperative games with transferable utility, the value of a coalition is simply a real number, corresponding to payoff that can be divided amongst coalition members in any way they see fit [25, p. 257]. In such games, utility is transferable because it can be transferred between coalition members. Most existing work in multi-agent negotiation and resource allocation assumes transferable utility [22,28]. The rationale for our choice is that there *are* domains in which utility is non transferable, which is why *non-transferable utility games* (NTU games) have been studied in the literature [25, p. 268]. As an example with respect to our domain, when scientists collaborate on a joint paper, the resources they contribute include their experience and expertise. Such resources are not easy to value or trade explicitly, and the utility that one scientist gets from publishing a paper cannot usually be transferred to those that he or she cooperated with. (We elaborate on this issue in more detail in Section 3.)

We emphasise that the scenarios we consider typically require cooperation: it is not in general the case that an agent can achieve its goals in isolation, and will need to cooperate with others in order to do so. The possible outcomes of these cooperative games are structures in which coalitions commit to achieve certain goals, and in which members of a coalition each commit to contribute some part of their resource endowment. Presented with a number of different possible outcomes, we want an agent to choose one that achieves its delegated goal while minimizing the cost to itself (i.e., minimizes the quantity of resources that it contributes). Given that we are in the realm of cooperative games, a number of issues thus suggest themselves for consideration [29]: Which coalitions will form? And how will these coalitions choose an outcome from those that are available to them, given the different preferences that agents have over outcomes?

With respect to the former question, we formalise *the core* for our domain, and investigate the complexity of several questions surrounding the core. We show that it is co-NP-complete to check whether a particular outcome is in the core of a game, while it is co-NP-hard to check whether the core is non-empty, or to check whether the core is non-empty and contains a non-trivial outcome. However, as we will see, the core may be empty even with quite strong constraints on games, and this motivates us to consider other types of outcomes.

Our second contribution is to consider a constructive, bargaining-based approach to coalition structure generation. We present a negotiation protocol for the domain, and investigate its properties. Using backward induction, we derive strategies that are in subgame perfect equilibrium, and prove that outcomes generated will, on average, satisfy three desirable properties: pseudo-symmetry, dummy player, and Pareto optimality. Although n-agent bargaining for NTU games has been considered in the game theory literature [16], there has been little work on this in computer science/AI (see e.g., [7]).

A comment on notation and proofs. The remainder of the paper makes use of much notation, and for the reader's convenience, we summarise the main notations used in Appendix A. In addition, in the interests of readability, we omit all longer proofs from the main text, presenting them instead in Appendix B.

[1] This is not a frivolous example: car pooling is a major activity, heavily promoted by some national governments as a means of reducing road traffic (and hence pollution, etc.). See, for example:

`http://www.carpoolworld.com/.`

Carpools have already been the subject of study using coalitional games, although from a somewhat different (more abstract) perspective than the present paper [24].

2. Background and related work

If we are to build computer programs that can cooperate with each other, then it is natural to ask what models have been previously developed to model cooperative scenarios. Game theory is a valuable source of models for multi-agent systems in general, and *cooperative game theory* in particular studies cooperation, addressing itself to such problems as who will cooperate with who (i.e., which coalitions will form), and how such coalitions will share the benefits of cooperation.

Within cooperative game theory, perhaps the simplest, best-known, and most widely studied model of cooperative games is the *coalitional game with transferable utility* [25, p. 257]. Such a game is a structure $\langle Ag, \nu\rangle$ with Ag being a set of players, and $\nu : 2^{Ag} \to \mathbb{R}$ being the *characteristic function* of the game, which assigns a real valued utility, or payoff, to every possible coalition. The intuition is that $\nu(C)$ is the value that coalition C could earn, should they choose to cooperate with one-another. Notice that how a characteristic function is obtained or defined for any given scenario is not considered: such concerns are not considered at this abstract level of modelling. Given such structures, cooperative game theory attempts to answer such questions as which coalitions might be formed by rational agents, and how the payoff received by a coalition might be "reasonably" divided between the members of that coalition. With respect to the former question, concepts such as imputations, the core, stable sets, the kernel, and the nucleolus have been formulated [25]. These concepts represent progressively richer attempts to formalise when a coalition is stable (in the sense that there would be no incentive for a rational agent to do other than remain a member of the coalition).

A number of authors have taken ideas from cooperative game theory and attempted to apply them in multiagent systems. Sandholm et al. identify three key issues that have been addressed [29, pp. 210–211]:

- *Coalition structure generation*: The partitioning of a group of agents into coalitions, where the overall partition is a *coalition structure*.
- *Solving the optimization problem of each coalition*: Solving the "joint problem" of a coalition, i.e., finding the best way to maximize the utility of the coalition itself.
- *Dividing the value of the solution for each coalition*: Deciding "who gets what" in the final payoff of the game. With respect to this issue, concepts such as the Shapley value [25, p. 289] have been developed, which attempt to answer the question of how much an agent should receive based on an analysis of how much that agent contributes to a coalition.

With respect to coalition formation, the main approach considered in both the coalitional game literature and the multi-agent systems literature is to consider the *core* [25, p. 257]. The core of a coalitional game is the set of possible outcomes (i.e., distributions of coalitional value to members of a coalition) to which no sub-coalition could possibly object, in the sense of being able to obtain an outcome for themselves that was strictly preferred by all sub-coalition members. The fact that the core is non-empty is a necessary (but not sufficient) condition for coalition formation: if the core is empty, then the coalition will not form because some coalition has an incentive to work on their own.

With respect to distributing coalitional value, the *Shapley value* provides the best-known solution [25, p. 291]. The Shapley value states that an agent should get the average amount that it contributes to coalitions, considering all possible coalitions, which may be formed in any possible order. The Shapley value is particularly compelling because it is the unique solution to a set of axioms that characterise "fair" distributions of payoff to members of a coalition.

If we are interested in computational applications of coalitional games, it is of course important to know how hard it is to compute solution concepts such as the core and Shapley value. Since these problems involve quantifying over coalitions, it appears at first sight that they must be computationally very costly. However, this in fact depends upon the *representation* used for the game in the input to the decision problem. The naive representation of a coalitional game (explicitly listing the value of every coalition) is of size exponential in the number of agents, and, for example, checking core non-emptiness can be done in time polynomial in the size of this representation. However, it is generally accepted that such a representation is not of practical interest, and so much effort has been focussed around the complexity of computing solution concepts for *compact* or *succinct* representations of games.

A number of compact representations have been investigated. Deng and Papadimitriou, in perhaps the first detailed study of the complexity of cooperative games, considered a representation based on weighted graphs [12]. The idea was to have a graph $\langle Ag, E \subseteq Ag \times Ag\rangle$ with weights $w_{i,j}$ for every edge $(i, j) \in E$ in the graph. To compute the value $\nu(C)$ of a coalition in such a representation, we simply take the sum of weights of all edges in the graph induced by coalition C:

$$\nu(C) = \sum_{(i,j)\in E \,\&\, \{i,j\}\subseteq C} w_{i,j}.$$

The representation is compact because we only need to represent $O(|Ag|^2)$ weights $w_{i,j}$ in the input. However, checking core non-emptiness is NP-complete with such a representation, although computing the Shapley value can be done in polynomial time.

Other succinct representations for coalitional games have been considered in the literature. For example, Ieong and Shoham developed *marginal contribution nets*, a rule-based formalism for representing coalitional games [20], which generalises the induced sub-graph representation of Deng and Papadimitriou. They showed that while computing some solution concepts (e.g., the core) was hard with this representation, others — notably the Shapley value — were computationally easy. Conitzer and Sandholm considered the complexity of computing solutions based on a representation of *superadditive* games [9]. Elkind et al., building on results of Deng and Papadimitriou, investigated the complexity of solution concepts in weighted voting games [14]: here, computing the Shapley value is hard, while checking core-non-emptiness is easy. Bachrach and Rosenschein considered several other models for coalitional games, and the complexity of solution concepts on these games [2,3].

Another approach to coalition formation that has received much attention in the literature is that of forming coalition structures so as to *maximize the social welfare of the system*, that is, maximizing the sum of the values of the individual coalitions. Formally, given a coalitional game $\langle Ag, \nu \rangle$, the optimal coalition structure CS^* is given as follows:

$$CS^* = \arg \max_{CS \in \text{ partitions of } Ag} V(CS)$$

where

$$V(CS) = \sum_{C \in CS} \nu(C).$$

There will of course be exponentially many possible coalition structures, and so the problem of finding CS^* is computationally complex. Sandholm and colleagues developed algorithms to find optimal coalition structures CS^* (i.e., partitions of agents) with worst case guarantees (that is, within some given ratio bound k of optimal) [29]. They showed that finding the optimal coalition structure is NP-hard [29, pp. 224–225]. They were able to show that for a ratio bound $k = a$ (where a is the number of agents) their algorithm required searching 2^{a-1} nodes, and that in a precise sense, this is the best that could be expected of such an algorithm. They also went on to establish how more extensive search might be used to lower the bound k. In earlier work, Shehory and Kraus developed algorithms for coalition structure formation in which agents were modelled as having different capabilities, and were assumed to benevolently desire some overall task to be accomplished, where this task had some complex (plan-like) structure [31–33].

In coalitional games with transferable utility, the payoff or utility obtained by a coalition may be arbitrarily divided between coalition members. Thus *side payments* are possible between agents, which facilitates some outcomes that would not otherwise be feasible (see, e.g., [13]). However, such transferable utility is not a realistic assumption in many domains, and *coalitional games with non-transferable utility* (more commonly referred to simply as NTU games), model such scenarios. Formally, an NTU game can be understood as a structure $\langle Ag, \Omega, \succ_1, \ldots, \succ_n, \nu \rangle$ where Ag is a set of players, Ω is a set of *outcomes*, $\succ_i \subseteq \Omega \times \Omega$ is a binary preference relation over Ω for player $i \in Ag$, and $\nu : 2^{Ag} \to 2^{\Omega}$ is the characteristic function of the game, with the intended interpretation that $\nu(C)$ are all the outcomes that $C \subseteq Ag$ could cooperate to achieve. NTU games generalise coalitional games with transferable utility: we can understand $\nu(C)$ as being the set of distributions of payoff that C can obtain to members of C.

While the core has the same interpretation in NTU games as in TU games, it is less obvious how the Shapley value might be interpreted for NTU games. Hart and Mas-Colell propose a negotiation protocol for NTU games, and show that the protocol leads to outcomes which closely correspond to the Shapley value in the transferable utility case [16].[2] However, several quite significant restrictions are placed on the NTU games considered in [16]: in particular, the set of outcomes is infinite and continuous. It is not clear how these solutions might be applied to finite, discrete NTU games. Vidal-Puga report similar work, giving a similar bargaining protocol which leads to outcomes that correspond to the Owen value, an analogue of the Shapley value [36]. Bloch and Diamantoudi present a bargaining protocol for a class of NTU games called *hedonic* games [5]. In a hedonic game, we do not start with a characteristic function, but simply a preference ordering, one for every agent, over every coalition; in other words, an agent wants to join a coalition "for the pleasure of their company" (whence the term "hedonic"). They show that under certain circumstances, using this protocol leads to outcomes in the core of the game.

Two closely related types of NTU game were introduced specifically to model goal-oriented multi-agent domains, i.e., domains in which agents have goals to achieve. In a *qualitative coalitional game* (QCG), every agent has some set of goals, and desires to accomplish at least one of these, but is indifferent between its goals [38]. Each coalition is assumed to have a set of choices, each choice representing a set of goals that would be accomplished were the coalition to make the corresponding choice; coalitions then form to achieve mutually satisfying sets of goals. Such games are termed "qualitative" because there is no numeric measure of utility in such games: agents are simply satisfied or not. Coalitional resource games (CRGs), the model underpinning the work of the present paper, were introduced in [39]. The idea in CRGs was to consider situations in which the choices available to a coalition derive from the resources available to the coalition: each agent is endowed with a collection of resources, and different goals require different amounts of each resource to achieve them. Note that as considered in [39], CRGs retain the qualitative nature of QCGs: an agent's aim is simply to ensure that one if its goals is achieved. The present paper is motivated in part by one obvious aspect of resource-based situations that was not considered in [39]: namely, the very obvious desire to *minimize resource usage*. While resource bounds were considered in [39], no consideration was given to preferring solutions that minimize resource usage. So, while we base our work on the model of CRGs presented in [39], this model is extended with a natural notion of preference, to capture this very natural

[2] The Hart and Mas-Colell protocol was influential in determining the bargaining protocol of the present paper. Although there are several differences, as follows. Once the order of agents is chosen, our protocol is deterministic; in addition we allow the agents to give counter offers.

requirement. By introducing preferences, we also allow for solution concepts that could not be framed in the setting of [39]: for example, the notion of stability as embodied in the core.

3. Coalitional Resource Games (CRGs)

CRGs contain a non-empty, finite set $Ag = \{a_1, \ldots, a_n\}$ of *agents*. A *coalition*, typically denoted by C, is simply a set of agents, i.e., a subset of Ag. The *grand coalition* is the set of all agents, Ag. Each agent $i \in Ag$ is assumed to have associated with it a (finite) set G_i of *goals*, drawn from a set of overall possible goals G. The intended interpretation is that the members of G_i represent all the different ways that agent i's goals might be satisfied. That is, agent i would be happy if *any* member of G_i were achieved — but we are not concerned with preferences over individual goals. Thus, at this level of modelling, i is *indifferent* among the members of G_i: it will be *satisfied* if *at least one* member of G_i is achieved, and *unsatisfied* otherwise. Note that cases where more than one of an agent's goals are satisfied are not an issue — an agent's aim will simply be to ensure that at least one of its goals is achieved, and there is no sense of an agent i attempting to satisfy as many members of G_i as possible.

In order to bring about their goals, agents must expend *resources*. We assume a (fixed, finite, non-empty) set of resources, R, and assume that each agent is *endowed* with a (possibly zero) natural number quantity of each resource. We denote the amount of resource $r \in R$ that agent $i \in Ag$ is endowed with by $\mathbf{en}(i,r)$, thus $\mathbf{en}(i,r) \in \mathbb{N}$. Different goals may require different quantities of each resource for their achievement. We denote the amount of resource r required to achieve goal g by $\mathbf{req}(g,r)$; again, we assume that $\mathbf{req}(g,r) \in \mathbb{N}$.

Collecting these components together, we get *coalitional resource games* (CRGs). A CRG Γ is an $(n+5)$-tuple:

$$\Gamma = \langle Ag, G, R, G_1, \ldots, G_n, \mathbf{en}, \mathbf{req} \rangle$$

where:

- $Ag = \{a_1, \ldots, a_n\}$ is a set of *agents*;
- $G = \{g_1, \ldots, g_m\}$ is a set of *possible goals*;
- $R = \{r_1, \ldots, r_t\}$ is a set of *resources*;
- for each $i \in Ag$, $G_i \subseteq G$ is a set of goals, the intended interpretation being that any of the goals in G_i would satisfy i — but i is indifferent between the members of G_i;
- $\mathbf{en} : Ag \times R \to \mathbb{N}$ is an *endowment function*, with the intended interpretation that if $\mathbf{en}(i,r) = k$, then agent $i \in Ag$ is endowed with quantity $k \in \mathbb{N}$ of resource $r \in R$; and
- $\mathbf{req} : G \times R \to \mathbb{N}$ is a *requirement function*, with the intended interpretation that if $\mathbf{req}(g,r) = k$, then to achieve goal $g \in G$, it is necessary to expend quantity $k \in \mathbb{N}$ of resource $r \in R$.

We will assume that no goal in G is "trivially" attainable, i.e., every goal requires a non-zero expenditure of *at least one* resource. This assumption seems reasonable, since goals requiring no resources can be eliminated without altering the strategic structure of a game (since every agent can achieve such a goal, then these goals have no effect on the formation or otherwise of specific coalitions). Formally, we assume that

$$\forall g \in G, \exists r \in R \text{ such that } \mathbf{req}(g,r) > 0.$$

We extend the endowment function **en** to coalitions via the function $en : 2^{Ag} \times R \to \mathbb{N}$.

$$en(C,r) = \sum_{i \in C} \mathbf{en}(i,r).$$

Similarly, we extend the **req** function to sets of goals via the function $req : 2^G \times R \to \mathbb{N}$.

$$req(G',r) = \sum_{g \in G'} \mathbf{req}(g,r).$$

With a little abuse of notation, we use $req(G')$ to denote the total cost of resources that are required to satisfy the set of goals G':

$$req(G') = \sum_{r \in R} req(G',r).$$

A set of goals G' *satisfies* agent i if $G' \cap G_i \neq \emptyset$; we say that G' satisfies coalition $C \subseteq Ag$ if it satisfies every member of C.

A set of goals G' is *feasible* for coalition C if that coalition is endowed with sufficient resources to achieve all the goals in G'. Notice that monotonically increasing coalitions have monotonically increasing feasible goal sets. That is, if $C \subseteq C'$, then if G' is feasible for C then G' is feasible for C'. In the terminology of [38], CRGs are thus inherently *coalition monotonic*.

We define a function $sf : 2^{Ag} \to 2^G$ to return the set of goal sets that both satisfy and are feasible for a given coalition.

$$sf(C) = \{G' \subseteq G\colon\ G' \text{ is feasible for and satisfies } C\}.$$

Table 1
Endowments, feasible goal sets, and satisfying feasible goal sets for coalitions in Example 1.

	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
$en(C_x, r_1)$	0	2	0	2	1	3	1	3
$en(C_x, r_2)$	0	0	1	1	2	2	3	3
$feas(C_x)$	$\emptyset$	$\emptyset$	$\emptyset$	$\{g_2\}$	$\emptyset$	$\{\{g_1\}, \{g_2\}\}$	$\emptyset$	$\{\{g_1\}, \{g_2\}\}$
$sf(C_x)$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\{g_1\}$	$\emptyset$	$\emptyset$

We say a coalition C are *successful* if $sf(C) \neq \emptyset$: thus, if C are successful, then C are endowed with the resources to bring about some goal set G' such that G' will satisfy every member of C.

Given a set of goals G', we denote by $succ(G')$ the set of agents that would have their goals satisfied by G':

$$succ(G') = \{i\colon\ i \in Ag \ \&\ G' \cap G_i \neq \emptyset\}.$$

Consider the following example.

