A Message-Passing Approach to Computing Stable Coalitions on Graphs

ABSTRACT

Many real-life settings—such as communication networks, sensor networks, or the electricity grid—require the development of decentralised mechanisms that allow agents to organize into stable coalitions whose potential membership is restricted by a graph. To meet these challenges, in this paper we develop a novel graphical model representation scheme for coalitional games defined over graphs, and exploit this representation to devise decentralised algorithms leading to the formation of core-stable coalition structures. For games defined over trees, we propose an algorithm that allows agents to identify a stable coalition structure. That is, an optimal coalition structure along with a payoff allocation such that the resulting pair is in the core of the game. Similarly, for arbitrary graphs, we develop an algorithm that either outputs a core element—or, alternatively, detects core emptiness. Both algorithms build upon well-known message passing algorithms from the GDL family [20, 17] which we extend to this setting.

1. INTRODUCTION

Many real-life multiagent settings, such as communication or sensor networks, or, more recently, the *smart electricity Grid*, can benefit from the development of *decentralised* algorithms that allow agents to organize into stable coalitions. Moreover, it is natural in such settings to assume that coalition membership can be restricted by some kind of graph, reflecting realistic barriers to the formation of certain agent teams.

As an illustrating example, consider the need to maintain supply reliability in the modern electricity network while curtailing carbon emissions and costs. One key requirement in this domain is that demand should follow supply in order to make optimal use of intermittent sources of renewable energy, while ensuring that electricity generation and electricity consumption are perfectly matched [1]. Given this, a promising electricity demand-side management strategy is to promote the formation of coalitions among energy consumers with near-complementary consumption restrictions, by promising them rewards for collectively "flattening" their joint energy consumption curve via shifting around their individual consumption. From the Grid's perspective, the resulting coalitions should be achieving the highest possible savings in terms of energy consumption. At the same time, they should be as stable as possible, in the sense that it is more efficient for the Grid to interact with specific entities, while production-consumption balance is naturally hurt when consumers keep moving around in coalitions, altering consumption savings plans. However, consumers are rational utility maximizers, and have to be incentivised to stay in a coalition via appropriate monetary payoffs. Clearly, it is an absolute necessity in domains like this to devise decentralised processes to allow autonomous rational agents to join together into stable coalitions, that are also optimal from the system designer's point of view.

Such problems can be modelled naturally as *coalitional games*, where the following questions have to be resolved: *(i)* the set of coalitions with maximum collective value, that is, an optimal coalition structure, has to be identified; and *(ii)* each coalition's value has to be distributed among its members in such way that coalition members have no incentive to break away from the identified optimal structure [14]. When this happens, we say that a game outcome is in *the core*. Moreover, decentralised processes to achieve these objectives need to be identified.

Against this background, in this paper we present a model and tools to accomodate the needs of settings such as the ones listed above. Importantly, we develop a novel representation scheme for coalitional games over graphs, and exploit this to device *decentralised* algorithms that allow agents to find a stable coalition structure on arbitrary graphs. By so doing, we are proposing the first decentralised algorithm performing on general graphs that can deal simultaneously with two key activities which are usually treated separately in the coalition formation literature: namely, identifying an optimal coalition structure *and* an accompanying payoff allocation so that core-stable elements emerge. Thus, our algorithms not only compute, but also incentivize agents to form the optimal coalition structure (via providing them with structure-stabilizing payoffs).

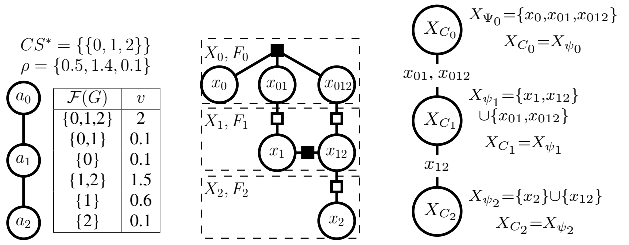
In more detail, our contributions are as follows. First, we provide a novel graphical model representation of the coalition structure generation problem over graphs. Second, building on this model, we show that one can use existing algorithms in the literature based on the GDL framework (e.g., [20, 15, 2]) to allow agents to identify the optimal coalition structure. Third, we formulate two novel message-passing algorithms that extend the well-known GDL message passing algorithm [2], and exploit our proposed representation to compute *stable* coalition structures. The first of these algorithms, *SCF-Trees*, is guaranteed to converge to a stable coalition structure on tree graphs; while *SCF-Graphs* builds on *SCF-Trees* to allow agents to compute an stable coalition structure or alternatively detect the emptiness of the core on *general graphs*.

The rest of this paper is structured as follows. In Section 2, we review the literature and in Section 3, we describe our novel graphical model representation. We present the decentralised coordination algorithms in Section 4 . Finally, Section **??** concludes.

2. BACKGROUND

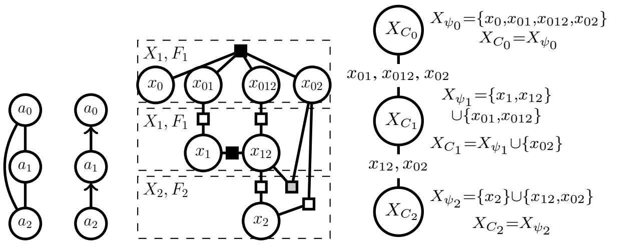
In this section, we present some essential background and position

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), June, 4–8, 2012, Valencia, Spain.



(a) Game over an acyclic graph (b) Representation (c) Junction tree

Figure 1: Example of a) a game on an acyclic graph; b) a representation of (a); and c) a junction tree of (b).



(a) G (b) PT (c) Representation (d) Junction tree

Figure 2: Example of a) a graph with a cycle; b) a representation of (b); and c) and junction tree of (b).

our approach within the existing literature.

2.1 Coalitional games on graphs

A coalitional ("transferable utility", or "characteristic function") game is traditionally defined as follows. Let $A = \{a_1, \ldots, a_n\}$ be a set of agents. A subset $S \subseteq A$ is termed a coalition. Then, a coalitional game CG is completely defined by its *characteristic function* $v : 2^A \to \Re$ (with $v(\emptyset) = 0$), which assigns a real value representing (transferable) utility to every feasible coalition [14]. Agents in a coalition are then permitted to freely distribute coalitional utility among themselves. Given a game CG, a *coalition structure* $CS = \{S_1, \ldots, S_k\}$ is an exhaustive disjoint partition of the space of agents into coalitions. We overload notation by denoting by $v(CS)$ the (intuitive) worth of a coalition structure: $v(CS) = \sum_{S \in CS} v(S)$. We also denote the set of all possible coalition structures by $\mathbf{CS}$.