Example 1. Consider the following CRG, which we refer to as Γ_1. We have three agents, $Ag = \{a_1, a_2, a_3\}$, with two possible goals, $G = \{g_1, g_2\}$, and two resources $R = \{r_1, r_2\}$. The goal sets for each agent are as follows:

$$G_1 = \{g_1\} \qquad G_2 = \{g_2\} \qquad G_3 = \{g_1, g_2\}.$$

The endowment function **en** is defined as follows:

$$\mathbf{en}(a_1, r_1) = 2 \qquad \mathbf{en}(a_1, r_2) = 0$$
$$\mathbf{en}(a_2, r_1) = 0 \qquad \mathbf{en}(a_2, r_2) = 1$$
$$\mathbf{en}(a_3, r_1) = 1 \qquad \mathbf{en}(a_3, r_2) = 2.$$

And the requirement function as follows:

$$\mathbf{req}(g_1, r_1) = 3 \qquad \mathbf{req}(g_1, r_2) = 2$$
$$\mathbf{req}(g_2, r_1) = 2 \qquad \mathbf{req}(g_2, r_2) = 1.$$

There are eight possible coalitions in the game, as follows:

$$C_0 = \emptyset \qquad C_1 = \{a_3\} \qquad C_2 = \{a_2\} \qquad C_3 = \{a_2, a_3\}$$
$$C_4 = \{a_1\} \qquad C_5 = \{a_1, a_3\} \qquad C_6 = \{a_1, a_2\} \qquad C_7 = \{a_1, a_2, a_3\}.$$

The endowments for these coalitions are summarised in Table 1, together with the feasible goal sets for each coalition, and the goal sets that are both feasible for and satisfy each coalition.

Thus far, our presentation of CRGs has faithfully followed that of [39]. At this point, we will start to introduce some new concepts and notations.

3.1. Contribution vectors

The first new concept we introduce captures the idea of an agent contributing a certain specified amount of each resource to a coalition. Formally, a *resource vector* is an $|R|$-tuple of natural numbers, indexed by elements of R: a resource vector will indicate the amount of each resource contributed by a specific agent. If $C \subseteq Ag$, then we say a C*-contribution vector* is a $|C|$-tuple of resource vectors, with members indexed by elements of C; each component of a C-contribution vector defines the contribution of a specific member of C.

We denote contribution vectors by ξ, ξ', etc., and we write ξ_i to denote the member of ξ indexed by i, i.e., ξ_i is the vector defining agent i's contribution to ξ. We let $\xi_{i,r}$ denote the amount of resource $r \in R$ contributed by agent $i \in Ag$ according to contribution vector ξ. We will not allow agents to contribute more of a resource than they are endowed with, and for this reason, we require that a contribution vector ξ for coalition C must satisfy the following natural *feasibility* requirement:

$$\forall i \in C, \quad \forall r \in R\colon\ \xi_{i,r} \leqslant \mathbf{en}(i, r).$$

Although we do not do so in this paper, it is of course possible to think of a contribution vector ξ as an $|Ag| \times |R|$ matrix of natural numbers, with the value $\xi[i, r]$ being the amount of resource r contributed by agent i.

Now, as stated earlier, one of the key aims of the present paper is to consider situations in which an agent has multiple options for achieving one of its goals, but these different options have different resource consumption requirements: the aim should be to minimize resource requirements. To capture the idea of resource consumption, we define the total *cost* of a contribution vector ξ for agent $i \in C$ to be the sum of its individual resource contributions. These costs will shortly be used to define preference relations, with the basic idea being that an agent prefers to minimize costs. This additive model of costs is of course relatively simple, and other models of costs and preferences are possible, but it is an useful starting point for our analysis. Formally, we denote the cost to agent $i \in Ag$ of contribution vector ξ by $c_i(\xi)$:

$$c_i(\xi) = \sum_{r \in R} \xi_{i,r}.$$

3.2. Cooperation structures

Coalitional games with transferable utility model cooperation at a very high level of abstraction. In particular, they do not indicate *what* coalitions will do – they simply indicate the *value* that the coalition may earn. This information provides the basis upon which the agent can make a decision about whether it is rational to work with one coalition or another. In CRGs, the picture is a little more complex. When an agent is deciding whether to work within a coalition, it needs to know what goals the coalition will work on, and what the expected resource contribution will be. Only then can it begin to consider how desirable this outcome is, and compare it to other possibilities. We thus introduce *cooperation structures*, which represent this information. Formally, a cooperation structure is a triple:

$$\lambda = \langle C, G', \xi \rangle$$

where $C \subseteq Ag$, $G' \subseteq G$, and ξ is a C-contribution vector, such that these components must satisfy the *feasibility* constraint that the coalition is endowed with sufficient resources to achieve the goals it commits to:

$$\forall r \in R, \quad \sum_{i \in C} \xi_{i,r} \geqslant req(G', r).$$

The intended interpretation of a cooperation structure $\lambda = \langle C, G', \xi \rangle$ is thus that C will cooperate to achieve goal set G', and that each agent $i \in C$ will contribute resources ξ_i towards this joint effort.

If λ is a cooperation structure, then we denote by C_λ, G_λ, and ξ_λ the coalition, goal set, and contribution vector of λ, respectively. With a slight abuse of notation, we lift the cost function $c_i(\cdots)$ from contribution vectors to cooperation structures, writing $c_i(\lambda)$ as shorthand for $c_i(\xi_\lambda)$. With another abuse of notation, let $succ(\lambda)$ denote the set of agents that would have their goals satisfied by λ, so $succ(\lambda)$ is a shorthand for $succ(G_\lambda)$.

3.3. Preferences

We now define, for every agent $i \in Ag$, a preference relation $\succ_i$ over cooperation structures containing i. This relation captures the idea that an agent's primary objective is to satisfy one of its goals; its secondary aim is to minimize costs. Thus, an agent prefers all cooperation structures in which it has one of its goals achieved over all those in which it does not; within the set of cooperation structures in which it has a goal achieved, it prefers to minimize its costs; and similarly for the set of cooperation structures in which it does not get any goal achieved.

Formally, for all cooperation structures λ_1, λ_2 containing agent i, we define $\succ_i$ so that $\lambda_1 \succ_i \lambda_2$ iff one of the following conditions is satisfied:

1. $i \notin succ(\lambda_1)$ and $i \notin succ(\lambda_2)$ and $c_i(\lambda_1) < c_i(\lambda_2)$.
2. $i \in succ(\lambda_1)$ and $i \notin succ(\lambda_2)$.
3. $i \in succ(\lambda_1)$ and $i \in succ(\lambda_2)$ and $c_i(\lambda_1) < c_i(\lambda_2)$.

The non-strict preference relation $\succeq_i$ then has the obvious interpretation.

Now, cooperation structures are superadditive, in the sense that they can be "merged" with no loss of utility to any of the participants. To make this idea precise, we define a *disjoint union* operator on cooperation structures, as follows. Suppose cooperation structures $\lambda_1 = \langle C_1, G_1, \xi^1 \rangle$ and $\lambda_2 = \langle C_2, G_2, \xi^2 \rangle$ are such that $C_1 \cap C_2 = \emptyset$ (i.e., they have no agents in common). Then we denote by $\lambda_1 \sqcup \lambda_2$ the cooperation structure $\lambda_3 = \langle C_3, G_3, \xi^3 \rangle$ whereby:

- $C_3 = C_1 \cup C_2$;
- $G_3 = G_1 \cup G_2$;
- for all $k \in \{1, 2\}$, $\forall i \in C_k$, $\forall r \in R$: $\xi^3_{i,r} = \xi^k_{i,r}$.

The disjoint union operator $\sqcup$ on cooperation structures then has the following property:

Proposition 1. *Let λ_1 and λ_2 be cooperation structures from some CRG Γ, containing disjoint sets of agents, and let $\lambda_3 = \lambda_1 \sqcup \lambda_2$. Then for all $k \in \{1, 2\}$, $\forall i \in C_k$, $\lambda_3 \succeq_i \lambda_k$.*

Proof. Suppose not. Then either (i) some agent i had its goal achieved in the respective component cooperation structure (either λ_1 or λ_2) but not in λ_3 – but this cannot be the case by construction; or else (ii) some agent $i \in C_{\lambda_3}$ contributes more in λ_3 than it did in the respective component cooperation structure – which again cannot be the case by construction. □

It is convenient to define preference relations in another way, based on a model of utility. The idea is that the preference relations induced by these utility functions will correspond to the preference relations as defined above. It is important to understand that we view utility simply as a numeric way of capturing preferences — and in particular, the fact that we have utility *does not* mean that utility is transferable. Defining the preference relation via numeric utilities makes it possible for us to use numeric optimization techniques — in particular, integer linear programs — to compute solutions associated with preferences. However, utility here is solely a convenient numerical representation of preference; it does not form a "common currency" with which agents can compensate each other, and individual utility values cannot be compared between agents. We return to this point below.

To define the utility functions $u_i(\cdots)$ for each agent i, we proceed as follows. First, we define V to be the maximal possible cost needed for satisfying one goal for each agent:

$$V = \sum_{i \in Ag} \left(\max_{g \in G_i} \sum_{r \in R} \mathbf{req}(g, r) \right).$$

Then, for $\lambda = \langle C, G', \xi \rangle$, we define the functions $u_i(\cdots)$ as follows:

$$u_i(\lambda) = \begin{cases} V - c_i(\lambda) & \text{if } i \in succ(\lambda) \\ -c_i(\lambda) & \text{otherwise.} \end{cases}$$

Now, given a coalitional resource game Γ and cooperation structures λ_1, λ_2, suppose we defined $\lambda_1 \succeq_i \lambda_2$ iff $u_i(\lambda_1) \geqslant u_i(\lambda_2)$, with the strict preference relation $\succ_i$ defined in the obvious way. It should be immediately clear from construction that the preference relations defined in this way directly correspond to the preference relations as defined earlier.

Given that we have just introduced a model of utility, it is natural to ask whether in fact our setting can be viewed as a TU game. The answer is no. To see why, consider the following scenario. We have:

$$Ag = \{a_1, a_2\}$$
$$G = \{g_1, g_2\}$$
$$R = \{r_1, r_2\}$$
$$G_1 = \{g_1\} \qquad G_2 = \{g_2\}$$
$$\mathbf{req}(g_1, r_1) = \mathbf{req}(g_2, r_1) = 1$$
$$\mathbf{req}(g_1, r_2) = \mathbf{req}(g_2, r_2) = 0$$
$$\mathbf{en}(a_1, r_1) = 1 \qquad \mathbf{en}(a_2, r_1) = 0 \quad \text{and}$$
$$\mathbf{en}(a_1, r_2) = 0 \qquad \mathbf{en}(a_2, r_2) = 1000.$$

Notice that resource r_2 plays no part in whether or not either agent's goal can be achieved. Now, there is only one way for agent a_1 to achieve its goal: he must use his total resource endowment in order to achieve g_1. However, if this happens, then a_2 must be unsatisfied; the only way agent a_2 can be satisfied is for agent a_1 to contribute his entire endowment to the achievement of g_2 — which of course would leave a_1 unsatisfied. But this latter situation would be the *worst* outcome from the point of view of agent a_1. Recall that what an agent values above everything else is the achievement of his goal: every outcome that achieves his goal is preferred over every outcome where he does not. In this scenario, there is no side-payment possible to a_1 that could induce him to transfer his endowment of r_1 to a_2, because the *only* way a_1 can achieve his goal is by using his endowment to this end, and achieving his goal is what matters the most to agent a_1. Resources and resource consumption are a *secondary* concern: goal achievement is the primary concern.

We can express this example informally, and perhaps somewhat morbidly, by the following rather tongue-in-cheek scenario. Suppose you and I are in a plane that is about to crash. You have a parachute, but unfortunately I do not. We each have a goal of staying alive, but if you give up your parachute to me — thereby allowing me to achieve my goal of staying alive — then you will be unable to achieve your goal of staying alive. Is utility transferable here? Surely not: there is no utility I could transfer to you that would compensate you for not being able to achieve your goal!

3.4. Coalition structures

Recall from Section 2 that in coalitional games with transferable payoff, a coalition structure is simply a partition of all the agents in the system. Intuitively, each element in the partition represents a set of agents who will work together. In this highly abstract setting, there is thus no indication of what each coalition will actually do, or how they will cooperate. The coalition structure generation problem for such domains involves finding a partition that maximizes the sum of all individual coalition values. In our domain, coalition structures are somewhat more complex. They are not simply partitions of the agent population: they must indicate what goals each coalition will work to achieve, and what resources each member of the coalition will contribute. Thus, for our domain, a *coalition structure*, σ, is a set of cooperation structures

$$\sigma = \{\lambda_1, \ldots, \lambda_k\}$$

such that $\{C_{\lambda_1}, \dots, C_{\lambda_k}\}$ is a partition of Ag. If σ is a coalition structure then we denote by $\lambda_{\sigma,i}$ the cooperation structure in σ of which i is a member. We let $\sigma_{\mathbf{0}}$ denote the "null" coalition structure, in which the grand coalition does nothing, and no agent makes any contribution.

Intuitively, then, a coalition structure is the outcome of a CRG. It specifies, for every agent in the game, what coalition that agent belongs to, what that agent will contribute to its coalition, and what goals will be achieved by the coalition. Of course, not all coalition structures will be equal: an agent may prefer one over another. This leads us to consider the notion of stability.

4. Stable coalition structures: The core

As we discussed in Section 2, the core is perhaps the best-known and most widely studied solution concept in coalitional games [25, p. 258]. The core is the set of outcomes of a game, feasible for the grand coalition, such that no coalition could achieve an outcome that they all strictly prefer. In this way, the core captures the key idea of coalitional stability: if the core is non-empty, then this means there is no rational incentive for any sub-coalition to defect. Note that the core in the setting of CRGs must be formulated with respect to coalition structures: an agent needs to know what goals are expected to be achieved and what it is expected to contribute in order to be able to make a judgement about the value of a coalition structure.

Formally, a cooperation structure λ is said to *block* a coalition structure σ if

$$\forall i \in C_\lambda\colon\ \lambda \succ_i \lambda_{\sigma,i}.$$

A coalition structure σ is *stable* if it is not blocked by any cooperation structure. The *core* of a CRG is its set of stable coalition structures.

Let us establish some properties of stable coalition structures.

Proposition 2. *The core is closed under disjoint union. More precisely, if σ is in the core and $\{\lambda_1, \lambda_2\} \subseteq \sigma$ $(\lambda_1 \neq \lambda_2)$, then the coalition structure $\{\lambda_1 \sqcup \lambda_2\} \cup (\sigma \setminus \{\lambda_1, \lambda_2\})$ is in the core.*

Proof. Suppose $\sigma = \{\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_k\}$ is in the core. Let $\sigma^* = \{\lambda_1 \sqcup \lambda_2, \lambda_3, \dots, \lambda_k\}$, and suppose that σ^* is blocked by some cooperation structure λ. But then λ would also block σ, since from Proposition 1, for every agent $i \in C$, the coalition structure σ^* is at least as good (and possibly better) than σ, and for every agent $i \in Ag \setminus C$, the coalition structure σ^* is exactly as good as σ. This cannot be the case, however, since σ is in the core. Hence no such λ exists, and hence σ^* is in the core as required. $\square$

Proposition 3. *Suppose $\sigma = \{\lambda_1, \dots, \lambda_k\}$ is in the core of Γ. Then the core of Γ contains a coalition structure σ^* containing a single, grand coalition cooperation structure, i.e., such that $\sigma^* = \{\lambda^*\}$ and $C_{\lambda^*} = Ag$.*

Proof. From Proposition 2: take a coalition structure in the core, and simply keep taking the disjoint union of cooperation structures until we obtain a grand coalition cooperation structure. $\square$

Notice that by this proposition, when we are looking for stable coalition structures, we can restrict our attention to looking at cooperation structures containing the grand coalition.

We will say a stable coalition structure is *atomic* if none of its component cooperation structures can be decomposed (w.r.t. disjoint union), such that the resulting coalition structure is stable. Formally, let $\sigma = \{\lambda_1, \dots, \lambda_k\}$ be a coalition structure in the core of some CRG Γ. Then we say σ is atomic iff for all $1 \leqslant i \leqslant k$, it is not the case that there exist disjoint, non-empty cooperation structures λ_i^1, λ_i^2 such that $\lambda_i = \lambda_i^1 \sqcup \lambda_i^2$ and $\{\lambda_i^1, \lambda_i^2\} \cup (\sigma \setminus \{\lambda_i\})$ is in the core. The following is now immediate:

Proposition 4. *If the core of Γ is non-empty, then the core contains an atomic coalition structure.*

Proposition 5. *Unsatisfied agents incur no cost in any stable coalition structure. More formally, if coalition structure σ is stable, and $i \in Ag$ is such that $i \notin succ(\lambda_{\sigma,i})$, then $c_i(\lambda_{\sigma,i}) = 0$.*

Proof. Agent i can always choose the singleton cooperation structure in which it contributes nothing and achieves no goals: such a cooperation structure would block any coalition structure in which i was unsatisfied but incurred some cost. $\square$

Proposition 6. *Suppose coalition structure σ is in the core, and let C be the set of agents that are unsatisfied in σ. Then for all $C' \subseteq C$, $sf(C') = \emptyset$.*

Proof. Suppose not: then $\exists C' \subseteq C$ such that $sf(C') \neq \emptyset$. So suppose $G' \in sf(C')$. Then the cooperation structure $\langle C', G', \xi\rangle$, where ξ represents the entire endowment of every member of C', would block σ, since C' would prefer to get their goals achieved even if it cost them their entire endowment rather than not get their goals achieved at all. $\square$

Proposition 7. *The following statements are equivalent*:

1. *No coalition is successful in Γ.*
2. *The core of Γ contains a coalition structure in which no agent expends any resources and no agent is satisfied.*

Proof.

- (1) $\Rightarrow$ (2): Suppose that (1) but not (2). Then the core contains a cooperation structure in which either (*i*) some agents expend some resources or (*ii*) some agent is satisfied. For case (*i*), it follows from (1), that at least one of the agents expending some resource must have its goal unsatisfied; but then this contradicts Proposition 5. For case (*ii*), suppose some agent – call it i – is satisfied. Note that i alone cannot achieve its goal, from (1), so some other agents, call them C, must contribute resources which lead to the achievement of one of i's goal. Some member of C must be unsatisfied, otherwise $C \cup \{i\}$ would be a successful coalition, which they cannot be from (1). Again from Proposition 5 we have a contradiction.
- (2) $\Rightarrow$ (1): Follows immediately from Proposition 6. □

A obvious question is whether there are natural conditions on CRGs that would ensure that the core is always non-empty; of these conditions, perhaps the most obvious is that of requiring agents to have at most one goal. But even in this case the core can be empty, as the following example illustrates.