Assume now that feasible coalitions are determined by a *graph* G: *(i)* each node of the graph represents an agent; and *(ii)* a coalition S is allowed to form if and only if every two agents in S are connected by some path in the subgraph induced by S. We denote the set of agent nodes in G by $A(G)$. Given a set of agents $A \subseteq A(G)$ we also denote G_A as the subgraph of G induced by A and $G_{\setminus A}$ as the subgraph of G induced by all the agents $A(G)$ excluding those in A.

Definition 1. *A coalitional game CG on a graph G is a tuple $\langle A, v, F(G)\rangle$ where: (i) $F(G)$ is the set of all* feasible *coalitions—i.e., coalitions permitted to form given G; and (ii) v is the characteristic function, defined for all coalitions in $F(G)$.*

An example of a game on a graph in which three agents interact in a line is given in Figure 1(a). Notice that $\mathbf{CS}$ is now restricted to the set of possible coalition structures given $F(G)$. Let $F_{A'}(G)$ be the set of feasible coalitions that contain some agent in A', $F_{A'}(G) = \{S \in F(G) | S \cap A' \neq \emptyset\}$. Then, in figure 1(a) the set of coalitions that contain agent a_0 is $F_{\{0\}}(G) = \{\{0\}, \{0,1\}, \{0,1,2\}\}$ and agent a_0 can form a coalition that contains itself and a_1 ($S = \{0,1\}$) with value $v(\{0,1\}) = 0.1$ but it can not compose a coalition with a_2 without a_1 (e.g. $S = \{0,2\} \notin F_{\{0\}}(G)$).

Given a game on a graph $\langle A, v, F(G)\rangle$, a *coalition structure* $CS = \{S_1, \ldots, S_k\}$ is a set of feasible ($\forall S \in CS : S \in F(G)$), exhaustive ($S_1 \cup \ldots \cup S_k \supseteq A$) disjoint ($\forall S, S' \in CS$: $S \cap S' \cap A = \emptyset$) coalitions with respect to agents in A. While the restriction of $\mathbf{CS}$ to be exhaustive and disjoint *with respect to* A differs from the traditional in the literature, it is required later on the formalisation of algorithms and proofs, to consider games in which the graph G contains agents nodes not in A ($A \subset G$). Observe that, if this is the case, this definition allows these *ghosts* agents to be in more than one coalition or in no coalition at all; while if $A = G$, the traditional setting persists.

A well-studied problem, due to its apparent significance, is the *coalition structure generation (CSG) problem*, aiming to identify the coalition structure CS^* that maximizes *social welfare*—i.e., the coalition structure with maximal value. As an example, in Figure 1(a) the social welfare-maximizing structure is composed of a single coalition that includes all the three agents, $CS^* = \{\{0,1,2\}\}$, with a value of 2. The CSG problem is known to be hard [18].

Now, agents are selfish and thus need to decide how the value of their coalition should be distributed. A vector $\rho = \{\rho_1, \ldots, \rho_n\}$ assigning some payoff to each agent $a_i \in A$ is called an *allocation*. We denote the set of payments for a subset of agents S, $\bigcup_{i \in S} \rho_i$, by $\rho(S)$ and the sum of these payments, $\sum_{i \in S} \rho_i$, by ρ_S. An allocation ρ is an *imputation* for a given CS, if it is efficient ($\rho_S = v(S)$ for all $S \in CS$), and individually rational (that is, $\rho_i \geq v(\{i\})$ for all a_i). Note that if ρ is an imputation for CS, then $\rho_A = v(CS)$. Thus, in Figure 1(a), $\{\rho_1 = \frac{2}{3}, \rho_2 = \frac{2}{3}, \rho_3 = \frac{2}{3}\}$ and $\{\rho_1 = 0.5, \rho_2 = 1.4, \rho_3 = 0.1\}$ are two different imputations for $CS^* = \{\{0,1,2\}\}$.

A game outcome is a (coalition structure, imputation) pair, assigning agents to coalitions and allocating payoffs to agents efficiently. However, this does not mean that a game outcome is necessarily stable: there might be agents that feel there are outcomes that can make them better-off. This raises the question of identifying stable outcomes. The *core* is arguably the main stability solution concept in cooperative games. It is the set of coalition structure-imputation tuples (CS, ρ) such that no feasible coalition has a deviation incentive. Formally:

$$Core(CG) = \{(CS, \rho) : \rho_A = v(CS) \,\&\, \rho_{S \cap A} \geq v(S)\, \forall S \in F(G)\}$$

Again notice that definition of the core is slightly modified to consider games where $A \subset G$ for which the value of a coalition S is shared only by agents in A; while if $A = G$, $S \cap A = S$,the traditional setting persists.

Thus, in Figure 1(a) allocation $\{\rho_1 = 0.5, \rho_2 = 1.4, \rho_3 = 0.1\}$ is in the core of the game but allocation $\{\rho_1 = \frac{2}{3}, \rho_2 = \frac{2}{3}, \rho_3 = \frac{2}{3}\}$ is not because $\frac{2}{3} \cdot 2 \leq v(\{1,2\}) = 1.5$.

Notice that *only optimal coalition structures might admit an element in the core*: intuitively, if the current structure is suboptimal then a subset of agents can be made strictly better off by moving to an optimal coalition structure. Note also that the core is a strong solution concept, as it is empty in a plethora of games; therefore, the question of the core non-emptiness is key in many settings.

2.2 GDL message-passing algorithm

GDL is a general message-passing algorithm that exploits the way a global function factors into a combination of local functions to compute the objective function in an efficient manner. The importance of the GDL framework stems from unifying a family of techniques (e.g. Viterbi's, Pearl's belief propagation or Shafer-Shenoy algorithms to name a few) which have been widely used in different areas such as information theory or computer vision. Thus, consider a function F, that is dependent on N variables, $\mathcal{X} = \{x_1, \ldots, x_n\}$, and is defined as the combination of M factors $\mathcal{F} = \{f_1, \ldots, f_m\}$ such that $F(X) = \bigotimes_{f \in \mathcal{F}} f_m(X_m)$ where $X_m \subseteq \mathcal{X}$ are the variables in the domain of f_m and $\bigotimes$ stand for the combination (also called joint) operator. This global function can be encoded using a particular type of graphical model, called factor graph, a bipartite graph composed of two kinds of elements: variable nodes ($\mathcal{X}$) and function nodes ($\mathcal{F}$). Then the objective function is to find the assignment of variables in $\mathcal{X}$, the optimal solution of the factor graph $\langle \mathcal{X}, \mathcal{F} \rangle$ that maximize the global function $X^* = arg \max_X \bigotimes_{f \in \mathcal{F}} f_m(X_m)$.