Example 2. Consider the following CRG. We have four agents, $Ag = \{a_1, a_2, a_3, a_4\}$, with four possible goals, $G = \{g_1, g_2, g_3, g_4\}$, and three resources $R = \{r_1, r_2, r_3\}$. The goal sets for each agent are as follows.

$$G_1 = \{g_1\} \qquad G_2 = \{g_2\} \qquad G_3 = \{g_3\} \qquad G_4 = \{g_4\}.$$

The requirement function **req** is as defined in the following table:

req$(\cdots)$	r_1	r_2	r_3
g_1	4	0	1
g_2	4	1	0
g_3	4	1	0
g_4	4	0	2

The endowment function **en** is as defined in the following table:

en$(\cdots)$	r_1	r_2	r_3
a_1	8	1	0
a_2	8	0	1
a_3	8	0	1
a_4	4	2	0

We consider whether any possible coalition could be in the core:

1. $\sigma_{\mathbf{0}}$ is not in the core, since no agent would have goals achieved by this coalition structure; and yet there are successful coalitions, which would block $\sigma_{\mathbf{0}}$, since they would rather have their goals achieved than otherwise.
2. The grand coalition, and coalitions $\{a_1, a_4\}$, $\{a_1, a_2, a_3\}$, $\{a_1, a_2, a_4\}$, $\{a_1, a_3, a_4\}$, $\{a_2, a_3\}$, $\{a_2, a_4\}$, $\{a_3, a_4\}$, $\{a_1\}$, $\{a_2\}$, $\{a_3\}$, and $\{a_4\}$ are unsuccessful. This is because, for each of these coalitions, there exists some resource such that the total amount of this resource required to satisfy the goals of all coalition members is larger than the total endowment of this resource.
3. Consider coalition $\{a_1, a_2\}$. In order to satisfy both agents' goals, 8 units of the resource r_1 are required. Therefore one of these agents must contribute at least 4 units of resource r_1. Given this we consider the two following cases:
   (a) Agent a_1 contributes at least 4 units of resource r_1. In this case the coalition can be blocked by the cooperation structure $\langle \{a_1, a_3\}, \{g_1, g_3\}, \xi \rangle$, where the values of the contribution vector ξ (that are different from 0) are: $\xi_{1,2} = 1$, $\xi_{3,1} = 8$, $\xi_{3,3} = 1$. This is because the cost to a_1 is smaller and a_3 is satisfied.
   (b) Agent a_2 contributes at least 4 units of resource r_1. The coalition can be blocked by the cooperation structure $\langle \{a_2, a_3, a_4\}, \{g_2, g_3, g_4\}, \xi \rangle$, where the values of the contribution vector ξ (that are different from 0) are: $\xi_{2,3} = 1$, $\xi_{3,1} = 8$, $\xi_{3,3} = 1$, $\xi_{4,2} = 2$ and $\xi_{4,1} = 4$. This is because the cost to a_2 is smaller and a_3 and a_4 are satisfied.
4. Coalition $\{a_1, a_3\}$ is not stable from similar considerations to the ones we specified for coalition $\{a_1, a_2\}$ (notice that agents a_2 and a_3 are symmetric, in the sense that they have goals which would require the same amount of each resource in order to satisfy them, and they have identical endowments).
5. Consider coalition $\{a_2, a_3, a_4\}$. In order to satisfy all agents goals, 12 units of resource r_1 are required. As agent a_4 has only 4 units of resource r_1, a_2 or a_3 must contribute at least 4 units of resource r_1. Given this we consider the two following cases:

(a) Agent a_2 contributes at least 4 units of resource r_1. In this case the coalition can be blocked by the cooperation structure $\langle\{a_1,a_2\},\{g_1,g_2\},\xi\rangle$, where the values of the contribution vector ξ (that are different from 0) are: $\xi_{1,1}=8$, $\xi_{1,2}=1$, $\xi_{2,3}=1$. This is because the cost to a_2 is smaller and a_1 is satisfied.
(b) Agent a_3 contributes at least 4 units of resource r_1. In this case the coalition can be blocked by the cooperation structure $\langle\{a_1,a_3\},\{g_1,g_3\},\xi\rangle$, where the values of the contribution vector ξ (that are different from 0) are: $\xi_{1,1}=8$, $\xi_{1,2}=1$, $\xi_{3,3}=1$. This is because the cost to a_3 is smaller and a_1 is satisfied.

Thus no coalition structure is stable, and so the core is empty.

4.1. Complexity of stability checking

It is natural to ask how hard it is to check whether a given coalition structure is in the core, or more generally, whether the core is non-empty. We have the following results:

Proposition 8. *Given a* CRG Γ *and coalition structure* σ *over* Γ*, the problem of checking whether* σ *is in the core of* Γ *is co-*NP*-complete.*

Proof. We work with the complement of the problem, i.e., the problem of checking, for any Γ and σ, whether σ is unstable. For membership of NP, simply guess a cooperation structure λ and check that λ blocks σ: the size of λ is clearly only polynomial in the size of Γ, and verifying that λ blocks σ can easily be done in polynomial time. For NP hardness, we reduce the problem of checking whether the grand coalition in a given CRG is successful. This problem, named GRAND COALITION SUCCESS, was proved to be NP-complete in [39]. Let

$$\Gamma=\langle Ag,G,R,G_1,\dots,G_n,\mathbf{en},\mathbf{req}\rangle$$

be the input instance of this problem. We create an instance Γ^*,σ^* as follows. First we set $Ag^*=Ag$, $G^*=G$, and $G_i^*=G_i$. R^* contains all members of R, and in addition, for each agent $i\in Ag$, we create a resource r_i. For each goal $g\in G$, and resource $r\in R$, we fix $\mathbf{req}^*(g,r)=\mathbf{req}(g,r)$, and for each new resource r_i, we fix $\mathbf{req}^*(g,r_i)=1$. Then for all $r\in R$, we fix $\mathbf{en}^*(i,r)=\mathbf{en}(i,r)$, while for all new resources r_i, we fix $\mathbf{en}(i,r_j)=|G|$ if $i=j$ and 0 otherwise. Notice that this construction has the property that (i) the only coalition in Γ^* which *can* be successful is the grand coalition; (ii) the grand coalition can succeed in Γ^* iff they can succeed in Γ. Finally, we define $\sigma^*=\{\sigma_{\mathbf{0}}\}$, i.e., σ^* contains the grand coalition cooperation structure in which no goals are achieved, and no agent incurs any cost. The *only* cooperation structures that would be block σ^* are those in which every agent is successful. We conclude that the grand coalition is successful in Γ iff σ^* is not in the core iff some coalition in Γ^* is successful iff the grand coalition is successful in Γ^*. □

Now, consider the more general problem of checking, for any given Γ, whether the core of Γ is non-empty. This amounts to checking the following property:

$$\exists\sigma^*\colon\ \sigma^*\text{ is in the core of }\Gamma$$

i.e., checking whether:

$$\exists\sigma^*\colon\ \forall C\subseteq Ag,\quad \forall\lambda_C\colon\ \Bigl[\bigvee_{i\in C}(\lambda_{\sigma^*,i}\succeq_i\lambda_C)\Bigr].$$

We can prove:

Proposition 9. *Checking whether the core of a* CRG *is non-empty is* NP*-hard.*

Proof. See Appendix B. □

Now, simply knowing that the core is non-empty may in fact not be very useful, for the following reason. Suppose *no* coalition is successful; that is, there is no set of agents that could pool their resources to achieve a mutually satisfactory set of goals. Proposition 7 tells us that in such cases, the core will contain a cooperation structure in which the grand coalition do nothing: i.e., they achieve no goals and expend no resources. In other words, the fact that the core is non-empty does not mean it contains "meaningful" coalition structures; it could be that it contains "empty" coalition structures. In regular coalitional games, such a situation would occur if no coalition could obtain non-zero utility; this is not generally considered as an outcome. So, let us say that the core is *strongly* non-empty if it is non-empty, and does *not* contain the trivial coalition structure $\sigma_{\mathbf{0}}$. Similarly, we say the core of a game is *weak* if it contains $\sigma_{\mathbf{0}}$. With respect to these notions, first notice that the reduction employed in the proof of Proposition 8 immediately gives us the following:

Proposition 10. *Checking whether the core of a* CRG *is weak is co-*NP*-complete.*

```
1. If $\exists i \notin succ(\lambda_{\sigma,i})$ such that $c_i(\lambda_{\sigma,i}) > 0$ return $\{\langle\{i\}, \emptyset, \xi_{\mathbf{0}}\rangle\}$.
2. For each $G'$ in $2^G$:
   2.1. Set $C' = \{i\colon\ i \in succ(G')$ & $(i \notin succ(\lambda_{\sigma,i})$ or $c_i(\lambda_{\sigma,i}) > 0)\}$.
   2.2. Check whether there exists a preferred contribution vector $\xi$ for $C$ and $G'$ given $\sigma$ (Proposition 12), and if so, then return
        $\langle C', G', \xi\rangle$.
3. Return "$\sigma$ is stable".
```

Fig. 1. Algorithm for finding a cooperation structure (if one exists) that blocks σ.

If we get a "no" answer to this question. then there are two possibilities. Either the core is non-empty and does not contain $\sigma_{\mathbf{0}}$, or else the core is empty. So, the question of whether or not the core is strongly non-empty is distinct from knowing whether the core is weak. We have the following:

Proposition 11. *Checking whether the core of a* CRG *is strongly non-empty is* NP*-hard.*

Proof. See Appendix B. □

4.2. Positive results

Having established a series of negative results, which imply that checking stability is going to be hard in general, it is obvious to ask whether there are any positive results relating to stability checking. We show that, given a coalition structure σ, it is possible to check whether σ is stable in time exponential in the number of goals, but polynomial in the number of agents and resources. Since in many real world situations, the number of goals is relatively small and may even be bounded by a constant, the *actual* complexity of determining whether a given coalition structure is in the core may in many cases be considered polynomial in the size of the input.

Our result is constructive, in that we present an algorithm for checking stability with the desired properties. The key step in the algorithm is given by the following result. Suppose we are given a CRG Γ, a coalition structure σ over Γ, a set of agents C in Γ and a set of goals G' in Γ: we are asked whether there is a feasible resource vector ξ such that the cooperation structure $\langle C, G', \xi\rangle$ blocks σ; if such a ξ exists, then we are asked to report it, otherwise we should announce that no such ξ exists (notice that for $\langle C, G', \xi\rangle$ to be a cooperation structure, it must be that ξ carries enough resources to achieve the goals G', and in addition, C must be endowed with sufficient of these resources). Let us call this problem that of *computing a preferred contribution vector.* We have:

Proposition 12. *The problem of computing a preferred contribution vector can be solved in polynomial time.*

Proof. The proof is given in detail in Appendix B; here we give an overview. The idea is to build a *flow network* corresponding to the desired situation, such that if a maximum flow with certain properties exists in this flow network, then we can "read off" the contribution vector ξ from this flow network. A flow network can be understood as a directed graph with a source node s and a sink node t; edges in the graph are associated with a capacity, indicating how much flow can travel along the edge. The classic question associated with flow networks is to compute the maximum flow from source s to sink t. The overall structure of the construction is illustrated in Fig. 2: the capacity x_i for edge (s, a_i) for each agent i is the maximum cost that agent a_i could incur in a preferred cooperation structure and still prefer it over σ; the capacity $\xi_{i,j}$ on edge (a_i, r_j) for agent a_i and resource r_j indicates the endowment of agent a_i with resource r_j; and the capacity y_i on edges (r_i, t) indicate the total amount of resource r_i required for goal set G'. We then ask whether there is an integer flow of at least $y_1 + y_2 + \cdots + y_t$. If the answer is yes, then the actual flow on edges (a_i, r_i) gives the preferred contribution vector.

The size of the flow network is polynomial in the size of the inputs, and computing the maximum flow can be done in time polynomial in the size of the network (e.g., using the Edmonds–Karp version of the Ford–Fulkerson algorithm). □

Given an algorithm for computing a preferred contribution vector, the overall algorithm for checking whether a coalition structure σ is stable is given in Fig. 1. We can explain the algorithm as follows:

- First, in line (1), we check to see whether any agent is left unsatisfied while incurring some cost: if so, then this immediately gives us a blocking structure by Proposition 5.
- Line (2) tries to build a blocking cooperation structure. It does this by considering each possible set of goals G' in turn, and trying to see whether there is some coalition C that would prefer G' achieved through some resource contribution. If there is such a coalition, then it must contain either agents that incur some cost in both σ and G' (who might benefit from reducing costs) or else agents that do not have their goal achieved in σ, but do get their goal achieved in G' (who would benefit from getting their goal achieved) — line (2a). At line (2b), we have a candidate goal set G' and a relevant coalition C, so we check to see whether there exists a preferred contribution vector; if such a vector ξ exists, then we return it, otherwise, we continue to the next goal set.

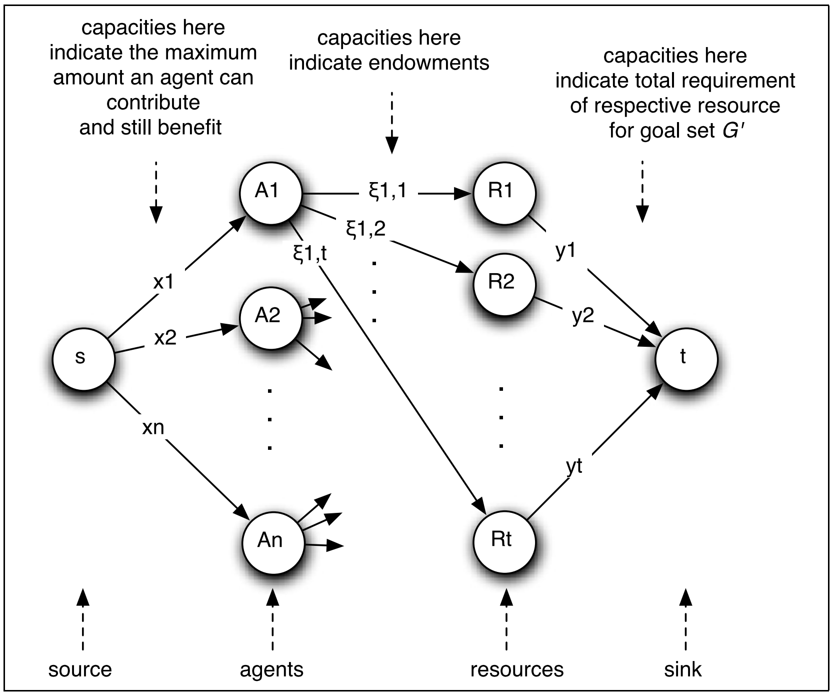
capacities here indicate the maximum amount an agent can contribute and still benefit
capacities here indicate endowments
capacities here indicate total requirement of respective resource for goal set G'
A1
A2
An
R1
R2
Rt
s
t
x1
x2
xn
ξ1,1
ξ1,2
ξ1,t
y1
y2
yt
source
agents
resources
sink

Fig. 2. Overall structure of the flow network construction for Proposition 12.

- If we have considered all goal sets, then we indicate failure, i.e., that σ is stable.

Now, the key properties of the algorithm are stated in the following proposition.

Proposition 13. *The algorithm in Fig.* 1 *is correct: if it returns "σ is stable", then σ is indeed stable, while if it returns a cooperation structure $\langle C', G', \xi\rangle$, then this cooperation structure blocks σ. Moreover, the algorithm terminates, and runs in time exponential in the number of goals but polynomial in the number of agents and resources.*

Proof. See Appendix B. □

To illustrate the algorithm, consider the following example.

Example 3. Consider the following CRG. We have three agents, $Ag = \{a_1, a_2, a_3\}$, with three possible goals, $G = \{g_1, g_2, g_3\}$, and four resources $R = \{r_1, r_2, r_3, r_4\}$. The goal sets for each agent are as follows:

$$G_1 = \{g_1\} \qquad G_2 = \{g_2, g_1\} \qquad G_3 = \{g_1, g_3\}.$$

The requirement function *req* is defined as follows:

req$(\cdots)$	r_1	r_2	r_3	r_4
g_1	4	3	14	0
g_2	0	0	3	3
g_3	0	2	3	0

The endowment function **en** is defined as follows:

en$(\cdots)$	r_1	r_2	r_3	r_4
a_1	4	3	0	0
a_2	0	3	3	0
a_3	0	0	11	3

Consider $\sigma = \{\langle Ag, \{g_1\}, \xi\rangle\}$ where

$$\xi_{1,r_1} = 4 \qquad \xi_{1,r_2} = 3 \qquad \xi_{1,r_3} = 0 \qquad \xi_{1,r_4} = 0$$
$$\xi_{2,r_1} = 0 \qquad \xi_{2,r_2} = 0 \qquad \xi_{2,r_3} = 3 \qquad \xi_{2,r_4} = 0$$
$$\xi_{3,r_1} = 0 \qquad \xi_{3,r_2} = 0 \qquad \xi_{3,r_3} = 11 \qquad \xi_{3,r_4} = 0.$$

The set of the goals includes seven nonempty sets: $\{g_1, g_2, g_3\}$, $\{g_1, g_2\}$, $\{g_2, g_3\}$, $\{g_1, g_3\}$, $\{g_1\}$, $\{g_2\}$ and $\{g_3\}$. The algorithm considers goal sets $\{g_1, g_2, g_3\}$ and $\{g_1, g_2\}$ without succeeding, and then considers $\{g_2, g_3\}$. $C' = \{a_2, a_3\}$ is the coalition of agents that would have their goals satisfied by $\{g_2, g_3\}$ and do not obtain their maximum utility from σ, and we check to see whether a preferred contribution vector exists; in this case it does, and the non-zero contributions are $\xi_{2,r_2} = 2$, $\xi_{3,r_3} = 6$ and $\xi_{3,r_3} = 3$. The algorithm returns the cooperation structure $\langle\{a_2, a_3\}, \{g_2, g_3\}, \xi\rangle$ and terminates. Note that we illustrate the construction of the preferred contribution vector for this example in the proof of Proposition 12.

5. Fair coalition structures through bargaining

In the previous section, we investigated one type of solution for CRGs, in which the main concern was to check whether a particular coalition structure was stable against defections. This assumes that a particular coalition structure has been found, but of course does not address the issue of where such a structure comes from. In this section, we address the issue of the method by which a group of agents can find a coalition structure. Our approach is to present a *negotiation protocol* for finding coalition structures. Although negotiation protocols have long been the object of study in multi-agent systems research (see e.g., [22,28] for well-known examples), little research in computer science/AI has addressed n-agent NTU bargaining (see Section 2).

We are looking for a protocol such that it will be possible to compute in reasonable time the strategies that are in equilibrium and that the resulted coalitions will satisfy some desirable properties.