In order to ensure optimality and convergence, GDL arranges the objective function to assess in a junction tree (also known as junction tree or clique tree). A junction tree for $\langle \mathcal{X}, \mathcal{F} \rangle$ is a tree of cliques that can be represented as a triple $\langle \mathcal{C}, \Psi, \mathcal{S} \rangle$ where:

- $\mathcal{C} = \{X_{C_1}, \ldots, X_{C_n}\}$ is a set of cliques, where each clique X_{C_i} is a subset of variables $X_{C_i} \subseteq \mathcal{X}$.
- $\Psi = \{\psi_1, \ldots, \psi_m\}$ is a set of potentials, one per clique, where a potential ψ_i is defined as the combination of a set of functions $\mathcal{F}_i \subseteq \mathcal{F}$, $\psi_i(X_{\psi_i}) = \bigotimes_{f \in \mathcal{F}_i} f(X_f)$.
- S is a set of separators, where a separator $Sep_{ij} \in \mathcal{S}$ is an edge between clique X_{C_i} and X_{C_j} containing their intersection, namely $Sep_{ij} = X_{C_i} \cap X_{C_j}$.

Furthermore, the following properties must hold: (i) (*Covering*) Each potential domain is a subset of the clique to which it is assigned ($X_{\psi_i} \subseteq X_{C_i}$) and each function $f \in \mathcal{F}$ is included in exactly one potential; (ii) (*Running intersection*) If a variable x_i is in two cliques X_{C_i} and X_{C_j}, then it must also be in all cliques on the path between them.

The purpose of GDL is that cliques distributedly compute the objective function that is factored among them. With that goal GDL defines a message-passing phase for cliques to exchange information about their variables. Here, we focus on the single-vertex message-passing version of GDL, in which the goal is to compute the objective function at only one clique X_{C_i}. The single-vertex GDL algorithm is executed over a rooted junction tree, in which each edge is directed toward the target clique X_{C_i}.

Then during the execution of the *single-vertex GDL algorithm* over a rooted junction tree $\langle \mathcal{C}, \Psi, \mathcal{S} \rangle$, each clique X_{C_i} exchanges a message $\mu_{i \to p}$ with its clique parent X_{C_p}, when, for the first time, it has received messages from all its children:

$$\mu_{i \to p}(Sep_{ip}) = \max_{X_{C_i} \setminus Sep_{ip}} \psi_i(X_{\psi_i}) \otimes \bigotimes_{j \in Ch_i} \mu_{j \to i}(Sep_{ji}) \quad (1)$$

where Ch_i stands for index of cliques's children of X_{C_i} in the rooted junction tree.

Then, upon receiving messages from all its children, each clique X_{C_i} computes its state function (also known as belief or knowledge function) as:

$$s_i(X_{C_i}) = \psi_i(X_{\psi_i}) \otimes \bigotimes_{j \in Ch_i} \mu_{j \to i}(Sep_{ji}) \quad (2)$$

After the single-vertex GDL execution is over, the state function of any clique $X_{C_i} \in \mathcal{C}$ summarizes its local knowledge over variables $X^*_{C_j}$, $s_i(X_{C_i}) = \max_{X \setminus X_{C_i}} \bigotimes_{j \in D_i \cup \{i\}} \psi_j(X_{C_j})$ where D_i stands for the index of the descendants of X_{C_i} in the rooted junction tree. Then, cliques can infer the optimal values of its variables by executing a *value-propagation* phase that recursively applies: $X^*_{C_j} = arg \max_{X_{C_j}, Sep_{jp} = Sep^*_{jp}} s_j(X_{C_j})$ where p stands for the parent of X_{C_i} in the directed junction tree and Sep^*_{jp} stands for the values of Sep_{jp} variables already inferred on cliques up X_{C_j} in the junction tree. Notice that $\bigcup_{X_{C_j} \in \mathcal{C}} X^*_{C_j}$ recovers the optimal solution X^*.

2.3 Related Work and Discussion

There is an abundance of papers dealing with various aspects of the coalition formation problem—some focusing on coalition structure generation, others on the problem of allocating payoff to the agents in some fair or stable manner. For instance, a number of algorithms have been proposed to compute the set of optimal coalitions [12, 6, 16]. However, these methods ignore the fact that individual agents—say electricity consumers—have their own preferences, and thus need to be provided with incentives in order to form the optimal coalition structure. On the other hand, the provision of such incentives via the allocation of substantial payoff, has long been studied in the cooperative games literature [9, 21, 5, 8]. However, relevant research to date mostly focuses on characterising the coalitional game outcomes, and on determining the complexity of identifying such outcomes, rather than providing algorithms that the agents can use in order to actually form stable coalitions. Moreover, in many occasions the optimality of the grand coalition[1] is assumed; hence, the payoff allocation problem is kept (artificially) isolated from the coalition structure generation one. In our work here, we tackle both problems simultaneously. Furthermore, while most previous work has viewed coalition formation as a problem to tackle in a centralized manner, not satisfying the decentralisation requirement present in many domains [6, 16], we provide *decentralized algorithms* to identify core-stable (coalition structure, payoff allocation) pairs.

Now, the idea of having coalitions whose potential membership is restricted by some kind of graph is an old one, since it naturally reflects many real-life situations. Work in *network formation*, in particular, following the seminal work of Myerson [13], has attempted to solve the problem of *progressively building* stable coalitional structures in networks, through the addition and removal of links among nodes [11]. That line of research, however, focused mostly on non-cooperative aspects of the coalition formation problem—for instance, by modelling the problem as a bargaining game or some other type of game in extensive form.

In *cooperative* settings, starting with the seminal work of Deng and Papadimitriou in [8], there has been work on graph-inspired *representations* for coalitional games. Such representations include Ieong and Shoham's marginal contribution nets [10], Bachrach *et al.*'s hypergraph-based representation to tackle coalition structure generation in skill games [3], and Brafman *et al.*'s work on identifying succinct coalitional game representations to model multiagent *planning* problems [4]. Moreover, there has been some work on *cooperative solution concepts* in graph-restricted games [19, **?**]. However, that work has *not* for the most part focused on the concept of the core, neither has it attempted to address the question of how core-stable coalitions emerge.

One exception is the work of Demange [7]. She proved that when the graph restricting a game is a tree there always exists an element in the core; and, moreover, presented a process that identifies a coalition structure and a payoff allocation that lie in the core.