5.1. A negotiation protocol for coalition formation

We now present and analyze a negotiation protocol by means of which the agents within a CRG can agree upon a coalition structure to implement: the resulting coalition structure, while not necessarily maximizing overall social welfare, will nonetheless exhibit a number of desirable properties. The protocol is described in Algorithm 1:

Algorithm 1 Negotiation protocol for CRG bargaining.

```
Choose an ordering $\langle a'_1, \ldots, a'_n\rangle$ of $Ag$.
{Negotiation stage starts}
$t := 0$
repeat
  $accept$ := true
  $t := t + 1$;
  $a'_t$ proposes a cooperation structure, $\lambda_t(t) = \langle C, G', \xi\rangle$ with $C = \{a'_t, a'_{t+1}, \ldots, a'_n\}$
  $i := t$
  repeat
    $i := i + 1$
    if $a'_i$ rejects $\lambda_t(t)$ then
      $accept$ := false
      $a'_t$ forms the singleton coalition $\{a'_t\}$ in the final coalition structure.
    else
      {$a'_i$ accepts the offer} $a'_i$ accepts the offer and $\lambda_t(i)$ is set to $\lambda_t(i-1)$.
    else
      {$a'_i$ makes a counteroffer}
      $a'_i$ proposes $\lambda_t(i)$, a proposal that must satisfy $\bigwedge_{j=t}^{i-1} [u_j(\lambda_t(i)) \geqslant u_j(\lambda_t(j))]$
      {i.e., The proposal made by $a'_i$ cannot decrease the utility that an agent $a'_j$, preceding $a'_i$ in the ordering obtains from its earlier proposal $\lambda_j(t)$ in
      round $t$.
      }
    end if
  until (not $accept$) or ($i = n$)
until ($t = n$) or ($accept$)
The coalition structure consists of $\lambda_t(n)$ and the singleton cooperation structures for each $a'_i$, $0 \leqslant i < t$.
```

The intuition behind this protocol is that it balances the power given to the proposers and the responders:

- The responders can reject the offer, in which case the proposer gets nothing from negotiation. Thus, the proposer must provide them with "beneficial" offers — if it makes them a poor offer, they can reject it, forcing the proposer to leave negotiation, and forego the benefits of cooperation.
- However, the proposer has the power to choose *which* offer to make: between all of the beneficial offers that it could make, it can choose its most preferred one.

This protocol is a modification of the protocol proposed by Hart and Mas-Colell [16]. The modifications were motivated by two goals. Our first goal was to design a deterministic protocol. This makes it easier to compute the equilibrium strategies. In Hart and Mas-Colell's protocol if a proposal is rejected then with probability ρ the proposer continues to be active and continues to participate in the negotiation process and with probability $1 - \rho$ the proposer drops out of the negotiation and gets a final payoff of 0. In our protocol, the proposer of a rejected proposal *always* drops out of the negotiation. Furthermore, in the Hart and Mas-Colell protocol, the proposer is chosen at random out of the active set at the beginning of each round of the negotiation, while in our protocol the order of the agents is determined before the beginning of the negotiation. Thus, once the order of agents is chosen, our protocol is deterministic.

Our second goal was to direct the agents to a Pareto optimal solution. The Hart and Mas-Colell protocol directs the agents to almost Pareto optimal solutions, but with the changes above, even this weaker property was not true anymore. Thus, while in the Mas-Colell protocol the responder can either accept or reject the offer, in our protocol we allow the responders to improve upon previous proposed cooperation structures, given that the other agents will not suffer from such an improvement.

There are, however, a number of questions to be dealt with in order to fully analyze the approach described in Algorithm 1. For example: How does the agent, a'_t determine the exact form of the cooperation structure, $\lambda_t(t)$, that it will propose in round t? How do the agents a'_i (for $i > t$) determine their responses to the current proposal and the form that counterproposals, $\lambda_t(i)$ should take? And so on. Recalling that each agent is seeking to realise one of its desired goals and maximize its utility in the light of other agents having similar aims, we shall see that such issues can be considered by formulating a system of inequalities, solutions to which not only respect the constraints set by the negotiation protocol just described but also provide a means for establishing important equilibrium properties of the strategies defined from these solutions. Before considering the form and properties of these systems of inequalities, we first review the concept of "strategic equilibrium" to be used.

5.2. Subgame perfect equilibrium

In the following we use the concepts of Nash equilibrium and subgame perfect equilibrium in order to analyze negotiation strategies. These are standard concepts in game theory, and we will not give a detailed presentation: see, e.g., [25].

A *strategy* for an agent is simply a function from the history of negotiation to the choices available to that agent in the current state of negotiation. Thus, a strategy is simply a rule telling the agent how to negotiate. A *strategy profile* is a sequence of strategies, one for each agent.

Now, Nash equilibrium is the most commonly used solution concept in game theory [25, p. 14]. This notion defines a stable state of a game, but does not attempt to examine the process by which this state is reached.

Definition 1 *(Nash equilibrium).* A strategy profile $(f_1, f_2, \ldots, f_n)$ is a Nash equilibrium if each agent i does not have a different strategy yielding an outcome that it prefers to that generated when it chooses f_i, given that the other players follow their profile strategies.

This means that if all agents use the strategies specified in the strategy profile of the Nash equilibrium, then no agent is motivated to deviate and use another strategy. However, the use of the Nash equilibrium is not an effective way to analyze the outcomes of our negotiation model, since it evaluates the desirability of a strategy only from the viewpoint of the agents only at the start of negotiation. In view of the fact that in our negotiation model agents know the history until their move, there may be some point in the negotiation where one or more agents prefer to diverge from their Nash equilibrium strategies. That is, Nash equilibrium strategies may be in equilibrium only in the first step of the negotiation, but may be unstable in subsequent stages. Furthermore, there can be many (even infinitely many), Nash equilibria.

Motivated by these arguments, we use the concept of *subgame perfect equilibrium* [25, p. 97], which is a stronger concept, and will be used in order to analyze the negotiation.

Definition 2 *(Subgame perfect equilibrium).* A strategy profile is a subgame perfect equilibrium if the strategy profile induced in every subgame is Nash equilibrium of that subgame.

This means that in any step of the negotiation process, no matter what the history is, no agent is motivated to deviate and use another strategy other than that defined in the strategy profile. Once the order $a'_1, \ldots, a'_n$ of the agents is fixed, we can compute the strategies that are in subgame perfect equilibrium using backward induction [25, p. 99].

Having reviewed the formulation of equilibrium we can now return to the questions raised following the description of the negotiation protocol in Algorithm 1, i.e., the methods by which agents decide the form of cooperation structures put forward in the course of negotiation.

5.3. Determining proposals by integer linear programming

Suppose, having fixed an ordering $\langle a'_1, a'_2, \ldots, a'_n\rangle$ of the agents, negotiation has reached round t and it is the turn of agent a'_i (where $t \leqslant i \leqslant n$) to contribute, i.e., to formulate the cooperation structure $\lambda_t(t)$ (for $t = i$) or to make a counterproposal $\lambda_t(i)$ or to reject. We denote by $u_i(\{i\})$ the maximal utility agent $i \in Ag$ can obtain from its singleton cooperation structure, i.e., $u_i(\{i\}) = \max\{u_i(\lambda) \mid \lambda = \langle\{i\}, G', \xi\rangle\}$.

We denote by $u(i,t)$ the utility that an agent a'_i obtains from its proposal at round t, so that $u(i,t) = u_i(\lambda_t(i))$. Agent a'_i is seeking to maximize this utility, but does not have complete freedom (within the protocol defined by Algorithm 1) simply to propose a structure which is solely in its own interest: to do so, when $i = t$, might lead to agents a'_j rejecting its proposal outright ($j > i$); and, when $i > t$, the protocol described in Algorithm 1 does not allow $\lambda_t(i)$ to reduce the utility according to $\lambda_t(j)$ of the agents that are ordered before it (i.e., for $t \leqslant j < i$).

The constraints governing the form that proposed cooperation structures can take may be described by considering a system of inequalities (over three disjoint sets of variables) in which the integer values assigned to these variables in any solution exactly describe the agents (C), realized goals (G') and contributions (ξ) featuring in a cooperation structure.

Thus we have the following variables featuring in the system describing those proposals, $\lambda_t(i)$, that could be made by a'_i in round t:

$$Y = \{y_l\colon \text{ for each } g_l \in G\}$$
$$S = \{s_j\colon \text{ for each } a'_j \in \{a'_t, a'_{t+1}, \ldots, a'_n\}\}$$
$$X = \{x_{jr}\colon \text{ for each } a'_j \in \{a'_t, a'_{t+1}, \ldots, a'_n\} \text{ and } r \in R\}.$$

Informally these play the following roles:

- The variable $y_l \in Y$ will take the value 1 if the corresponding goal g_l is among those realized in G' of the cooperation structure, $\lambda_t(i)$ achieving utility $u(i,t)$ proposed by a'_i in round t; otherwise $y_l = 0$.
- The variable $s_j \in S$ will take the value 1 if (at least) one of the goals for the agent $a'_j \in \{a'_t, \ldots, a'_n\}$ is realized in the cooperation structure, $\lambda_t(i)$ achieving utility $u(i,t)$ proposed by a'_i in round t; otherwise $s_j = 0$.
- The variable x_{jr} describes the amount of resource $r \in R$ contributed by a'_j within the cooperation structure, $\lambda_t(i)$ achieving utility $u(i,t)$ proposed by a'_i in round t; x_{jr} is an integer value with $0 \leqslant x_{jr} \leqslant \mathbf{en}(a'_j, r)$.

We recall that

$$V = \sum_{i \in Ag} \left(\max_{g \in G_i} \sum_{r \in R} \mathbf{req}(g, r) \right)$$

is the maximum possible cost that needs to be expended in order for every agent to be able to achieve at least one of its goals. Given that a'_i seeks to maximize its own utility in round t, the optimal proposal (from a'_i's viewpoint) is that which would maximize $u(i,t)$, i.e., solves

$$\text{maximize } s_i V - \sum_{r \in R} x_{ir}. \tag{1}$$

As we have already noted, however, the protocol described in Algorithm 1, does not allow a'_i total latitude in constructing its proposal: the maximization problem specified in Eq. (1) must be solved subject to a number of *constraints*, captured in terms of the following six groups of inequalities.

$$\sum_{g_l \in G_j} y_l \geqslant s_j \quad \forall t \leqslant j \leqslant n. \tag{2}$$

Eq. (2) will be satisfied by instantiations of $\langle Y, S, X \rangle$ for which $s_j = 1$ *only* if some goal acceptable to a'_j is amongst those realized in the cooperation structure $\lambda_t(i)$ proposed by a'_i in round t, i.e., in proposing a cooperation structure, $\lambda_t(i) = \langle C, G', \xi \rangle$, a'_i can rely on the assistance of a'_j ($a'_j \in C$) in achieving the utility $u(i,t)$ only if, in return, a'_j benefits by having one of its goals realized.

$$s_j V - \sum_{r \in R} x_{jr} \geqslant u(j,t) \quad \forall t \leqslant j \leqslant i-1. \tag{3}$$

Eq. (3) represents the constraint specified in the protocol described in Algorithm 1 that any proposal $\lambda_t(i)$ by a'_i must provide all of those agents preceding a'_i in round t – the agents $\{a'_t, a'_{t+1}, \ldots, a'_{i-1}\}$ – with at least the utility each can already obtain, i.e., a'_j in round t may achieve $u(j,t)$ by virtue of the cooperation structure $\lambda_t(j)$ proposed earlier in the round. Hence the left-hand side of (3) describes the utility, $u_j(\lambda_t(i))$, that a'_j accrues from the cooperation structure proposed by a'_i in round t; the right-hand side of (3) defines the utility that a'_j obtains from its (earlier) proposed cooperation structure in round t, i.e., $\lambda_t(j)$.

$$s_i V - \sum_{r \in R} x_{ir} \geqslant \begin{cases} u_i(\{i\}) & i = t \\ u(i, t+1) & \text{otherwise.} \end{cases} \tag{4}$$

Whereas (3) is concerned with conditions imposed on $\lambda_t(i)$ arising from the expectations of agents other than a'_i, i.e., those in $\{a'_t, \ldots, a'_{i-1}\}$, (4) sets constraints on utility obtainable by a'_i irrespective of the actions of agents. Noting that a'_i can always achieve the utility afforded by forming a singleton coalition, if a'_i is the first agent to make a proposal in round t (so that $i = t$), any cooperation structure, $\lambda_t(i)$ proposed must (in a'_i's own interests) be as beneficial to a'_i as could be achieved by a'_i forming a singleton coalition: the $i = t$ case on the RHS of (4) captures this constraint. Alternatively, if a'_i is *not* the first agent to bid in round t ($i \neq t$), then (noting that $i > t$ in this case), it is certainly the case that any proposal $\lambda_t(i)$ made by a'_i must obtain at least the utility for a'_i that could be realized in a later round of negotiation, i.e., the value $u(i, t+1)$.

$$s_j V - \sum_{r \in R} x_{jr} \geqslant u(j, t+1) \quad \forall i+1 \leqslant j \leqslant n. \tag{5}$$

The conditions captured in (5) indicate that the utility gained by a'_j ($i < j \leqslant n$) from any proposal, $\lambda_t(i)$ made by a'_i in round t, should be at least as great as that which a'_j could realise from the proposal that a'_j would be able to make in a later round: were this not the case, a'_j would either reject the proposal $\lambda_t(i)$ or make a counterproposal obtaining utility at least $u(j, t+1)$ for itself in a later round:

$$0 \leqslant x_{jr} \leqslant \mathbf{en}(j, r) \quad \forall t \leqslant j \leqslant n, r \in R \tag{6}$$

$$\sum_{t \leqslant j \leqslant n} x_{jr} \geqslant \sum_{g_l \in G} y_l \, \mathbf{req}(l, r) \quad \forall r \in R. \tag{7}$$

The constraints specified in (6), (7) ensure the cooperation structure $\lambda_t(i)$ is feasible (regardless of whether any agent adopts it). For each agent a'_j and resource $r \in R$, the proposal made by a'_i allows a'_j to expend at most its endowment of the resource r (6); and, (7) admits only cooperation structures $\lambda_t(i) = \langle C, G', \xi \rangle$ for which each goal g_l included in G' (so that $y_l = 1$) has sufficient of each resource provided in ξ ($\sum_{j=t}^{n} x_{jr}$) to meet its requirement $\mathbf{req}(g_l, r)$.

We denote by $LP(i, t)$ the space of instantiations π of $\langle Y, S, X \rangle \in \mathbb{Z}^{|G|+|X|+n-t+1}$ for which

- $\pi \in LP(i, t)$ maximizes $u(i, t)$, as described in (1).
- $\pi \in LP(i, t)$ satisfies each of the inequalities specified in (2)–(7).

We refer to such instantiations subsequently as *solutions of* $LP(i, t)$. We note that $LP(i, t)$ may be empty (no cooperation structure $\lambda_t(i)$ can be proposed by a'_i in round t that satisfies the conditions imposed by the inequalities (2)–(7)), or $LP(i, t)$ may have several solutions.[3] In this latter case it is unimportant which solution of $LP(i, t)$ is chosen by a'_i in round t: given the formulation of (1)–(7), a'_i will achieve the same utility — $u(i, t)$ — for each.

Given a solution of $LP(i, t)$, let $\lambda_t(i) = \langle C, G', \xi \rangle$ so that:

- $C = \{a'_t, \ldots, a'_n\}$,
- $G' = \{g_l\colon\ g_l \in G_j \text{ and } y_l = 1\}$, and
- $\xi_j = \langle x_{j1}, \ldots, x_{j|R|} \rangle$, $t \leqslant j \leqslant n$.

We observe that from the instantiation of $\langle Y, S, X \rangle$ defining a solution of $LP(i, t)$ the exact form of the cooperation structure $\lambda_t(i)$ is easily determined. Thus from a solution of $LP(i, t)$ and its associated cooperation structure $\lambda_t(i) = \langle C, G', \xi \rangle$, the utility an agent i obtains from its proposed cooperation structure at round t is $u(i, t) = s_i V - \sum_{r \in R} x_{ir}$.

We wish to use the protocol of Algorithm 1 together with the system of inequalities (1)–(7) to determine $\lambda_t(n)$ the cooperation structure agreed when protocol described in Algorithm 1 ends. Specifically, we wish to treat the maximization of (1) subject to the inequalities (2)–(7) as an integer linear program. One problem is, however, apparent with these conditions. The values V, $\mathbf{en}(a, r)$, and $\mathbf{req}(g, r)$ are already known as part of the CRG instantiation and we wish to find solutions (values for $\langle Y, S, X \rangle$) of $LP(i, t)$. The problem is that the solutions of $LP(i, t)$ are given in terms of solutions of $L(j, t)$ where $t \leqslant j < i$ so we need to determine the correct values for $u(j, t)$ to use in Eqs. (3), (4) and (5). In other words, we cannot find the value of $u(i, t)$ until we have calculated the value of each $u(j, t)$ for $t \leqslant j < i$.

Assume that the current round is t and it is the turn of agent a'_i to propose a cooperation structure. The value of $u(j, t)$ to be used in finding a solution of $LP(i, t)$ for each agent j preceding i according to the predefined order, i.e., those for which $t \leqslant j \leqslant i-1$, can be calculated easily from the proposal agent a'_{ji-1} has already made. That is, $u(j, t)$ is determined by the cooperation structure that was proposed by a'_j at round t and does not require obtaining solutions of $LP(j, t)$: such will already have been found and these values can be substituted for use in (3).

In order to compute the value $u(j, t+1)$, for $i \leqslant j \leqslant n$, to be used in finding a solution of $LP(i, t)$, however, will require computing the values of $u(j, t')$ $\forall t' > t$ and $\forall t' \leqslant j \leqslant n$, i.e., solutions of $LP(i, t)$ are specified in terms of solutions of $LP(j, t')$ for every $t' > t$ and $t' \leqslant j \leqslant n$.

We resolve this difficulty as follows: to compute the required values the agent uses a backward induction starting from computing $u(n, n)$ by finding a solution of $LP(n, n)$ which will lead to $u_n(\{n\})$. Then the agent computes $u(n-1, n-1)$ and $u(n, n-1)$ by finding solutions of $LP(n-1, n-1)$ and $LP(n, n-1)$. In general, given a round $t' > t$ the agent computes

[3] Although, only a *finite* number of solutions are possible, since there are a finite number of possible cooperation structures.

$u(j,t')$ by finding a solution of $LP(j,t')$ according to the agents predefined order (i.e., it starts solving $u(j,t')$ for $j=t'$ till $j=n$). Note that when the agent calculates $u(j,t')$ the values of $u(k,t')$ are already been calculated for $t' \leqslant k \leqslant j-1$ and the values of $u(k,t'+1)$ are already computed for $t'+1 \leqslant k \leqslant n$. It is easy to see, by backward induction, that $LP(t,t)$ will always have a solution, i.e., $LP(t,t)$ computed in that way is always non-empty. These processes are formally presented in the following proposition.