Our work here differs to the work of Demange. As a matter

[1]The grand coalition is the coalition of all agents.

of fact, we extend that work, in the sense that ours is a decentralized algorithm, defined over a novel graphical representation of the coalition formation problem. In clear distinction to Demange's approach, our algorithm works on a representation which, while being a junction tree of the original graph, is nevertheless a tree whose nodes are *not* agents—but, rather, variable and function nodes of a factored graph. Indeed, it extends the well-known GDL message passing algorithm in this domain, and is thus both intuitive and able to fit within the taxonomy of algorithms in the related literature. As such, it also readily allows the treatment of graphs with cycles, while Demange's algorithm was confined to trees and did not provide any intuitions on how to perform such an extension.

3. PROBLEM REPRESENTATION

In this section we present a new representation of a coalitional game on a graph G in terms of a factor graph that efficiently captures the interactions among agents in G. This novel representation is based on arranging agents into a pseudotree PT.

Definition 2 (Pseudotree)**.** *A pseudotree PT of a graph G over a set of agents A is a rooted tree with agents A as nodes and the property that for any pair of agents $i, j \in A$ if exists any path between them in G composed of agents not in A then i, j are on the same branch*[2] *in PT. In the case $A(G) = A$, this property reduces to: any two agents that share an edge in G are on the same branch in PT.*

Figure 2(b) shows a pseudotree, rooted at agent a_0, of the cycle graph G in Figure 2(a) over agents $A(G) = \{a_1, a_2, a_3\}$. Observe that any pseudotree of G will form a line between agents in $A(G)$ because any pair of agents in $A(G)$ share an edge in G and hence, should be placed in the same branch. We denote $A(PT)$ as the set of agents nodes in PT. Then, given any agent $a_i \in A(PT)$ we denote Ch_i as its children, p_i as its parent, An_i as its ancestors and D_i as its descendants and PT_i as the subtree rooted at a_i in PT. Thus, in Figure 2(a), $Ch_1 = D_1 = \{2\}$, $p_i = 0$, $An_1 = \{0\}$ and PT_1 is a tree rooted at a_1 composed of agents a_1, a_2 sharing an edge between them.

Since the pseudotree defines a variable ordering among agents in A, given a game on a graph $CG = \langle A, v, F(G)\rangle$ and a pseudotree of G over A this ordering allows us to partition the set of feasible coalitions into $|A|$ disjoint sets $\{\mathbf{S_i} | a_i \in A\}$, one per agent, where the set of coalitions $\mathbf{S_i}$ contains all the feasible coalitions that include agent a_i but no agent up a_i in PT, $\mathbf{S_i} = F_{\{i\}}(G_{\setminus An_i})$.

Definition 3 (Required coalitions)**.** *Given a game on a graph $CG = \langle A, v, F(G)\rangle$ and a pseudotree PT of G over A, we define the set of required coalitions for a coalition $S \in \mathbf{S_i}$, $Req(S)$, as $Req(S) = \bigcup_{j \in Ch_i} Req(S, j)$ being $Req(S, j)$ recursively defined as:*

$$Req(S,j) = \begin{cases} \emptyset & \textit{if } S \cap A(PT_j) = \emptyset \\ S' \cup \bigcup_{k \in Ch_j} Req(S \setminus S', k) & \textit{otherwise} \end{cases} \tag{3}$$

where $S' = arg \max_{\{S'' \in \mathbf{S_j} | S'' \subseteq (S \cap A(PT_j))\}} |S'' \cap S|$, that is the coalition in $\mathbf{S_i}$, strictly composed of variables in $S \cap A(PT_j)$, with maximum intersection with S'.

In Figure 2(a), the set of required variables for coalition $\{012\}$ is $Req(\{012\}, a_1) = \{12\} \cup Req(\{0\}, a_2) = \{12\}$, whereas the set of required variables for $\{02\}$ is $Req(\{02\}, a_1) = \emptyset \cup Req(\{02\}, a_2) = \{2\}$.

[2]A branch stands for the path between a leaf node and the root node in PT.

The intuition behind the concept of required coalitions is that an agent a_i when aiming to form one of its local coalition $S \in \mathbf{S}_i$ ensures the participation, and exclusivity, of agents in S down a_i in the tree $(S \cap D_i)$, not directly through them but by means of the set of required coalitions $Req(S)$. Thus, agent a_0 for its local coalition $\{012\}$ will not negotiate the participation of a_2 directly with him but would negotiate directly with agent a_1 to obtain its exclusivity and those of agent a_2 through the required coalition $\{12\}$, local to agent a_1. Next, we formulate a novel representation of a coalitional game in terms of factor graph that exploits this idea in to efficiently capture the dependencies that emerge among agents when coalitions are restricted by a graph.

Definition 4 (Representation)**.** *Given a game on a graph $CG = \langle A, v, F(G)\rangle$ and a pseudotree PT of G over A, we define a factor graph representation of CG as $R(CG, PT) = \langle \mathcal{X}, \mathcal{F}\rangle$ where:*

. *$\mathcal{X} = \{X_1 \cup \ldots \cup X_{|A|}\}$ is a set of binary variables, one per feasible coalition, that are partitioned in $|A|$ disjoint sets, one per agent. Analogously to the concept of local coalitions, given an agent $a_i \in A$, its set of local variables X_i contains all the coalitions variables that include agent a_i but no agent up a_i in PT. Formally, $X_i = \{x_S | S \in F_{\{i\}}(G_{\setminus An_i})\}$.*

. *$\mathcal{F} = \{F_1 \cup \ldots \cup F_{|A|}\}$ is a set of functions that are partitioned in $|A|$ disjoint sets, one per agent. Given an agent a_i, its set of local functions F_i is composed of:*

- *$\{f_v(x_S) | x_S \in X_i\}$, a set of* value functions*, one per variable in X_i, where a function $f_v(x_S)$ returns the value of coalition S when $x_S = 1$ ($f_v(x_S = 1) = v(S)$).*

- *$f_u(X_i)$, a* unique function *that controls that one and only one of the variables X_i set to 1:*

$$f_u(X_i) = \begin{cases} 0, & \sum_{x_S \in X_i} x_S = 1 \\ -\infty, & \textit{otherwise} \end{cases} \tag{4}$$

- *A set of functions that capture the dependencies between each variable $x_S \in X_i$ and its set of requiring variables $X_S^{\setminus r}$ where $X_S^{\setminus r}$ stands for all the variables corresponding to coalitions that require S ($X_S^{\setminus r} = \{x_{S'} | S \in Req(S')\}$). Formally for each $x_S \in X_i$, the following two sets of variables are included:*

  . *$\{f_r(x_S, x_{S'}) | x_{S'} \in X_S^{\setminus r}\}$, a set of* require functions*, one per requiring variable of x_S. Given a requiring variable $x_{S'} \in X_S^{\setminus r}$ for x_S, the require function $f_r(x_S, x_{S'})$ controls that $x_{S'}$ is activated only if its requested variable x_S is also activated, substracting the value of x_S in this case. Formally,*

$$f_r(x_S, x_{S'}) = \begin{cases} -\infty, & x_{S'} = 1 \textit{ and } x_S = 0 \\ -v(S), & x_{S'} = 1 \textit{ and } x_S = 1 \\ 0, & \textit{otherwise} \end{cases} \tag{5}$$

  . *$\{f_b(x_{S'}, x_{S''}) | x_{S'}, x_{S''} \in X_S^{\setminus Req}\}$ a set of blocking functions, one per each pair of variables that requested the same variable x_S. Intuitively, the set of blocking functions control that a most one of the coalitions variables that require x_S activates. Formally,*

$$f_b(x_S, x_{S'}) = \begin{cases} -\infty, & x_S = 1 \quad x_{S'} = 1 \\ 0, & \textit{otherwise} \end{cases} \tag{6}$$

Figure 2(c) shows this factor graph representation for the game in Figure 2(a) where circle nodes stand for variables, square nodes stand for functions and each function node is linked to all variables included in its domain. Unique functions are filled in black, require in white and blocking in grey whereas value functions are omitted for the sake of clarity. Thus, for example, X_2 contains variables x_1 and x_{12}, one per a_2's local coalitions (those that a_2 can form in G with agents down PT). Then, a unique function (included in F_1) between x_1 and x_{12} controls that exactly one of them is activated. Finally F_1 also contains two require functions that encode the dependency between x_1 and its requiring variable x_{01} (linking x_1 and x_{02}) and between x_{12} and its requiring variable x_{012} (linking x_{12} and x_{012}. The only blocking function in this example is contained as part of a_3's local functions and controls that only one of the two variables that require x_2, x_{12} and x_2, is activated. Analogously, Figure 1(b) shows the factor graph representation but for the game in Figure 1(b).

Given the representation of a CG using the definition above, $R(CG, PT) = \langle \mathcal{X}, \mathcal{F} \rangle$, its optimal solution is defined as $X^* = arg\max_X \sum_{f\in\mathcal{F}} f(X) = \mathcal{F}(X)$. Then, to recover the optimal coalition structure CS^* in CG from X^*, next, we define a mapping Ω that maps any assignment of variables X in $R(CG, PT)$ to a coalition structure in CG.

Definition 5 (Ω). *Given a representation $R(CG, PT) = \langle \mathcal{X}, \mathcal{F} \rangle$, Ω is a function that maps any assignment for any set of variables $X \subseteq \mathcal{X}$ into a coalition structure CS composed of all coalitions S activated in X ($x_S = 1 \in X$) for which it does not exist any other coalition that contains all agents in S activated in X ($\nexists x_{S'} = 1 \in X$ such that $S \subset S'$).*

Thus, the solution of the representation in Figure 1(b) $X^* = \{x_0 = 0, x_{01} = 0, x_{012} = 1, x_1 = 0, x_{12} = 1, x_2 = 1\}$ is mapped by Ω to the optimal coalition structure CS^* of the corresponding game in 1(a), $\Omega(X^*) = \{\{012\}\}$.

Then, the following Theorem 2 proves that given mapping Ω, the representation $R(CG, PT)$ where $CG = \langle A(G), v, F(G) \rangle$ is correct (the optimal solution of $R(CG, PT)$ identify the optimal coalition structure in CG, $\Omega(X^*) = CS^*$).

Theorem 1. *Given a game on a graph $CG = \langle A(G), v, F(G) \rangle$ and a pseudotree PT of G over $A(G)$, the representation $R(CG, PT) = \langle \mathcal{X}, \mathcal{F} \rangle$ is correct: $\forall_{X|\mathcal{F}(X)\neq\infty} : \Omega(X) \in \mathbf{CS}$ & $\mathcal{F}(X) = v(\Omega(X))$.*

PROOF. By means of value functions $\{v(x_S)|x_S \in \mathcal{X}\}$, the value of any solution X in $R(CG, PT)$ adds the value of any coalition S whose corresponding variable is set to 1 in X ($\sum_{x_S=1\in X} v(S)$). Considering the definition of mapping Ω, to prove that $\forall_{X|\mathcal{F}(X)\neq\infty}$: $\Omega(X) \in \mathbf{CS}$ we only need to prove that any pair of variables activated in a valid solution X $x_S = 1, x_{S'} = 1 \in X$ any valid[3] satisfy one of the following conditions: (i) $S \cap S' = \emptyset$; or (ii) $S \subset S'$; or (iii) $S' \subset S$. Let's prove this by contraction. Let's assume that two variables $x_S \in X_i, x_{S'} \in X_j$ such that $S \cap S' \neq \emptyset$ are set to 1 in a valid configuration X but $S \not\subset S'$ nor $S' \not\subset S$.

Case $i = j$ clearly leads to a contradiction because function $f_u(X_i) \in \mathcal{F}$ restricts that only one variable in X_i is set to 1.