Proposition 14. *Given an order of the agents $(a'_1,\dots,a'_n)$, $a'_j \in Ag$, the following strategies are in subgame perfect equilibrium:*
For any round $1 \leqslant t < n$,

- *Agent a'_t offers $\lambda_t(t)$.*
- *Agent a'_j, $t < j \leqslant n$ in its turn at round t will do:*
  Given the cooperation structures $\lambda_t(l)$ that have already been proposed by a'_l, $t \leqslant l < j$, a'_j will compute $u(l,t)$. Then, a solution of $LP(j,t)$ will be computed as described above. If $LP(j,t)$ has no feasible solution, a'_j will "reject" the offer.
  Otherwise if $\lambda_t(j)=\lambda_t(j-1)$ then a'_j will accept, otherwise it will propose $\lambda_t(j)$.

Proof. The proof is by backward induction on the negotiation step (t), and for each step by backward induction according to the agent order.

For $t=n$: a solution of $LP(n,n)$ will yield a'_n a utility of $u_n(\{a'_n\})$ and it cannot do better. Note that there may be several solutions to $LP(n,n)$, but they will all yield the same utility to a'_n.

For $t=n-1$ and $j=n$: If a'_n rejects a'_{n-1}'s offer, the current round would end and a'_n obtain a utility of $u(n,n)=u_n(\{a'_n\})$ in the next round $(t=n)$. So, if the offer a'_{n-1} makes would yield a'_n at least $u_n(\{a'_n\})$, then a'_n cannot gain by deviating. This is stated in constraint (4) on solutions of $LP(n,n-1)$. In addition, suppose the utility for a'_{n-1} from the offer a'_{n-1} made earlier in the round is $u(n-1,n-1)$. According to the protocol a'_n can only make offers that yield a'_{n-1} at least that utility. This is stated in constraint (3) on solutions of $LP(n,n-1)$. Finally, the proposed offer must be feasible which is stated in constraints (6) and (7) on solutions of $LP(n,n-1)$. Among all these possible offers, a'_n should choose a cooperation structure that yields it the highest utility; this is stated by the target function (1). Thus, if there is a solution of $LP(n,n-1)$, if the offer associated with this solution is the same as the offer made by a'_{n-1} then a'_n should accept; if it is not the same it should make the offer associated with this as in both cases it cannot improve by deviating. If such a solution does not exist, a'_n should reject the offer and will get $u(n,n)=u_n(\{a'_n\})$.

For $t=n-1$ and $j=n-1$: a'_{n-1} is making an offer to a'_n. If a'_n rejects the offer, a'_{n-1} is forced to leave the negotiation and, thereby, obtains $u_{n-1}(\{a'_{n-1}\})$. So, a'_{n-1}'s utility from the proposal it makes should be at least not worse than $u_{n-1}(\{a'_{n-1}\})$. This is valid according to constraint (4).

If the offer a'_{n-1} makes yields a'_n at least $u_{n,n}=u_n(\{a'_n\})$, then, again, a'_n cannot gain by deviating. This is stated in constraint (5) on solutions of $LP(n-1,n-1)$. In order for it to be accepted by a'_n, $LP(n,n-1)$ must contain a solution. In particular, these must satisfy constraint (4) on solutions of $LP(n,n-1)$ which is the same as constraint (5) on solutions of $LP(n-1,n-1)$: while $LP(n,n-1)$ may have several solutions, all will be associated with the same $u(n,n-1)$. The feasibility of the proposal is ensured by (6) and (7) on solutions of $LP(n-1,n-1)$. Again among all these possible agreements a'_{n-1} considers the best to itself (either, accept the proposed offer if it is the same as the one a'_{n-1} commuted or proposes this agreement) and thus does not have an incentive to deviate.

Inductive step. Suppose the specified strategies are in subgame perfect equilibrium for $t > k \geqslant n$. We will show it for $t=k$.

First we will consider $j=n$. From the induction hypothesis a'_n will obtain $u(n,k+1)$ which is associated with solutions of $LP(n,k+1)$.

According to constraint (4) on solutions of $LP(n,k)$, a'_n it will obtain at least this amount, so if a solution to $LP(n,k)$ exists there is no incentive to wait to round $k+1$. Similarly to the base case the other constraints are necessary for following the protocol and making sure the proposed cooperation structure is feasible.

Assuming the induction hypothesis is correct for $k < m < j \leqslant n$, consider $j=m-1$. From the inductive hypothesis, if a'_j's offer will be rejected, it will obtain $u(j,k+1)$ associated with solutions of $LP(j,k+1)$ in the next time period. According to constraint (5) on solutions of $LP(j,k)$ a'_j will obtain at least this amount in the proposed cooperation structure. According to the induction hypothesis, a'_j will also receive at least this utility in the proposals made in the rest of the round given that there will be a solution of $LP(l,k)$, $j < l \leqslant n$. To ensure there will be such a solution, by a similar argument to that used in the base case, it must satisfy the constraint (4). In addition, to meet the protocol conditions and find a feasible cooperation structure, $LP(j,k)$ includes the constraints (3), (6) and (7).

Finally, consider $j=k$. Given the induction hypothesis the reasoning is similar to the base case, where $t=n-1$ and $j=n-1$, and a'_j does not have any incentive to deviate. □

If the agents follow the subgame perfect equilibrium strategies, there will be only one round of the negotiation. The first agent, a'_1 will make an offer; the second agent, a'_2 will make a counter offer trying to improve its own utility, while giving a'_1 the same utility as in the original offer, and so on, until the turn of the last agent, a'_n, which will try to improve its own utility while giving the others at least as much as they got in the offer of a'_{n-1}. The cooperation structure that will be implemented will consist of the cooperation structure $\lambda_1(n)$ that will be offered by the last agent in the first round.

5.4. Desirable properties

An obvious question is what outcomes the protocol will lead to, assuming rational action on the part of participant agents; in particular, it would be highly desirable to have a protocol so that, assuming rational play, a "good" outcome will result; so first, we need to consider what we mean by "good" in this context.

First, it seems essential that no agent can *lose* by participating. We capture this in the notion of individual rationality.

Proposition 15 *(Individual rationality). Given a sequence $(a'_1, \ldots, a'_n)$, $u_i(\lambda_1(n)) \geqslant u_i(\{i\})$.*

Proof. The claim is clear for a'_1, since according to constraint (4) on solutions of $LP(1,1)$ its utility associated with the cooperation structure a'_1 proposes is at least $u_1(\{a'_1\})$. According to constraint (3) on solutions of $LP(n,1)$, $u_1(\lambda_1(n)) \geqslant u_1(\{1\})$.

Consider any of the other agents, a'_j, $1 < j \leqslant n$. From constraint (4) on solutions of $LP(j,j)$, $u(j,j) \geqslant u_j(\{j\})$. It is easy to show by backward induction, using constraint (5) that $u(j,t)$ for $1 \leqslant t < j$ are also at least $u_j(\{j\})$. □

An additional important property is Pareto Optimality of the outcome. This property ensures that it is not possible to improve the outcome of one agent without decreasing the outcome of the others. In particular, in our case the cooperation structure that is formed if the agents follow their subgame perfect equilibrium strategies is "efficient" in the sense that there is no other cooperation structures in which at least one agent will obtain a higher utility without the others obtaining lower utility. Note that this property addresses the outcome, not the strategies. It is possible that stable strategies (i.e., strategies that are in equilibrium) may yield non Pareto Optimal outcomes (e.g., the Nash equilibrium strategies in the Prisoner's Dilemma game). We formally prove this property in the next proposition.

Proposition 16 *(Pareto Optimality). Given a sequence $(a'_1, \ldots, a'_n)$, there is no $\lambda = \langle Ag, G', \xi' \rangle$ such that there exists $a_i \in Ag$ such that $u_i(\lambda_1(n)) > u_i(\lambda)$ and for $a_j \in Ag$, $j \neq i$, $u_j(\lambda_1(n) \geqslant u_j(\lambda)$.*

Proof. The proposition is clear from the maximization expression of each $LP(i,1)$, $1 \leqslant i \leqslant n$ and constraint (3) on solutions of $LP(i,1)$, $1 \leqslant i \leqslant n$. Intuitively, each agent tries to maximize its own utility, given the utilities of agents preceding him. □

A common criticism of Pareto Optimality is that it may lead to unjust outcomes. In particular there may be Pareto Optimal outcomes where one of the players obtains a very large utility, while the others do not obtain any utility. However, defining a fair distribution of utility is not easy. The example of one player obtaining all the utility may be considered fair if this agent is the only one that contributed to the creation of this utility. However, such a distribution might be regarded as unfair if all agents contributed equally.

Thus, first, we would like to characterize agents that do not contribute anything to the joint utility of a coalition. We will refer to such an agent as a *dummy* agent.[4] We will require that in a "fair" cooperation structure a dummy agent will not obtain any utility. In order to define the notion of a dummy agent we first recall what it means for a coalition to be successful: A coalition $C \subseteq N$ is successful iff $sf(C) \neq \emptyset$. Intuitively, in a successful coalition at least one goal of each agent can be satisfied.

Definition 3 *(Dummy agents).* Agent $i \in Ag$ is a *dummy* iff for all $C \subseteq N$, $C \cup \{i\}$ is not successful.

It follows that if $i \in Ag$ is a dummy agent then $u_i(\{i\}) = 0$. Furthermore, adding a dummy agent cannot increase the maximal sum of the utilities a group of agents could obtain without it. Given this definition, the "dummy agent" property will state that a dummy agent should not obtain any utility from the coalition structure. In order to show that a cooperation structure that the agents agree upon if they follow their subgame perfect equilibrium strategies in our proposed protocol satisfies the dummy agent property, we will first prove some properties of the resulting structure.

Proposition 17. *Suppose we are given a sequence $(a'_1, \ldots, a'_n)$, and the associated agreement is $\lambda_1(n)$. Let $C \subseteq Ag$ be the set of agents such that for any $a_i \in C$, $u_i(\lambda_1(n)) > 0$. Then C is a successful coalition.*

Proof. From constraint (2) and Proposition 15 an agent a_j that at least one of its goals is satisfied will not contribute any resources, i.e., $x_{lr} = 0$ and its utility will be zero. So, for each agent in C at least one of its goals are satisfied and in constraints (6) and (7), for only $j \in C$ $x_{jr} > 0$. Thus C is successful. □

[4] We borrow this name from the definition of the Shapley value [25, p. 292].

We now prove the dummy player property holds in our framework.

Proposition 18 *(Dummy agents). Suppose we are given a sequence $(a'_1, \ldots, a'_n)$, and the associated agreement is $\lambda_1(n)$. Then if a_i is a dummy agent then $u_i(\lambda_1(n)) = 0$.*

Proof. Let $C \subseteq Ag$ be the set of agents such that for any $a_j \in C$, $u_j(\lambda_1(n)) > 0$. From Proposition 17 C is successful and thus according to the definition of the Dummy agent property, $a_i \notin C$. □

Another notion of fairness is that similar agents will obtain the same utilities when taking part in the same cooperation structure. Similarity is defined in the next definition, in the sense of being able to replace one agent by the other in a cooperation structure, without changing the utility obtained by all agents.

Definition 4 *(Interchangeable agents).* Agents $i, j \in Ag$ are said to be *interchangeable* iff:

1. $u_i(\{i\}) = u_j(\{j\})$;
2. for any set of agents $C \subseteq Ag$, where $i, j \in C$ and for any $\lambda = \langle C, G', \xi \rangle$ there exists $\lambda' = \langle C, G'', \xi' \rangle$ such that:
   (i) $u_i(\lambda) = u_j(\lambda')$, and $u_j(\lambda) = u_i(\lambda')$;
   (ii) For any $l \in C$, $l \neq i$, $l \neq j$ $u_l(\lambda') = u_l(\lambda)$.
3. Suppose $\lambda = \langle C, G', \xi \rangle$ such that $i \notin C$ and $j \notin C$. Then, for any $\lambda^i = \langle C \cup \{i\}, G^i, \xi^i \rangle$ there is $\lambda^j = \langle C \cup \{j\}, G^j, \xi^j \rangle$ s.t.
   (i) $u_i(\lambda^i) = u_j(\lambda^j)$; and
   (ii) for any $l \in C$ $u_l(\lambda^i) = u_l(\lambda^j)$.

It seems fair that the expected utility of two interchangeable agents from the coalition structure formed is the same. Unfortunately, not each scenario of the protocol leads to a cooperation structure that satisfies such a symmetry property, because the order of the agents can make a significant difference. However, if we look at *expected* utility, this property is satisfied. Given a sequence of agents $\varsigma = \{a'_1, \ldots, a'_n\}$, let $\lambda_1^{\varsigma}(n)$ be the associated agreed cooperation structure. We can then define the expected utility EU^i of agent i:

$$EU^i = \sum_{\varsigma=(a'_1,\ldots,a'_n) a'_i \in Ag, a'_i \neq a'_j} \frac{1}{n!} u_i\big(\lambda_1^{\varsigma}(n)\big).$$

Proposition 19 *(Pseudo symmetry). If $i, j \in Ag$ are interchangeable then $EU^i = EU^j$.*

Proof. Consider $a'_1, \ldots, a'_n$ and $a''_1, \ldots, a''_n$ where $a'_k = a_i$, $a'_l = a_j$, $a''_k = a_j$ and $a''_l = a_i$ and for any $1 \leqslant h \leqslant n$ if $a'_h \neq a_i$ and $a'_h \neq a_j$, then $a'_h = a''_h$. Denote by $\lambda_1(n)$ and $\lambda'_1(n)$ the agreements reached when the agents follows the subgame perfect equilibrium strategies of Theorem 14 associated with the first ordering of the agents and the second ordering respectively.

We show by induction on n that $u_i(\lambda_1(n)) = u_j(\lambda'_1(n))$ and $u_j(\lambda_1(n)) = u_i(\lambda'_1(n))$ and for any $h \neq i$ and $h \neq j$, $u_h(\lambda_1(n)) = u_h(\lambda'_1(n))$:

- *Base case*: for $n = 2$ there are only two possible orders $a'_1 = a_1$, and $a'_2 = a_2$ and $a''_1 = a_2$, and $a''_2 = a_1$.
  Let $\lambda_1(1)$ be the agreement proposed by a'_1 which is a solution of $LP(1,1)$ with respect to a'_1, a'_2.
  According to (2) of the definition of the interchangeable agents (Definition 4) there is a cooperation structure $\lambda'_1(1)$ where $u_1(\lambda_1(1)) = u_2(\lambda'_1(1))$ and $u_2(\lambda_1(1)) = u_1(\lambda'_1(1))$.
  Since according to (1) of Definition 4 $u_1(\{a_1\}) = u_2(\{a_2\})$ $\lambda'_1(1)$ is a solution to $LP(1,1)$ associated with a''_1, a''_2. Thus, a'_1 and a''_1 will offer $\lambda_1(1)$ and $\lambda'_1(1)$ respectively.
  Similarly we can show that $u_1(\lambda_1(2)) = u_2(\lambda'_1(2))$ and $u_2(\lambda_1(2)) = u_1(\lambda'_1(2))$.
- *Inductive case*: Suppose the claim is true for $n = m$ and we will show for $n = m + 1$.
  Consider first the case where $a'_1 \neq a_i$, $a'_1 \neq a_j$.
  From the induction hypothesis we have that $u_j(\lambda_2(m+1)) = u_i(\lambda'_2(m+1))$, $u_j(\lambda_2(m+1)) = u_i(\lambda'_2(m+1))$ and for every $h \neq i$ and $h \neq j$ $u_h(\lambda_2(m+1)) = u_h(\lambda'_2(m+1))$.
  Let $\lambda_1(1)$ be the offer made by a'_1. According to the specifications of solutions in $LP(1,1)$ it yields all the other agents at least $u(l, 2)$ (constraint (5)), to a'_1 at least its $u_1(\{a'_1\})$ (constraint (6)), and maximizes its utility given these constraints. From (2) of Definition 4 there exists λ' such that all the agents except i and j gets the same utility as in $\lambda_1(1)$ and $u_i(\lambda_1(1)) = u_j(\lambda')$ and $u_j(\lambda_1(1)) = u_j(\lambda')$. It is easy to see that λ' can be associated with $LP(1,1)$ for the $a''_1, \ldots, a''_n$ order.[5] In a similar way it can be shown that for any $\lambda_1(h)$ associated with the $a'_1, \ldots, a'_n$ there is a $\lambda'_1(h)$ associated with the $a''_1, \ldots, a''_n$ that yields the same utility to all agents but a_i and a_j and $u_i(\lambda_1(h)) = u_j(\lambda'_1(h))$ and $u_j(\lambda_1(h)) = u_i(\lambda'_1(h))$. In particular this is true for $h = m + 1$.

[5] Note that there can be more than one such λ'; however all yield the same utilities to all agents.

Consider the case where $a'_1 = a_i$ and $a'_l = a_j$ and $a''_1 = a_j$ and $a''_l = a_i$. Consider first $\lambda_h(h)$ for $h \geqslant l$. It is clear that since the orders of the agents in both sequences starting in h are the same it will be associated with the same utilities as that associated with $\lambda'_h(h)$.
Suppose $a'_l = a_j$ made an offer $\lambda_l(l)$. According to (3) of Definition 4 there is $\lambda'_l(l)$ in which $u_j(\lambda_l(l)) = u_i(\lambda'_l(l))$ and for any $h \neq i$ and $h \neq j$ $u_h(\lambda_l(l)) = u_l(\lambda'_l(l))$. It is easy to see that $\lambda'_l(l)$ will be offered by $a''_l = a_i$.
Given this it can be shown that for any $h \neq i, h \neq j$ $u_h(\lambda_2(m+1)) = u_h(\lambda'_2(m+1))$ and $u_j(\lambda_2(m+1)) = u_i(\lambda'_2(m+1))$.
The proof now continues as in the previous case.

Given the claim that we proved above and the definition of EU, we can conclude that $EU^i = EU^j$. □

5.5. Time complexity of finding equilibrium strategies

Next we prove that given the integer program problem $LP(j,t')$, $1 \leqslant t' \leqslant n$, $t' \leqslant j \leqslant n$, finding a cooperation structure in which the agents' utilities satisfy the constraints on solutions of $LP(j,t')$ (Eqs. (2)–(7)) and maximizes the objective function (Eq. (1)) is exponential only in the number of goals. Since each agent when its turn arrives to respond to an offer and it needs to determine its strategy (reject the proposal or to offer a cooperation structure) has to solve only a polynomial number of such problems, computing the subgame perfect equilibrium strategy is exponential only in the number of the goals. As mentioned before in many real world situations, the number of goals is relatively small and can be bounded by a small constant number. Thus the complexity of computing the subgame perfect equilibrium strategy can be considered tractable.

Proposition 20 *(Time complexity). Given the integer program problem $LP(j,t')$, $1 \leqslant t' \leqslant n$, $t' \leqslant j \leqslant n$, finding a cooperation structure* (*if any exists*) *that satisfies the constraints on solutions of $LP(j,t')$* (*Eqs.* (2)–(7)) *and maximizes the objective function* (*Eq.* (1)) *is exponential only in the number of goals.*

Proof. See Appendix B. □

5.6. Bargaining solutions and the core

The two types of solution considered in this paper — bargaining solutions and the core — can be understood as representing two camps of game theory researchers, namely noncooperative and cooperative. The most fundamental difference between them is the type of deviation that the two frameworks require their solution concepts to be immune to. It is interesting to ask whether the gap between these two approaches can be closed. In particular, in our context, this means asking whether the coalition structure that is obtained when the agents follow their equilibrium strategies in the negotiation protocol is in the core.

Some previous work has studied the relation of noncooperative models and the core (e.g., [8,27]). However, most previous work focuses on TU-games. Works on NTU coalition games includes [17,23,30]. In [18,19], partition games with externalities are studied. Since our utility function is just a mechanism for expressing the preferences of agents over coalitions, our games have many similarities to hedonic games, where each player's preferences only depend on the coalition they belong to. However, unlike hedonic games, in our model preferences also depend on the goals that a coalition satisfies, and the resources they need must contribute (i.e., not just with respect to coalitions, but with respect to cooperation structures). It turns out the relationships between the core and the outcomes of negotiation can be understood by considering the relationships between them in hedonic games.

We begin by modifying the definition of *top-coalition* of hedonic games [4], and define the concept of a *top-cooperation structure* in a CRG.

Definition 5 *(Top-cooperation structure).* A cooperation structure $\lambda = \langle C, G', \xi\rangle$ is a *top-cooperation structure* if for every agent $i \in C$ and any cooperation structure $\lambda' = \langle C', G'', \xi'\rangle$ such that $i \in C'$ and $\lambda' \neq \lambda$, $u_i(\lambda) > u_i(\lambda')$.

The key point about top-cooperation structures is this: *every participant in a top-cooperation structure strictly prefers this cooperation structure over all others that it could be involved with.* Thus top-cooperation structures imply a certain kind of uniformity for the preference relations of members of the top-cooperation structure. We can show:

Proposition 21. *If a top cooperation structure $\lambda = \langle C, G', \xi\rangle$ exists in a CRG Γ, then:*

1. *λ contains the grand coalition* (*i.e., $C = Ag$*)*.*
2. *λ is the* only *top-cooperation structure in Γ.*
3. *The core of Γ is non-empty, and contains a single coalition structure $\sigma^* = \{\lambda\}$.*

Proof. For item (1), suppose for sake of contradiction that λ is a top-cooperation structure but that $C \neq Ag$. Let $C'' = Ag \setminus C$. Since $C \neq Ag$ then $C'' \neq \emptyset$. Now consider the cooperation structure $\lambda'' = \langle C'', \emptyset, \xi'' \rangle$ in which $\xi''_{i,r} = 0$ for all $i \in C''$ and $r \in R$ (i.e., the agents C'' contribute nothing and achieve no goals). Take $\lambda''' = \lambda \sqcup \lambda''$. Clearly λ''' is feasible, since λ is feasible and λ''' introduces no additional goals. By Proposition 1, for all $i \in C$, we have $u_i(\lambda''') \geqslant u_i(\lambda)$. But then λ cannot be a top-cooperation structure, giving a contradiction. For item (2), suppose for sake of contradiction that there exists more than one top-cooperation structure in Γ, and let λ' be one such top-cooperation structure ($\lambda \neq \lambda'$). From item (1), we know that both λ and λ' will contain the grand coalition. Since λ is a top-cooperation structure, then for all $i \in Ag$, we have $u_i(\lambda) > u_i(\lambda')$. But since λ' is also a top-cooperation structure, then for all $i \in Ag$, we have $u_i(\lambda') > u_i(\lambda)$, giving a contradiction. For item (3), to see that the core is non-empty, simply note that *no* cooperation structure λ' can block $\sigma^* = \{\lambda\}$ since λ contains the grand coalition, and every member of Ag strictly prefers λ over any other cooperation structure of which it could be a member. To see that $\sigma^* = \{\lambda\}$ is the unique member of the core, observe again that every member of λ (i.e., the grand coalition) strictly prefers λ over any other cooperation structure of which it could be a part, and so λ will block any other coalition structure σ'. □

Notice that this implies a kind of efficiency with respect to the core as a solution concept: if there is a top-coalition structure, then this will be in the core. Of course, the notion of a top-cooperation structure is a strong one, and this begs the question of whether top-cooperation structures *ever* exist. The answer is yes, as the following example illustrates:

Example 4. We define a CRG with two agents, each of which shares a common goal, and each of which needs to contribute its entire endowment in order to have the goal accomplished. Formally, let $Ag = \{a_1, a_2\}$, let $G = G_1 = G_2 = \{g\}$, let $R = \{r_1, r_2\}$, and define $\mathbf{req}(g, r_1) = 1$, $\mathbf{req}(g, r_2) = 1$, $\mathbf{en}(a_1, r_1) = 1$, $\mathbf{en}(a_1, r_2) = 0$, $\mathbf{en}(a_2, r_1) = 0$, and $\mathbf{en}(a_2, r_2) = 1$. Let $\lambda = \langle \{a_1, a_2\}, \{g\}, \xi \rangle$ in which $\xi_{a_1,r_1} = 1$, $\xi_{a_1,r_2} = 0$, $\xi_{a_2,r_1} = 0$, and $\xi_{a_2,r_2} = 1$. It is straightforward to see that λ is a top-cooperation structure: the only other feasible cooperation structures achieve no goals, and both agents would strictly prefer λ over these.

However, as the following example illustrates, there are of course CRGs that have no top-cooperation structure:

Example 5. We define a CRG with two agents, each of which has one goal, which can only be accomplished by the other agent contributing its resources. The agents collectively can accomplish both goals, but each would prefer the other to accomplish its goal while making no contribution of its own. Formally, let $Ag = \{a_1, a_2\}$, let $G = \{g_1, g_2\}$, let $G_i = \{g_i\}$, let $R = \{r_1, r_2\}$, and define $\mathbf{req}(g_1, r_1) = 0$, $\mathbf{req}(g_1, r_2) = 1$, $\mathbf{req}(g_2, r_1) = 1$, $\mathbf{req}(g_2, r_2) = 0$, $\mathbf{en}(a_1, r_1) = 1$, $\mathbf{en}(a_1, r_2) = 0$, $\mathbf{en}(a_2, r_1) = 0$, and $\mathbf{en}(a_2, r_2) = 1$. There are only four feasible cooperation structures, none of which is a top-cooperation structure. For example, consider the cooperation structure $\lambda = \langle \{a_1, a_2\}, \{g_1, g_2\}, \xi \rangle$ in which $\xi_{a_1,r_1} = 1$, $\xi_{a_1,r_2} = 0$, $\xi_{a_2,r_1} = 0$, and $\xi_{a_2,r_2} = 1$. It is easy to see that both agents would prefer to contribute nothing and allow the other agent to accomplish their goal.

Now, where top-cooperation structures exist, the core and bargaining solutions coincide:

Proposition 22. *For every CRG that has a top-cooperation structure, the outcome of the negotiation process in which agents follow their subgame perfect equilibrium strategies will be the top-cooperation structure, and thus the outcome of negotiation will be in the core.*

This result is quite intuitive: according to the subgame perfect equilibrium strategies, when making the first proposal, the first agent in the negotiation order a'_1 tries to maximise its own utility while ensuring that other agents do at least as well as they would do had bargaining proceeded to the stage where they were in the position of making a proposal. Agent a'_1 will thus propose the top-cooperation structure, and since *no* agent can improve on this, the proposal will be accepted.

Another obvious question is whether any element of the core can be reached by picking an appropriate negotiation order of the agents. In general, this is not true. Furthermore, there are CRGs in which for any order of the agents the resulting coalition structure is not in the core. The following example illustrates this.

Example 6. Consider the following CRG. We have four agents, $Ag = \{a_1, a_2, a_3, a_2\}$. Intuitively, every pair of agents and each triple can form a successful cooperation structure. A single agent is not successful. Each coalition is associated with a different goal. For the set of goals we have:

$$G = \{g_{12}, g_{23}, g_{13}, g_{14}, g_{24}, g_{34}, g_{12,3}, g_{13,2}, g_{23,1}, g_{12,4}, g_{14,2}, g_{24,1}, g_{13,4}, g_{14,3}, g_{34,1}, g_{2,34}, g_{34,2}, g_{24,3}\}.$$

(The naming convention for goals will become clear below.) For every goal and every agent such that its index appears in the goal name, there is a resource such that its superscript index is the agent's index. For example, for goal g_{12} we have resources r^1_{12} and r^2_{12}, while for goal $g_{12,3}$ we also have resource $r^3_{12,3}$.

The goal sets for each agent are the goals in which its index appears, for example

$$G_1 = \{g_{12}, g_{13}, g_{14}, g_{12,3}, g_{13,2}, g_{23,1}, g_{12,4}, g_{14,2}, g_{24,1}, g_{13,4}, g_{14,3}, g_{34,1}\}.$$

According to the endowment function **en** each agent has 3 units of the resource in which its index appears as the superscript, e.g., agent a_1 has 3 units of the resource r^1_{12}.

The requirement function is as follows. A goal with two indexes, g_{ij}, requires 2 units for each resource with the same indexes, i.e., 2 units of r^i_{ij} and 2 for r^j_{ij}. For example, g_{12} requires 2 units of r^1_{12} and 2 units of r^2_{12}. A goal with three indexes $g_{ij,l}$ requires 1 unit of $r^i_{ij,l}$, 1 unit for $r^j_{ij,l}$ and 3 units for $r^l_{ij,l}$. For example, $g_{12,3}$ requires 1 unit of $r^1_{12,3}$, 1 of $r^1_{12,3}$ and 3 of $r^3_{12,3}$.

The core of this CRG consists of any two cooperation structures of two agents satisfying the associated goal and each agent contributes its relevant required resource. For example, $\sigma = \{\lambda_1, \lambda_2\}$ is in the core where

$$\lambda_1 = \langle \{a_1, a_2\}, \{g_{12}\}, \xi^1 \rangle$$

where according to ξ^1, a_1 contributes 2 units of r^1_{12} and a_2 contributes 2 units of r^2_{12} and

$$\lambda_2 = \langle \{a_3, a_4\}, \{g_{34}\}, \xi^2 \rangle$$

where according to ξ^2, a_3 contributes 2 units of r^3_{34} and a_4 contributes 2 units of r^4_{34}. In addition the grand cooperation structures merging such two coalitions are also in the core.

We now demonstrate that for any order of the agents the result of the negotiation will not be in the core. For example, suppose that the order is a_1, a_2, a_3, a_4. We have $\lambda_4(4) = \{a_4\}$ and $\lambda_3(3) = \langle \{a_3, a_4\}, \{g_{34}\}, \xi^{34} \rangle$ where according to ξ^{34}, a_3 contributes 2 units of r^3_{34} and a_4 contributes 2 units of r^4_{34}. Furthermore, $\lambda_2(2) = \langle \{a_2, a_3, a_4\}, \{g_{34,2}\}, \xi^{22} \rangle$ where according to ξ^{22}, a_3 contributes 1 unit of $r^3_{34,2}$, a_4 contributes $r^4_{34,2}$ and a_2 contributes 3 units of $r^2_{34,2}$. Finally, $\lambda_1(1) = \langle \{a_1, a_2, a_3, a_4\}, \{g_{34,2}\}, \xi^{22} \rangle$ where ξ^{22} is as above and a_1 is not successful. It is easy to see that $\lambda_1(1)$ is not in the core since a_1 and a_2 can block it. Similar observations can be made to any other order.

An obvious question is whether there are other cases in which the result of negotiation is always in the core. As was shown in other papers that study the relationships between the core and the negotiation outcome in NTU games [23] the subgame perfect equilibria of such games strongly depends on the order in which players move. Therefore, these papers have to rely on some mechanism for making players symmetric. We will follow this direction, and define a stronger notion of equilibrium that does not depend on the order of the agents: we adapt the notion of *order independent equilibria* [23] to CRGS.

OIES were introduced in [23] as a way of linking cooperative and non-cooperative games. The starting point was the observation that in a bargaining setting such as our that of [16] (or indeed our bargaining protocol), the outcome available to a player depends crucially on the order in which players are able to make proposals: each different order defines a different game, and so an equlibrium for one order will not necessarily be an equilibrium for another order. OIES were introduced to capture the idea of an equilibrium across *all* possible orders of players. Intuitively, an OIE was defined to be a collection of strategies that forms a subgame perfect equilibrium for every possible ordering of agents, and also, produces an outcome that is independent of the ordering. It was then proved that, if a strategy profile forms an OIE, then the outcome was guaranteed to be in the core [23, Proposition A].

We adapt the concept of OIE for our setting. We first assume that a strategy specifies the action the agent will take at any decision point *for any possible order of players* (and not only for a given order as we discussed above). Given this, we define:

Definition 6 *(OIE).* A profile of extended strategies is an *order independent equilibrium* (OIE) of the negotiation protocol if:

1. the strategy profile is a subgame perfect equilibrium; and
2. the offers an agent will make and accept according to its strategy in all subgames in which the sets of active agents are the same (regardless of the order) are the same.

Now, according to OIE in the first round a given agent will make the same offer and will accept the same offers regardless of the order of the agents. Even though in our model time is not costly we will assume that if an agent can get the same utility for two coalition structures it will prefer the one formed at an earlier time. Thus, we can assume that the negotiation will end at the first round since if the negotiation ends at time t then the disjoint union of the singleton cooperation structures of the first $t-1$ agents and the members of the coalition structures that was form in time t can be achieved in the first round.

Proposition 23. *The outcome of negotiation if the agents follow OIE is in the core.*

Proof. For contradiction, assume that the order of the agents was $a'_1, \ldots, a'_n$ with negotiation ending at the first round with a coalition structure σ, but that σ is not in the core. Thus there is a cooperation structure $\lambda = \langle C, G', \xi \rangle$ that blocks σ. Now, consider a different order of the agents where the members of C are placed at the end of the order. In that order, if the negotiation does not terminate until the round in which only the members of C remain, there will be at least one agent such that its utility from the coalition structure that will form will be at least that obtained from λ. Otherwise λ could be formed. This agent will not accept σ in the first round, since it could obtain a higher utility if negotiation continued. However, since the agents follow OIE it would not accept σ in the first round also when the agents are ordered according to the original order $a'_1, \ldots, a'_n$; a contradiction. $\square$

Note that Proposition 23 does not require that there is a top-cooperation structure. Also note that the converse of Proposition 23 does not hold. That is, there are core outcomes that cannot be reached by any OIE. In [23] it was enough to require that for every subgame of the game the core is not empty. This is not enough for CRGs, as demonstrated by Example 6.

6. Conclusions

Coalitional resource games provide a natural model of scenarios in which groups of agents cooperate by pooling resources to achieve mutually satisfying sets of goals. In this paper, we have considered two types of solutions for such games: stable solutions, in which each agent contributes some resources to the achievement of some goal set, and no coalition has any incentive to deviate from the solution; and bargaining solutions, which satisfy the desirable properties of Pareto Optimality, dummy player, and pseudo-symmetry.

Several issues suggest themselves for future work. First, it would be interesting to look at more sophisticated models of resources and resource consumption. For example, [6] considers processes that have some duration, and that have different resource usage requirements at different stages. Interesting issues arise, for example, when one considers combining such processes. We might think of different goals being achieved by different processes, and then imagine scheduling such processes in such a way as to satisfy agents while staying within stated resource bounds. This also leads us to consider temporal extensions to the present framework, as temporal extensions to qualitative coalitional games were considered in [1]. Finally, of course, other bargaining approaches might be considered: for example there is work in the NTU bargaining literature on bargaining protocols that lead to stable coalition structures (i.e., coalition structures in the core). It would be interesting to consider whether such bargaining protocols might be developed for the present framework.

Acknowledgements

We are extremely grateful to the referees for their detailed and encouraging comments. This research was supported by a Royal Society travel grant, by the EPSRC under project GR/T10657/01 ("Market Based Control of Complex Computational Systems"), by the U.S. Army Research Laboratory and the U.S. Army Research Office under grant number W911NF-08-1-0144, by NSF grant 0705587 and ISF #1357/07.

Appendix A. Notational conventions

Ag	The set of agents (a.k.a. players). Members of the set are typically denoted i, j, and subsets of Ag (*coalitions*) are denoted by C, C', etc.
G	The set of overall possible goals; members are typically denoted by g, g', etc.
R	The set of resources, members denoted r, r', etc.
G_i	The goals of agent $i \in Ag$ ($G_i \subseteq G$).
en	The endowment function of a CRG: $\mathbf{en} : Ag \times R \to \mathbb{N}$. Thus $\mathbf{en}(i, r)$ defines how much of resource r agent i is endowed with.
req	The requirement function of a CRG: $\mathbf{req} : G \times R \to \mathbb{N}$. Thus $\mathbf{req}(g, r)$ denotes the amount of resource r required to achieve goal g.
Γ	A coalitional resource game: $\Gamma = \langle Ag, G, R, G_1, \ldots, G_n, \mathbf{en}, \mathbf{req}\rangle$.
$en(C, R)$	The total amount of resource r that coalition $C \subseteq Ag$ are endowed with.
$req(G', r)$	The total amount of resource r required to achieve goal set $G' \subseteq G$.
$sf(C)$	The set of goals sets that are both feasible for and would satisfy C. If $sf(C) \neq \emptyset$ then we say that C are successful.
ξ	A contribution vector: a C-tuple of resource vectors, the idea being that a contribution vector for C defines for every member i of C how much of every resource i contributes. The notation $\xi_{i,r}$ denotes how much of resource r is contributed by agent i. A feasible contribution vector is one where every agent contributes no more than its endowment of any given resource. We let $c_i(\xi)$ denote the total cost of is contribution in ξ.
λ	A cooperation structure $\lambda = \langle C, G', \xi\rangle$ has the intended meaning that coalition $C \subseteq Ag$ will work together to achieve goals $G' \subseteq G$, with the contribution of each agent being as defined in contribution vector ξ. C_λ, G_λ and ξ_λ denote the coalition, goal set, and contribution vector components of λ, and $c_i(\lambda)$ is shorthand for $c_i(\xi_\lambda)$.
$req(G')$	The total amount of all resources required to achieve goal set G'.
$succ(G')$	The agents that get a goal achieved through G'; $succ(\lambda)$ is a shorthand for $succ(G_\lambda)$.
$\succ_i$	Preference relation for i over cooperation structures: i prefers all cooperation structures in which it achieves its goals over all those in which it does not, and prefers to minimize its resource contribution.
$u_i(\lambda)$	The utility i obtains from cooperation structure λ (will always be positive it i gets its goal achieved through λ).