Consider now the case $i \neq j$. Notice that, due to the recursive nature of require functions, the activation of a variable x_S not only requires the activation of variables corresponding to the required coalitions of S, but also of variables corresponding to the required coalitions of those and so on. For example, in Figure 2(c), the activation of x_{012} directly requires the activation of x_{12} but also the activation of x_2 (indirectly required through coalition $\{12\}$). Let X_S^{Act} be the set of variables recursively required for the activation of x_S. X_S^{Act} contains a coalition $x_{S''} \in X_k$, that we refer to as $x_S^{Act,k}$, per agent $k \in S$ where $S'' \subseteq S$ stands for the local coalition of a_k with maximum intersection with S ($S'' = arg\max_{\{x_{S_i} \in X_i | S_i \subseteq S\}} |S_i \cap S|$). Then, X_S^{Act} and $X_{S'}^{Act}$, contain, for each $k \in S' \cap S$, one coalition in X_k. Now from here we will distinct two cases. Consider that $\exists k \in S' \cap S$ such that $x_S^{Act,k}$ and $x_{S'}^{Act,k}$ are different ($x_S^{Act,k} \neq x_{S'}^{Act,k}$). This leads to a contradiction because $f_u(X_k)$ restricts that only one variable can be set to 1 at X_k. This is the case in Figure 2(c) of coalitions x_{01} and x_{12}, the two respective coalitions overlap in agent a_1 but in X_1 variable x_{01} requires x_1 whereas x_{12} requires x_{12}. Now consider the remaining case in which $\forall k \in S' \cap S : x_S^{Act,k} = x_{S'}^{Act,k}$. Let $x_{S''} = x_S^{Act,k} = x_{S'}^{Act,k}$. Then in this case, that means that for some $\forall k \in S' \cap S$, $\exists\ l \in S' \setminus S$ and $m \in S \setminus S'$ such that $x_S^{Act,l}, x_{S'}^{Act,m} \in X_{S''}^{\setminus r}$ leading to the contradiction because function $f_b(x_S^{Act,l}, x_{S'}^{Act,m})$ blocks the joint activation of $x_S^{Act,l}$ and $x_{S'}^{Act,m}$. For example, in Figure 2(a) variables x_{12} and x_{02}, that overlap on agent a_2, require the activation of the same variables in X_2, namely $X_{\{12\}}^{Act,2} = X_{\{02\}}^{Act,2} = \{x_2\}$ $X_{\{12\}}^{Act,1} = x_{12}$, $X_{\{02\}}^{Act,0} = x_{02}$ and $x_2 \in Req(\{12\})$, $x_2 \in Req(\{x_{02}\})$, a blocking function exists between x_{12} and x_{02}. Thus, we proved that for any X, $\forall_{X|\mathcal{F}(X)\neq\infty}\Omega(X) \in \mathbf{CS}$.

Now to prove Theorem 2 it only remains to show that $\forall_{X|\mathcal{F}(X)\neq\infty}$ $\mathcal{F}(X) = v(\Omega(X))$. Since value functions $\{v(x_S)|x_S \in \mathcal{X}\}$ add the value of any coalition S whose corresponding variable is set to 1 in X ($\sum_{x_S=1\in X} v(S)$) we only need to prove that for any pair of variables x_S, x'_S activated in X such that $S' \subset S$, the value of coalition S' is substracted. In any valid solution X, if x_S and $x_{S'}$ are activated being $S' \subset S$, that means that $S' \in X_S^{Act}$ and since by sequentially application of require functions starting from x_S the value of all variables in $X_S^{Act} \setminus \{x_S\}$ is substracted, Theorem 2 holds.

Theorem 2. *Given a game on a graph $CG_{A(G)\setminus A} = \langle A, v, F(G) \rangle$ and a pseudotree PT of G over A, the representation $R(CG, PT) = \langle \mathcal{X}, \mathcal{F} \rangle$ satify that: $\forall_{X|\mathcal{F}(X)\neq\infty} : \Omega(X) \in \widetilde{\mathbf{CS}}$ & $\mathcal{F}(X) = v(\Omega(X))$.*

Thus, given a game over a graph $CG = \langle A(G), v, F(G) \rangle$ under the proposed representation you can use a message-passing GDL algorithm, as the one reviewed in Section 2.2, to allow agents to identify the optimal coalition structure in CG. Since GDL algorithms are executed over a junction tree, next we formulate a particular junction tree for $R(CG, T)$.

Definition 6 (γ). *Let γ be a function that given a game on a graph $CG = \langle A, v, F(G) \rangle$ and a pseudotree PT of G over A maps them to a junction tree $\gamma(CG, PT) = \langle \mathcal{C}, \Psi, \mathcal{S} \rangle$, where:*

- *$\Psi = \{\psi_i | a_i \in A\}$ contains one potential per agent in A, where $\psi_i(X_{\psi_i})$ is defined as the combination of functions F_i (F_i defined as in Def. 4). Then, $X_{\psi_i} = \bigcup_{f\in F_i} X_f = \{X_i \cup \bigcup_{x_S\in X_i} X_S^{\setminus r}\}$.*
- *$\mathcal{C} = \{X_{C_i} | a_i \in A\}$ contains one clique per agent in A, where $X_{C_i} = X_{\psi_i} \cup \bigcup_{j\in Ch_i} X_{C_i} \setminus X_j$.*
- *$\mathcal{S}$ is a set of separators that contains one separator Sep_{ij} per pair of cliques X_{C_i} and X_{C_j} such that a_j is parent of*

[3] A solution is valid when it does not violate any hard constraint: $\sum_{f\in\mathcal{F}} f(X) \neq \infty$.

a_i in PT. As in definition, a separator Sep_{ij} includes the intersection of cliques X_{C_i}, X_{C_j} and hence, in this case $Sep_{ij} = X_{C_i} \setminus X_i$.

In this formulation we assume the cliques of the junction tree $\gamma(CG, PT)$ distributed among agents such that each agent $a_i \in A$ is assigned a single clique X_{C_i} and hence, the GDL message-passing scheme among cliques is indeed a message-passing scheme among agents in PT.

Figure 1(c) shows the γ-junction tree for the game in Figure 1(a) whose circles stand for cliques and edges (between cliques) stand for separators. Since the graph of this game is acyclic, the clique variables of any agent $a_i \in A$, X_{C_i} is equal to its potential domain, X_{ψ_i} , and strictly composed of a_i's local variables and all the local variables of a_i's parent whose corresponding coalition contains a_i ($\{x_S \in X_{p_i} | a_i \in S\}$). Thus, since a_0 is the root, its clique and potential domain is strictly composed of its local variables, namely $X_{C_0} = X_{\psi_0} = \{x_0, x_{01}, x_{12}\}$ whereas the clique and potential domain of a_1 is composed of its local variables, $\{x_1, x_12\}$, and all variables local to its parent a_0 whose corresponding coalition contains a_1, namely $\{x_{01}, x_{012}\}$. In contrast, the γ-junction tree for the game in Figure 2(a) depicted in Figure 2(d) whose graph contains a cycle, the potential domain of agent a_2 contains, in addition of a_2's local variables, variables that are not local to its parent, namely x_{02} that is local to a_0. Moreover, in this cyclic case the clique of an agent may contain more variables than the ones in its potential domain. For example, the clique of agent a_1 contains, in addition of variables in X_{ψ_1}, variable x_{02}. This variable is included in a_2's clique in order to satisfy the running intersection property of the junction tree.