Table 2
Resource requirements and endowments for the reduction in Proposition 9.

	r_0	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}
$\mathbf{req}(\cdots)$												
g_0	4	0	0	0	0	0	0	k	0	0	k	$2k$
g_1	0	0	$2zk$	0	0	0	$2k$	k	0	0	k	0
g_2	0	0	0	3	0	0	0	0	k	k	$2k$	0
g_3	0	0	0	0	3	0	0	$2k$	k	k	0	0
g_4	0	0	0	0	0	$3zk$	k	0	0	$2k$	0	k
g_5	0	2	0	0	0	0	k	0	$2k$	0	0	k
$\mathbf{en}(\cdots)$												
a_i	0	0	$2z$	0	0	$3z$	2	0	0	0	0	2
a_1	4	0	0	0	3	0	0	$2k$	0	0	$2k$	0
a_2	0	2	0	3	0	0	0	0	$2k$	$2k$	0	0

σ	A coalition structure: a set of cooperation structures, such that the sets of agents in these cooperation structures represent a partition of Ag. $\sigma_\mathbf{0}$ denotes the grand coalition achieves no goals and contributes no resources, and $\lambda_{\sigma,i}$ denotes the cooperation structure in σ of which i is a member.

Appendix B. Proofs

We here present proofs that were omitted from the main text in the interests of readability.

Proposition 9. *Checking whether the core of a* CRG *is non-empty is* NP*-hard.*

Proof. We reduce from SAT [26, p. 171]. An instance of SAT is given by a propositional logic formula Φ over z Boolean variables $x_1, \ldots, x_z$, the aim being to answer "yes" if there is some valuation to the variables under which the formula is satisfied. Without loss of generality, we assume that $z > 1$ and Φ is presented in Conjunctive Normal Form (CNF), i.e., a conjunction of k clauses:

$$\Phi = \bigwedge_{i=1}^{k} \psi_i$$

where ψ_i is a disjunction of literals. We create an instance of the core non-emptiness problem as follows:

- *Possible Goals*: For each literal ℓ occurring in Ψ we create a goal g_ℓ. We also create six additional goals, $g_0, \ldots, g_5$.
- *Agents*: For each clause ψ_i, we create an agent a_{ψ_i}, and in addition create two further agents a_1 and a_2. Let Ag_ψ denote the set of agents corresponding to clauses.
- *Goal sets for agents*: For each agent $a_i \in Ag_\psi$, define the goal set G_{a_i} to be all the goals corresponding to literals in ψ_i, together with goals g_0 and g_5; thus if $\psi_i = \ell_1 \vee \cdots \vee \ell_y$ then $G_{a_i} = \{g_{\ell_1}, \ldots, g_{\ell_y}, g_0, g_5\}$. For agent a_1 we define $G_{a_1} = \{g_1, g_2\}$, while for a_2 we define $G_{a_2} = \{g_3, g_4\}$.
- *Resources*: For each propositional variable x, we create a resource r_x. We then create twelve additional resources, r_0 to r_{11}.
- *Requirements*: For each goal g corresponding to a variable, and resource r_x, we define the quantity of r_x required for g as follows:

$$\mathbf{req}(g, r_x) = \begin{cases} k & \text{if } g = g_x \text{ or } g = g_{\neg x} \\ 0 & \text{otherwise.} \end{cases}$$

  Requirements for $\{g_0, g_1, g_2, g_3, g_4, g_5\}$ are given in Table 2. (Recall that z is the number of Boolean variables in the SAT input formula, while k is the number of clauses.)
- *Endowments*: For each agent a_i corresponding to a clause ψ_i, and for each resource r_x corresponding to a variable x, define the endowment function so that $\mathbf{en}(a_i, r) = 1$. For resources $r_0, \ldots, r_{11}$, endowments are defined in Table 2.

We prove that the core of Γ_Φ is non-empty if and only if Φ is satisfiable.

($\Rightarrow$) Assume Φ is not satisfiable. We prove this implies the core is empty. First, observe that in this case we can in effect treat the clause agents as a single agent, since the clause agents alone do not form a successful coalition — they can only be satisfied by satisfying either g_0 or g_5.

So, consider the possible cases:

1. Observe that $\sigma_\mathbf{0}$ is not in the core, since no agent would have goals achieved by this coalition structure; and yet there are successful coalitions, which would block $\sigma_\mathbf{0}$, since they would rather have their goals achieved than otherwise.

2. Consider the grand coalition. The grand coalition is unsuccessful. For suppose the contrary is true, and note the following property:

   Consider any pair of distinct goals $P = \{g_i, g_j\}$ from $\{g_0, g_1, g_2, g_3, g_4, g_5\}$ then P can be achieved by the grand coalition $\{Ag_\psi, a_1, a_2\}$ if and only if P is one of $\{\{g_0, g_1\}, \{g_2, g_3\}, \{g_4, g_5\}\}$; in all other cases there is some resource r_i, $6 \leqslant i \leqslant 11$ requiring $> 2k$ to be expended, and the total endowment of the grand coalition, for any resource r_i, $6 \leqslant i \leqslant 11$, is exactly $2k$.

   Since Ag_ψ cannot achieve the literal goals (from unsatisfiability) one of $\{g_0, g_5\}$ must be in G'; for a_1 to be satisfied one of $\{g_1, g_2\}$ is in G'; and for a_2 to be satisfied one of $\{g_3, g_4\}$ must be in G'. From the property we just stated, we have a contradiction, so the grand coalition is unsuccessful
3. Consider coalition $Ag_\psi \cup \{a_1\}$ achieving goals $\{g_0, g_1\}$ at cost $4k + 4$ to a_1 (contributing its entire endowment of $\{r_0, r_7, r_{10}\}$) and cost $2z + 4$ to each clause agent (contributing its entire endowment of $\{r_2, r_6, r_{11}\}$). This cooperation structure would block the coalition structure in (1). The corresponding coalition structure would involve a_2 being unsuccessful.
4. Consider coalition $\{a_1, a_2\}$ achieving goals $\{g_2, g_3\}$ at cost $4k + 3$ to each agent: a_1 contributes its endowment of $\{r_4, r_7, r_{10}\}$ while a_2 contributes its endowment of $\{r_3, r_8, r_9\}$. This cooperation structure would block the coalition structure in (3), since the cost to a_1 is smaller and a_2 is satisfied. The corresponding coalition structure would involve Ag_ψ being unsuccessful.
5. Consider coalition $Ag_\psi \cup \{a_2\}$. This coalition could achieve goals $\{g_4, g_5\}$ at cost $4k + 2$ to a_2 (contributing its endowment of $\{r_1, r_8, r_9\}$) and $3z + 4$ to each clause agent, each of which commit their endowment of $\{r_5, r_6, r_{11}\}$. This would block the coalition structure in (4), since a_2 would incur a lower cost while the clause agents would get their goals achieved. The corresponding coalition structure would involve a_1 being unsuccessful.
6. Finally, notice that the coalition structure in (3) would block the coalition structure in (5), since in the coalition structure in (3), Ag_ψ would incur lower cost than in (5), while agent a_1 would have a goal satisfied.

In sum, if Φ is unsatisfiable, then every coalition structure in Γ_Φ is blocked, and so the core of Γ_Φ is empty.

($\Leftarrow$) Assume Φ is satisfiable. The reader may verify that the coalition Ag_ψ of clause agents is successful in this case, at a cost to each clause agent of at most z. In this case, the core of Γ_Φ will contain a coalition structure in which:
- clause agents Ag_ψ achieve a set of goals corresponding to a valuation for Φ; and
- agents a_1 and a_2 cooperate to achieve goals g_2 and g_3, at cost $4k + 3$ to each.

To see that such a coalition structure is stable, observe that the clause agents cannot achieve their goals any more cheaply; they will not cooperate with a_1 or a_2 since to do so would incur much greater cost, and so the only way that a_1 and a_2 can achieve their goals would be to work together. □

Proposition 11. *Checking whether the core of a* CRG *is strongly non-empty is* NP*-hard.*

Proof. We reduce from SAT [26, p. 171]. An instance of SAT is given by a propositional logic formula Φ, the aim being to answer "yes" if there is some valuation to the Boolean variables $x_1, \ldots, x_y$ that satisfies the formula. Without loss of generality, we assume that Φ is presented in Conjunctive Normal Form (CNF), i.e., a conjunction of k clauses:

$$\Phi = \bigwedge_{i=1}^{k} \psi_i$$

where ψ_i is a disjunction of literals. We create an instance of the core non-emptiness problem as follows.

- For each literal ℓ occurring in Ψ we create a goal g_ℓ.
- For each clause ψ_i, we create an agent a_{ψ_i}, and define G_i to be the set of goals corresponding to the literals that occur in ψ_i. That is, if $\psi_i = \ell_1 \vee \cdots \vee \ell_k$, then we define $G_i = \{g_{\ell_1}, \ldots, g_{\ell_k}\}$. Intuitively, each clause agent wants one of its literals to be satisfied.
- For each propositional variable x, we create a resource r_x.
- For each goal g and resource r_x, we define the quantity of r_x required for g as follows:

$$\mathbf{req}(g, r_x) = \begin{cases} k & \text{if } g = g_x \text{ or } g = g_{\neg x} \\ 0 & \text{otherwise.} \end{cases}$$

- For each agent a_i define its endowment function so that for all resources r, we have $\mathbf{en}(a_i, r) = 1$.

We prove that the core of Γ_Φ is strongly non-empty if and only if Φ is satisfiable:

($\Rightarrow$) Assume the core of Γ_Φ is strongly non-empty, so the core contains some coalition structure in which some agents expend some resources and achieve some goals. It is immediate by construction that this coalition structure must involve the grand coalition being successful; since the requirement of any resource r_x for any goal $g_{(\neg)x}$ is k (i.e., the number of agents in the system) and each agent has exactly 1 unit of resource r_x, to satisfy any agent's goal would require the cooperation of the grand coalition, and moreover it would only be possible to satisfy either g_x or $g_{\neg x}$, not both. So, if any agent was unsatisfied in the coalition structure in the strongly non-empty core, it would object, by doing nothing and achieving non of its goals. Finally, observe that if the core is strongly non-empty, then we can straightforwardly extract a valuation for Φ from any coalition structure in the core: the goals achieved will tell us which variables to make true in a satisfying assignment, and the consistency of the valuation follows from the arguments above.

($\Leftarrow$) Assume Φ is satisfiable, and let $\zeta \subseteq \{x_1, \ldots, x_y\}$ be a satisfying valuation. Construct a coalition structure $\sigma^* = \{\langle C, G', \xi\rangle\}$ from ζ as follows: C is the grand coalition; in ξ every agent contributes all its resources; and define G' by:

$$G' = \{g_x\colon x \in \zeta\} \cup \{g_{\neg x}\colon x \notin \zeta\}.$$

We claim σ^* is in the core of Γ_Φ. First, observe that every agent is satisfied in this coalition structure: every clause agent has one of its goals satisfied since every clause in Φ must be satisfied by the valuation ζ. So, suppose for sake of contradiction that σ^* is not in the core. Then there must be an objection to it. Since every agent has their goal satisfied, the only objection could be from some coalition that could have their goals satisfied at reduced cost. But by construction, every goal set corresponding to a satisfying assignment would have equal cost, (every agent would have to contribute their entire endowment) which gives a contradiction. So σ^* is in the core, and hence the core is strongly non-empty. □

Proposition 12. *The problem of computing a preferred contribution vector can be solved in polynomial time.*

Proof. Given a set of goals G' and a corresponding coalition C', the algorithm's target is to find a contribution vector ξ that blocks σ, if it exists. The overall algorithm for this is given in Fig. 4. The algorithm makes use of *flow network* construction, as follows.

First, some notation. Given a set of goals G' we use $res(G')$ to denote the set of resources that are required for this set of goals, $res(G') = \{r\colon\ r \in R, \exists g \in G \text{ s.t. } req(g, r) > 0\}$. We use $en(i)$ to denote the overall quantity of resources that the agent is endowed with, $en(i) = \sum_{r\in R} \mathbf{en}(i, r)$.

Now, we use $\max c(i)$ to denote the agent's maximum cost in ξ in a way that $\langle C', G', \xi\rangle$ can still block σ. In other words, given G' and C' the only cooperation structures $\lambda = \langle C', G', \xi\rangle$ that are candidates for blocking σ are those in which $\forall i \in C'$, $c_i(\xi) \leqslant \max c(i)$. In order to determine the value of $\max c(i)$ we consider the two following cases:

1. The agent would have its goals satisfied by σ. Since the agent would have its goals satisfied both in λ and σ, the agent's cost in λ must be strictly less than in σ. Therefore $\max c(i) = c_i(\lambda_{\sigma,i}) - 1$.
2. The agent would not have its goals satisfied by σ. Since the agent would have its goals satisfied in λ but not in σ, the agent's cost in λ can be even equal the total amount of resource that agent $i \in Ag$ is endowed with, $\max c(i) = \mathbf{en}(i)$. This is because an agent prefers all those cooperation structures which would result in one of its goals being satisfied over all those where it is not.

We thus define:

$$\max c(i) = \begin{cases} c_i(\lambda_{\sigma,i}) - 1 & i \in succ(\lambda_{\sigma,i}) \\ en(i) & \text{otherwise.} \end{cases}$$

In order to find such ξ that $\forall i \in C'$, $ci(\xi) \leqslant \max c(i)$ and $\forall r \in R$, $\sum_{i\in C'} \xi_{i,r} \geqslant req(G', r)$, the algorithm constructs a flow network as described bellow.

Given a set of goals $G' \in 2^G$ and its corresponding set of agents C', we construct the corresponding network $G_f = (V, E)$ as follows:

1. Set of vertices: $V = \{V_A \cup V_R \cup \{s, t\}\}$ where V_A represents all agents in C' and V_R corresponds to the resources in $res(G')$. In addition we let the source s and sink t to be two additional new vertices.
2. Set of edges: a directed edge connects between s and each vertex in V_A and between each vertex in V_R and t. In addition a directed edge connect between $v_i \in V_A$ and $v_j \in V_R$ if $en(i, r_j) > 0$.

$$E = \big\{(s, v_i) \mid v_i \in V_A\big\} \cup \big\{(v_j, t) \mid v_j \in V_R\big\} \cup \big\{(v_i, v_j) \mid v_i \in V_A, v_j \in V_R, en(i, r_j) > 0\big\}.$$

3. Capacity function: for all $v_i \in V_A$ we set $c(s, v_i) = \max c(i)$. For all $v_j \in V_R$ we set $c(v_j, t) = req(G', r_j)$. For each edge that connects between $v_i \in V_A$ and $v_j \in V_R$ we set $c(v_i, v_j) = en(i, r_j)$.

Intuitively:

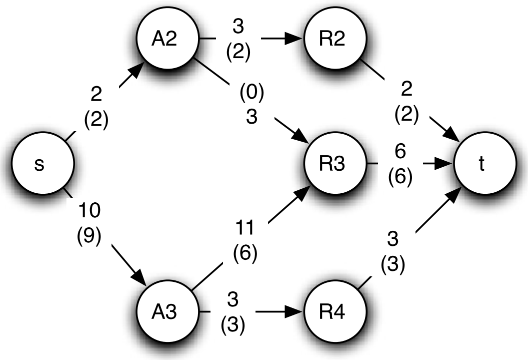

Fig. 3. The corresponding flow network G_f for $G' = \{g_1, g_2\}$.

1. Build the corresponding flow-network G_f.
2. Find the maximum flow $|f|$ in G_f.
3. If $|f| < req(G')$ then announce "no preferred contribution vector exists" and quit.
4. For all agents i and resources r_j such that $(v_i, v_j) \in E$, set $\xi_{i,r_j} = f(v_i, v_j)$, and for all other agents i and resources r_j set $\xi_{i,r_j} = 0$.

Fig. 4. Algorithm for computing a preferred contribution vector, if such exists (Proposition 12).

- The flow on edges from the source s to agents will indicate an agents total contribution to the resource vector; the capacity constraints ensure that the contribution is one that would be strictly preferred by the agent.
- The capacity constraints on edge from an agent to a resource correspond to the endowment of the resource that the agent has — the flow on the edge indicates how much of the resource the agent contributes.
- The flow on edges from resources to the sink node indicate how much of that resource is contributed overall (which is the same as the total requirement for that resource).

Now, we ask whether there is an integer-valued flow in G with value $|f(G)| = req(G')$: if the answer is "yes", then we can "read off" ξ from the flow that we construct. The only problem is that maximum flow algorithm may return non-integer amounts. However the Integrality theorem shows that in case the capacity function takes only integer values if we use Ford–Fulkerson method the maximum flow has the property that $|f|$ is an integer and for all edges the value of $f(u, v)$ is an integer [10]. Thus we can find whether such a ξ exists by running the Edmonds–Karp version of Ford–Fulkerson method [10]. □

Example 7. (Continuation of Example 3.) Fig. 3 shows the flow network built for $\{g_2, g_3\}$. The values of the maximum flow are specified in the brackets. Since the value of the maximum flow ($|f| = 11$) equals $req(G')$, σ can be blocked.

Proposition 13. *The algorithm in Fig. 1 is correct: if it returns "σ is stable", then σ is indeed stable, while if it returns a cooperation structure $\langle C', G', \xi\rangle$, then this cooperation structure indeed blocks σ. Moreover, the algorithm terminates, and runs in time exponential in the number of goals in but polynomial in the number of agents and resources.*

Proof. Given a coalition structure σ, the algorithm first checks that unsuccessful agents in σ have 0 cost, i.e., that if $i \in Ag$ is such that $i \notin succ(\lambda_{\sigma,i})$, then $c_i(\lambda_{\sigma,i}) = 0$. If not, from Proposition 5, σ is not stable and the algorithm terminates.

Next, the algorithm checks for each set of goals G' in 2^G, whether there exists a coalition structure $\lambda = \langle C', G', \xi'\rangle$ that satisfies all goals in G' and blocks the coalition structure σ.

First note that the following agents cannot belong to the coalition C' in the cooperation structure λ that blocks σ:

1. Agents that would not have their goals satisfied by G'. Since these agents are not satisfied, $u_i(\lambda) \leqslant 0$. Thus these agents do not improve their utility from λ over σ.
2. Agents that would have their goals satisfied by σ and have 0 cost, i.e., $i \in succ(\lambda_{\sigma,i})$ & $c_i(\lambda_{\sigma,i}) = 0$. Since these agents obtain their maximum utility from σ $(U_i(\lambda_{\sigma,i}) = V)$, there is no way these agents could improve their utility, and could hence not form part of any blocking structure.

Given this, we define C' to be the set of agents that includes all agents in A_g that can belong to the cooperative structure λ that blocks σ (the agents that do not satisfy 1 and 2 above),

$$C' = \{i\colon\ i \in succ(G')\ \&\ (i \notin succ(\lambda_{\sigma,i}) \text{ or } c_i(\lambda_{\sigma,i}) > 0)\}. \tag{B.1}$$

In fact, it is sufficient to check only for C' if a contribution vector ξ exists such that $\lambda = \langle C', G', \xi \rangle$ blocks σ: there is no need to check for each subset of C', as established in the following lemma. $\square$

Lemma 1. *If $\exists C'' \subseteq C'$ and a contribution vector ξ such that $\lambda' = \langle C'', G', \xi \rangle$ blocks σ, then C' can block σ.*

Proof. Given a coalition $C'' \subseteq C'$ and a contribution vector ξ such that $\lambda' = \langle C'', G', \xi \rangle$ blocks σ. We define the following contribution vector ξ':

$$c_i(\xi') = \begin{cases} c_i(\xi) & \text{if } i \in C'' \\ 0 & \text{otherwise.} \end{cases}$$

Let $\lambda'' = \langle C', G', \xi' \rangle$. Since λ' blocks σ, for each $i \in C''$, $u_i(\lambda'') = u_i(\lambda') > u_i(\lambda_{\sigma,i})$. Moreover, for each $i \notin C''$, $u_i(\lambda'') = V > u_i(\lambda_{\sigma,i})$ (the agent's utility from σ is lower than V, since we do not include agents that satisfy condition 2 in C'). Consequently $\langle C', G', \xi' \rangle$ blocks σ. $\square$

Finally, consider the running time of the algorithm. The complexity of step 2 is exponential in the number of goals in G', simply because it goes through all members of 2^G. However, the complexity of the internal step is polynomial (Proposition 12). Therefore the total complexity of the algorithm is exponential in the number of goals, but polynomial in the number of agents and resources.

Proposition 20. *Given the integer program problem $LP(j, t')$, $1 \leqslant t' \leqslant n$, $t' \leqslant j \leqslant n$, finding the cooperation structure (if one exists) that satisfy the constraints on solutions of $LP(j, t')$ (Eqs. (2)–(7)) and maximizes the objective function (Eq. (1)) is exponential only in the number of goals.*

Proof. Given the integer program problem $LP(j, t')$ (Eqs. (1)–(7)) we assume that all the utilities values of the right side of the constraints are already computed. This is because that when the agent calculates $LP(j, t')$ the values of $u(k, t')$ are already been calculated for $t' \leqslant k \leqslant j - 1$ and the values of $u(k, t' + 1)$ are already computed for $t' + 1 \leqslant k \leqslant n$. Given this in order to prove the above proposition we develop an algorithm that is similar to the algorithm we developed for checking whether σ can be blocked (the algorithm described in Fig. 1 in Section 4.2). Our algorithm goes through the power set of goals 2^G. For each set of goals $G' \in 2^G$, the algorithm finds a cooperation structure $\lambda'_G = \langle \{a'_{t'}, \ldots, a'_n\}, G', \xi' \rangle$ (if exists) that satisfies the constraints of $LP(j, t')$ (given in Eqs. (2)–(7)), satisfies all the goals in G' and maximizes a'_j's utility. After the algorithm computes such a cooperation structure (if exists) for each $G' \in 2^G$, it returns the cooperation structure (of all these computed cooperation structure) that maximizes a'_j's utility. This cooperation structure maximizes the objective function of $LP(j, t')$ (Eq. (1)). If no cooperation structure exists the algorithm returns that no feasible solution exists (in which case the agent rejects the offer – see Proposition 14).

In the description of the algorithm we use the following definition: Given an integer program $LP(j, t')$ and an agent a'_k we use $\lambda_p(k)$ to denote the cooperation structure that determines the minimum utility that a'_j can propose to a'_k at round t' (the utility that is specified in the right side of the constraints of $LP(j, t')$). Namely the utility that a'_k obtains from the cooperation structure that a'_j proposes at round t' should be equal to or higher than the utility that a'_k obtains from $\lambda_p(k)$. In order to determine $\lambda_p(k)$ we consider the following cases:

1. If $k < j$, $\lambda_p(k)$ is the cooperation structure that a'_k proposes at round t'.
2. If $k > j$ or $k = j$ and $j \neq t'$, $\lambda_p(k)$ is the cooperation structure that a'_k proposes at round $t' + 1$ computed by $LP(k, t' + 1)$.
3. If $k = j$ and $j = t'$, $\lambda_p(k)$ is the singleton cooperation structure (that includes only agent a'_k) and maximizes the utility of agent a'_k.

First note that if there exists an agent a'_k, $t' \geqslant k \geqslant n$, which is not satisfied by G' but is satisfied by $\lambda_p(k)$, no cooperation structure that satisfied only the goals in G' can satisfy the constraints given in Eqs. (2)–(7). This is because the maximum utility that a'_k can obtain from any cooperation structure that satisfies only goals in G' cannot be greater than 0 while a'_k's utility from $\lambda_p(k)$ is greater than 0. Therefore the algorithm can ignores G' and continues to consider the next sets of goals in 2^G. Thus we consider G' in which each agent is either satisfied by G' or is not satisfied both by G' and $\lambda_p(k)$. Given such a set of goals G', the integer program problem $LP(j, t')$, and the coalition $\{a'_{t'}, \ldots, a'_n\}$ the algorithm's target is to find a contribution vector ξ (if exists) that satisfies constraints (Eqs. (2)–(7)), satisfies all goals in G' and maximizes a'_j utility. For that purpose we use $\max' c(k)$ (similar to the way we define $\max c(i)$ in Section 4.2) to denote agent a'_k maximum cost in ξ in a way that constraints specified in Eqs. (2)–(7) are still satisfied. In other words the only cooperation structures $\lambda = \langle \{a'_{t'}, \ldots, a'_n\}, G', \xi \rangle$ that can satisfy constraints (Eqs. (2)–(7)) are those in which $\forall k$, $c_k(\xi) \leqslant \max' c(k)$. In order to determine the value of $\max' c(k)$ we consider the following cases:

(1) a'_k would have at least one of its goal satisfied by $\lambda_p(k)$. Since the agent would have its goals satisfied both by G' and $\lambda_p(k)$, the agent's cost in ξ must be less than or equal to its cost in $\lambda_p(k)$. Therefore $\max' c(k) = c_k(\lambda_p(k))$.

```
1. $f_{min} = null$, $G_{min} = null$.
2. For each $G'$ in $2^G$:
  2.1. If $\exists a'_k \notin succ(G')$ and $a'_k \in succ(\lambda_p(k))$ go to 2.
  2.2. Build the corresponding flow-network $G_f$.
  2.3. Find the minimum cost flow in $G_f$.
  2.4. If $f$ exists and one of the following conditions are satisfied:
      (1) $f_{min} = null$.
      (2) $a'_j$ is satisfied by $G'$ but $a'_j$ is not satisfied by $G_{min}$.
      (3) $a'_j$ is satisfied by $G'$ and by $G_{min}$ and cost of $f$ is smaller than cost of $f_{min}$,
      set $f_{min} = f$, $G_{min} = G'$.
3. If $f_{min} = null$ return no feasible solution exists otherwise set $\xi_{k,r_l} = f_{min}(v_k, v_l)$, $\forall (v_k, v_l) \in E$ and $\xi_{k,r_l} = 0$, $\forall (v_k, v_l) \notin E$, return
   $\langle \{a'_{t'}, \dots, a'_n\}, G_{min}, \xi \rangle$.
```

Fig. 5. Algorithm for finding a ξ (if one exists) that satisfies constraints (2)–(7) on solutions of $LP(j,t')$ and maximizes the objective function of $LP(j,t')$ (Eq. (1)).

(2) a'_k would not have its goals satisfied by $\lambda_p(k)$ and would have its goals satisfied by G'. Since a'_k would has its goals satisfied in G' but not in $\lambda_p(k)$, a'_k's cost in ξ can be even equal the total amount of resource that a'_k is endowed with, $\max' c(k) = \mathbf{en}(k)$. This is because an agent prefers all those cooperation structures which would result in one of its goals being satisfied over all those where it is not.

(3) a'_k would not has its goals satisfied both by $\lambda_p(k)$ and G'. Since the agent is not satisfied, agent's cost in ξ should be equal to 0, $\max' c(k) = 0$.

Consequently, we obtain:

$$max'c(k) = \begin{cases} c_k(\lambda_p(k)) & k \in succ(G') \text{ and } k \in succ(\lambda_p(k)) \\ en(k) & k \in succ(G') \text{ and } k \notin succ(\lambda_p(k)) \\ 0 & k \notin succ(G') \text{ and } k \notin succ(\lambda_p(k)). \end{cases}$$

Given an integer program $LP(j,t')$ a set of goals $G' \in 2^G$, in order to find such ξ that $\forall a'_k \in \{a'_t, \dots, a'_n\}$, $c_k(\xi) \leqslant \max' c(k)$, satisfies all goals in G' ($\forall r \in R$, $\sum_{k \in \{a'_{t'}, \dots, a'_n\}} \xi_{k,r} \geqslant req(G', r)$) and maximizes a'_j's utility, the algorithm solves a min cost flow problem.[6] For that purpose the algorithm constructs the corresponding network $G_f = (V, E, c, b, \zeta)$ as follows:

1. Set of vertices: $V = \{V_A \cup V_R \cup \{s,t\}\}$ where V_A represents all agents in $\{a'_{t'}, \dots, a'_n\}$ and V_R corresponds to the resources in $res(G')$. In addition we let the source s and sink t to be two additional new vertices.
2. Set of edges: a directed edge connects between s and each vertex in V_A and between each vertex in V_R and t. In addition a directed edge connect between $v_k \in V_A$ and $v_l \in V_R$ if $en(k, r_l) > 0$:
$$E = \{(s, v_k)\colon\ v_k \in V_a \cup (v_l, t)\colon\ v_l \in V_R \cup (v_k, v_l)\colon\ v_k \in V_a,\ v_l \in V_R,\ en(k, r_l) > 0\}.$$
3. Capacities: for all $v_k \in V_A$ we set $c(s, v_k) = \max' c(k)$. For all $v_l \in V_R$ we set $c(v_l, t) = req(G', r_j)$. For each edge that connects between $v_k \in V_A$ and $v_l \in V_R$ we set $c(v_k, v_l) = en(k, r_l)$.
4. Costs: For each edge that connects between v_j (the vertex that presents agent a'_j) and $v_l \in V_R$ we set $\zeta(v_j, v_l) = 1$. The rest of edges have no cost.
5. Balances: we set $b(s) = req(G')$, $b(t) = -req(G')$ and 0 to the rest vertices.

The following lemma follows immediately from the construction.

Lemma 2. *Given the integer program problem $LP(j,t')$ and a set of goals $G' \in 2^G$, there exists ξ that satisfies the constraints* (2)–(7), *satisfies all goals in G' and a'_j's cost in ξ is equal to r, $c_j(\xi) = r$ if and only if there exists a feasible flow in G_f* (*that satisfies both capacity and balance constraints*) *with cost r. Moreover the amount of a resource $r_l \in R$ contributed by agent k,* (ξ_{k,r_l}) *corresponds to the value of $f(v_k, v_l)$ in the corresponding flow network G_f.*

Given this we can find ξ (if exists) that maximizes agent a'_j utility, satisfies all goals in G' and satisfies the constraints on solutions of $LP(j,t')$ (Eqs. (2)–(7)) by running an algorithm for finding the minimum cost flow on G_f. The integrality theorem for min cost flow shows that in case of flow network with costs has capacities and balances integer valued and a feasible flow in the network exists, then there is a minimum cost feasible flow which is integer valued. Furthermore,

[6] Let $G = (V, E, c, b, \zeta)$ be a directed graph defined by a set of vertices V, and a set of directed edges E. Each edge $(v,u) \in E$ has a capacity $c(v,u) \in R$ and a cost $\zeta(v,u) \in R$. In addition each vertex $v \in V$ has a balance $b(v) \in R$. The min cost flow problem is to find a flow function $f : V \times V \to R$ of minimum cost (the cost of the flow is defined by $\sum_{(v,u)\in E} \zeta(v,u) f(v,u)$) which satisfies the supply/demand constraints (for each vertex $v \in V$, $b(v) = \sum_{(u,v)\in E} f(u,v) - sum_{(v,u)\in E} f(v,u)$) and the capacity constraints (for all $(v,u) \in E$, $f(v,u) \leqslant c(v,u)$) [35].

Klein's algorithm finds such a feasible flow [21] (see [10, pp. 643–698] for a detailed discussion of related algorithms). Our algorithm is shown in detail in Fig. 5.

The complexity of the algorithm is exponential only in the number of goals in G'. This is because the complexity of step 2 is exponential in the number of goals in G', simply because it goes through all the 2^G's subsets. However, the complexity of the internal steps 2.1–2.4 is polynomial, as the problem of finding the minimum cost flow with integral data can be solved in polynomial time [35]. Therefore the total complexity of the algorithm is only exponential in the number of goals but polynomial in the number of agents and in number of resources. □

References

[1] T. Ågotnes, W. van der Hoek, M. Wooldridge, Temporal qualitative coalitional games, in: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2006), Hakodate, Japan, 2006.

[2] Y. Bachrach, J.S. Rosenschein, Computing the Banzhaf power index in network flow games, in: Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2007), Honolulu, Hawaii, 2007, pp. 335–341.

[3] Y. Bachrach, J.S. Rosenschein, Coalitional skill games, in: Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2008), Honolulu, Hawaii, 2008.

[4] S. Banerjee, H. Konishi, T. Sonmez, Core in a simple coalition formation game, Social Choice and Welfare 18 (2001) 135–153.

[5] F. Bloch, E. Diamantoudi, Noncooperative formation of coalitions in hedonic games, Unpublished working paper, 2005.

[6] A. Chakrabarti, L. de Alfaro, T.A. Henzinger, M. Stoelinga, Resource interfaces, in: Proceedings of the Third International Conference on Embedded Software (EMSOFT), in: LNCS, vol. 2855, Springer-Verlag, Berlin, 2003, pp. 117–133.

[7] G. Chalkiadakis, E. Markakis, C. Boutilier, Coalition formation under uncertainty: Bargaining equilibria and the Bayesian core stability concept, in: Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2007), Honolulu, Hawaii, 2007.

[8] K. Chatterjee, B. Dutta, D. Ray, K. Sengupta, A noncooperative theory of coalitional bargaining, The Review of Economic Studies 60 (2) (1993) 463–477.

[9] V. Conitzer, T. Sandholm, Complexity of constructing solutions in the core based on synergies among coalitions, Artificial Intelligence 170 (2006) 607–619.

[10] T.H. Cormen, C.E. Leiserson, R.L. Rivest, Introduction to Algorithms, The MIT Press, Cambridge, MA, 1990.

[11] P. Cramton, Y. Shoham, R. Steinberg (Eds.), Combinatorial Auctions, The MIT Press, Cambridge, MA, 2006.

[12] X. Deng, C.H. Papadimitriou, On the complexity of cooperative solution concepts, Mathematics of Operations Research 19 (2) (1994) 257–266.

[13] P.E. Dunne, M. Wooldridge, M. Laurence, The complexity of contract negotiation, Artificial Intelligence 164 (1–2) (2005) 23–46.

[14] E. Elkind, L. Goldberg, P. Goldberg, M. Wooldridge, Computational complexity of weighted threshold games, in: Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (AAAI-2007), Vancouver, British Columbia, Canada, 2007.

[15] U. Endriss, J. Lang (Eds.), Proceedings of the First International Workshop on Computational Social Choice Theory 2006 (COMSOC-2006), Amsterdam, The Netherlands, December 2006.

[16] S. Hart, A. Mas-Colell, Bargaining value, Econometrica 64 (2) (1996) 357–380.

[17] M. Horniacek, Negotiation, preferences over agreements, and the core, International Journal of Game Theory 37 (2) (2008) 235–249.

[18] C.Y. Huang, T. Sjöstrőm, Consistent solutions for cooperative games with externalities, Games and Economic Behavior 43 (2003) 196–213.

[19] C.Y. Huang, T. Sjöstrőm, Implementation of the recursive core for partition function form games, Journal of Mathematical Economics 42 (2003) 771–793.

[20] S. Ieong, Y. Shoham, Marginal contribution nets: A compact representation scheme for coalitional games, in: Proceedings of the Sixth ACM Conference on Electronic Commerce (EC'05), Vancouver, Canada, 2005.

[21] M. Klein, A primal method for minimum cost flows with application to the assignment and transportation problem, Management Science 14 (1967) 205–220.

[22] S. Kraus, Strategic Negotiation in Multiagent Environments, The MIT Press, Cambridge, MA, 2001.

[23] Benny Moldovanu, Eyal Winter, Order independent equilibria, Games and Economic Behavior 9 (1995) 21–34.

[24] M. Naor, On fairness in the carpool problem, Journal of Algorithms 55 (2005) 93–98.

[25] M.J. Osborne, A. Rubinstein, A Course in Game Theory, The MIT Press, Cambridge, MA, 1994.

[26] C.H. Papadimitriou, Computational Complexity, Addison–Wesley, Reading, MA, 1994.

[27] M. Perry, P.J. Reny, A noncooperative view of coalition formation and the core, Econometrica 62 (4) (1994) 795–817.

[28] J.S. Rosenschein, G. Zlotkin, Rules of Encounter: Designing Conventions for Automated Negotiation among Computers, The MIT Press, Cambridge, MA, 1994.

[29] T. Sandholm, K. Larson, M. Andersson, O. Shehory, F. Tohmé, Coalition structure generation with worst case guarantees, Artificial Intelligence 111 (1–2) (1999) 209–238.

[30] R. Serrano, R. Vohra, Non-cooperative implementation of the core, Journal Social Choice and Welfare 14 (4) (1997) 513–525.

[31] O. Shehory, S. Kraus, Coalition formation among autonomous agents: Strategies and complexity, in: C. Castelfranchi, J.-P. Müller (Eds.), From Reaction to Cognition — Fifth European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW-93, in: LNAI, vol. 957, Springer-Verlag, Berlin, 1995, pp. 56–72.

[32] O. Shehory, S. Kraus, Task allocation via coalition formation among autonomous agents, in: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95), Montréal, Québec, Canada, August 1995, pp. 655–661.

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

[34] Y. Shoham, K. Leyton-Brown, Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations, Cambridge University Press, Cambridge, 2008.

[35] P.T. Sokkalingam, R.K. Ahuja, J.B. Orlin, New polynomial-time cycle-canceling algorithms for minimum-cost flows, Networks 36 (1) (2000) 53–63.

[36] J.J. Vidal-Puga, A bargaining approach to the consistent value for NTU games with coalition structure, Unpublished working paper, 2003.

[37] M. Wooldridge, An Introduction to Multiagent Systems, John Wiley & Sons, New York, 2002.

[38] M. Wooldridge, P.E. Dunne, On the computational complexity of qualitative coalitional games, Artificial Intelligence 158 (1) (2004) 27–73.

[39] M. Wooldridge, P.E. Dunne, On the computational complexity of coalitional resource games, Artificial Intelligence 170 (10) (2006) 853–871.