Proposition 1. *$\gamma(CG, PT)$ is a junction tree for $R(CG, PT)$.*

PROOF. We prove proposition 1 by showing how $\gamma(CG, PT)$ satisfies the required *covering* and *running intersection* properties. Since $\forall a_i \in A : X_{\psi_i} \subseteq X_{C_i}$, $\{F_1 \cup \ldots \cup F_{|A|}\} = \mathcal{F}$ and $\forall_{i,j \in A} F_i \cap F_j = \emptyset$, covering is satisfied. Then, by definition 3, $\forall x_S \in X_i$ the set $X_S^{\setminus r}$ does not contain any variable local to any descendant of a_i in PT. Thus, any variable $x_S \in X_i$ does not appear in any set $X_j^{\setminus r}$ of any agent a_j ancestor of a_i in PT and does not need to be included by the running intersection to any clique of any ancestor of a_i. Therefore, for each agent $a_i \in A$ the inclusion of the set of variables $\bigcup_{j \in Ch_i} X_{C_i} \setminus X_j$ that contains by recursion all the variables that required some variable down a_i ($\bigcup_{a_j \in D_i} X_j^{\setminus r}$), excluding all variables local to agents down a_i ($\bigcup_{a_j \in D_i} X_j$) satisfies the running intersection.

Therefore, given a game on a graph $CG = \langle A(G), v, F(G)\rangle$ and a pseudotree PT of G over $A(G)$ the execution of a message-passing GDL algorithm, as the one reviewed in Section 2.2, over the junction tree $\gamma(CG, PT)$, recovers the optimal solution X^* of $R(CG, PT)$ and hence, by mapping Ω, the optimal coalition structure in CG.

More generally, given a game on a graph $CG = \langle A(G), v, F(G)\rangle$ and a pseudotree PT of G over $A(G)$, after the single-vertex GDL execution over $\gamma(CG, PT)$ is over, the *state* function of each agent $a_i \in PT$ recovers the value of the optimal coalition structure $CS^{*,i}$ of a subgame CG^i:

$$\max_{X_{C_i}} s_i(X_{C_i}) = \max_{X_{C_i}} \max_{\mathcal{X} \setminus X_{C_i}} \bigotimes_{j \in A(PT_i)} \psi_i(X_{C_i}) = v(CS^{*,i}) \quad (7)$$

where $CS^{*,i}$ stands for the best coalition structure that agent a_i and agents down a_i in PT can form among themselves, using a subset of coalitions of CG that do not contain any agent up a_i in PT. Hence, the state function of each agent $a_i \in PT$ recovers the value of the optimal coalition structure $CS^{*,i}$ of the subgame $CG^i = \langle A(PT_i), v, F(G_{A(PT_i)})\rangle$ composed of all the agents in PT_i (of a_i and its descendants in PT) and all the feasible coalitions in CG excluding those coalitions that require some agent up a_i in PT (that contain some ancestor of a_i). In the particular case where $A = A(G)$, at the root agent a_r, $v(CS^{*,r}) = v(CS^*)$ and the state function of a_r recovers the optimal coalition structure in CG.

More generally, given a game on a graph $CG = \langle A, v, F(G)\rangle$, $A \subseteq A(G)$, after the single-vertex GDL execution over $\gamma(CG, PT)$ is over, the *state* function of each agent $a_i \in PT$ recovers the value of the optimal coalition structure $CS^{*,i}$ of a subgame CG^i:

$$\max_{X_{C_i}} s_i(X_{C_i}) = \max_{X_{C_i}} \max_{\mathcal{X} \setminus X_{C_i}} \bigotimes_{j \in A(PT_i)} \psi_i(X_{C_i}) = v(\widetilde{CS}^{*,i}) \quad (8)$$

where $\widetilde{CS}^{*,i}$ stands for the best coalition structure that agent a_i and agents down a_i in PT can form, exhaustive and disjoint *with respect to agents in* $A(PT_i)$, using a subset of coalitions of CG that do not contain any agent up a_i in PT. Notice that in this case $\widetilde{CS}^{*,i}$ has not to be the same than the optimal coalition structure $CS^{*,i}$ of the subgame $CG^i = \langle A(PT_i), v, F(G_{\setminus An_i})\rangle$, because in any valid coalition structure for CG^i any two pair of coalitions are disjoint. Notice also that $v(\widetilde{CS}^{*,i}) \geq v(CS^{*,i})$.

Next, in section 4 we will show how to extend these GDL-based distributed message passing algorithms to the non-cooperative setting in order to, not only compute, but also incentive agents to form the optimal coalition coalition structure.

4. ALGORITHMS

Algorithm 1 allows agents in a game CG over a graph G to distributedly arrange into a junction tree of $R(CG, PT)$.

/*A requiring messages is composed of pairs of a requiring variable x_S and a set of required agents S' still not covered in S*/

4.1 Stable Coalition Formation on Trees

high level algorithm description

The *SCF-Trees*, whose pseudocode is outlined in Algorithm 4, has three phases: *preprocessing*, *demand propagation phase* and *offer propagation phase*.

First, agents start with a preprocessing phase (line 2, *TreeDecomposition* procedure) that compiles the problem into a junction tree of the compact representation (see section 3) to be used in the following two phases. In the preprocessing phase, agents arrange the graph into a pseudotree PT. Since in acyclic graphs the graph is already a tree, agents only need to root that tree. For example, in Figure **??** the tree of the game in Figure 1(a) is rooted at a_0. Then each agent a_i creates one binary variable x_S, along with a function with its value v_S, for each possible coalition that a_i can join with agents down the tree. For example, in Figure **??** a_0 creates three variables, namely x_0, x_{01}, x_{012}. Notice that the set composed of all these variables is X_i (e.g. $X_0 = \{x_0, x_{01}, x_{012}\}$). Then, each agent a_i waits for its parent's message that contains a set of tuples where each tuple is composed of a coalition up the tree x_S that require a set of agents S' in $A(PT_i)$. In an acyclic graph, these variables are singly composed of all parent's variables that contain a_i (X_{pi}). Then, for each variable $x_S \in X_{pi}$, each agent a_i creates a require function between x_S and $x_{G_{S'} \cap G_{A(PT_i)}}$. Thus, in Figure **??**, a_1 after receiving $\{\langle x_{01}, 1\rangle, \langle x_{012}, \{1, 2\}\rangle\}$ from a_0 creates two require functions, namely $r(x_{01}, x_1)$ and $r(x_{012}, x_{12})$. Then, each agent a_i communicates to each of its children $a_j \in Ch_i$

Algorithm 1 BuildJunctionTree(PT,)

Each agent a_i knows $\langle a_p, Ch_i, PT_i, v, \mathbf{S_i}\rangle$ and runs:

$X_i \leftarrow \{x_S | S \in \mathbf{S_i}\}$; /*Create the set of local variables, one per local coalition*/
$F_i \leftarrow \emptyset$; /*Initialize the set of local functions*/
$F_i \leftarrow \bigcup_{x_S \in X_i} f_v(x_S)$;/*Add for each local variable one value function encoding its value in v*/
$F_i \leftarrow F_i \cup f_u(X_i)$; /*Add a unique function that controls that only one variable in X_i activates*/
/*Then, agents run a procedure to compute for each of its local variables $x_S \in X_i$, the set of requiring variables $X_S^{\setminus r}$*/
if a_i is not the root /*Wait for parent's require message*/ **then**
  Wait for $Req_{p\rightarrow i}$ from a_p;
end if
for all $\langle x_S, A_S\rangle \in Req_{p\rightarrow i}$/*For each requiring variable x_S*/ **do**
  if $i \in A_S$ /*If a_i is in the required set of x_S*/ **then**
    $S' \leftarrow arg\max_{S''\in\mathbf{S_i}|S''\subseteq A_S \cap A(PT_i))\}} |S'' \cap A_S|$; /*Compute the local coalition $S'' \subset A_S$ with maximum intersection with $A_S \cap A(PT_i)$*/
    $F_i \leftarrow F_i \cup f_r(x_{S'}, x_S)$; /*Add a function between $x_{S'}$ and its requiring variable x_S*/
    for all $x_{S''} \in X_{S'}^{\setminus r}$ /*For each requiring variable of $x_{S'}$*/ **do**
      $F_i \leftarrow F_i \cup f_b(x_S, x_{S''})$; /*Add a blocking function between the two requiring variables x_S,$x_{S''}$ of $x_{S'}$*/
    end for
    $X_{S'}^{\setminus r} \leftarrow X_{S'}^{\setminus r} \cup x_S$; /*Add x_S as requiring variable of $x_{S'}$*/
    $Req_{p\rightarrow i}.\langle x_S, A_S\rangle = \langle x_S, A_S \setminus S'\rangle$; /*Update the set of required agents of x_S considering agents in S' covered*/
  end if
end for
for all $x_S \in X_i$/*Add requirements for local variables*/ **do**
  $Req_{p\rightarrow i} \leftarrow Req_{p\rightarrow i} \cup \langle x_S, S\rangle$;
end for
for all $a_j \in Ch_i$ /*Prepare a require message for each child*/ **do**
  $Req_{i\rightarrow j} \leftarrow \emptyset$;
  for all $\langle x_S, S'\rangle \in Req_{p\rightarrow i}$ /*For each requiring variable*/ **do**
    if $S' \cap A(PT_j) \neq \emptyset$/*If requires any agent in the subtree of a_j*/ **then**
      $Req_{i\rightarrow j} \leftarrow Req_{i\rightarrow j} \cup \langle x_S, S' \cap A(PT_j)\rangle$;/*Add the corresponding requirement*/
    end if
  end for
  Send $Req_{i\rightarrow j}$ to a_j;
end for
$\psi_i \leftarrow \bigotimes_{f\in F_i} f$ /*Compute potential function*/
return $\langle X_i, \bigcup_{x_S\in X_i} X_S^{\setminus r}, \psi_i\rangle$

a message that contains for each set of variables that include a_j, $x_S \in X_{ij}$, a tuple with x_S and the set of agents from S reachable from a_j, $S \cap A(PT_j)$. The intuition is that each agent a_j would act as a mediator negotiating the payment demanded by agents down PT_i to join a coalition x_S with a_i. Finally, each agent computes its local function f_i as the combination of: (i) function u_i that controls that one and only one of X_i coalition variables is activated (set to 1); (ii) value functions $\vec{v}$; and (iii) require functions $\vec{r}$ (note that in

Algorithm 2 DemandPropagation

for all $a_j \in Ch_i$ **do**
  Wait for the demand message $d_{j\rightarrow i}(Sep_{ji})$ from a_j;
end for
$p_i(X_{C_i}) = \psi_i(X_{\psi_i}) \otimes \bigotimes_{j\in Ch_i} d_{j\rightarrow i}(Sep_{ji})$; /*Compute the payment function*/
$\rho_i = \max_{X_{C_i}} p_i(X_{C_i})$; /*$a_i$ computes its payment */
if a_i is not the root /*Compute a demand message for $a_i's$ parent*/ **then**
  $d_{i\rightarrow p}(Sep_{ip}) = \max_{X_i} p_i(X_{C_i}) - \rho_i$;
  Send $d_{i\rightarrow p}(Sep_{ip})$ to a_p;
end if
return $\langle p_i, \rho_i\rangle$

Algorithm 3 ValuePropagation

Each agent a_i knows $\langle a_p, Ch_i, \bigcup_{j\in Ch_i} Sep_{ji}, p_i\rangle$ and runs:

if a_i is not the root **then**
  Wait for Sep^*_{ip} from a_p;
end if
$X^*_{C_i} = \arg\max_{X_{C_i}, Sep_{ip}=Sep^*_{ip}} p_i(X_{C_i})$; /*Compute best solution, slicing with respect the parent decision*/
for all $a_j \in Ch_i$ /*Send best coalition to each child*/ **do**
  Send $Sep^*_{ji} \leftarrow X^*_{C_i} \cap Sep_{ji}$ to a_j;
end for
return $X^*_{C_i}$

Algorithm 4 SCF-Trees ($\langle A, v, F(G)\rangle$)

Each a_i knows $\langle v, F_{\{i\}}(G)\rangle$ and runs:

/*Preprocessing phase*/
Pseudotree arrangment- run token based mechanism that arrange agents into a pseudotree PT
At completion, a_i knows a_p, Ch_i, PT_i;
$F \leftarrow$);
/*Junction tree arrangement*/
$\langle X_i, X_i^{\setminus r}, F_i\rangle \leftarrow$ buildJunctionTree($a_p, Ch_i, PT_i, v, F_{\{i\}}(G_{A(PT_i)})$);
/*Demand propagation phase*/
$\langle p_i, \rho_i\rangle \leftarrow$ DemandPropagation($a_p, X_i, X_i^{\setminus r}, Ch_i, f_i$);
/*Value propagation phase*/
$X^*_{C_i} \leftarrow$ ValuePropagation($a_p, p_i, Ch_i, \bigcup_{j\in Ch_i} Sep_{ji}$);
return $\langle \rho_i, X^*_{C_i}\rangle$

acyclic graphs the set of blocking functions is empty).

Hence, at the end of this preprocessing phase, each agent knows: (i) X_i; (ii) $\bigcup_{x_S\in X_i} X_S^{\setminus r}$ that in a tree is all the variables of a_p that includes a_i (X_{ip}); and (iii) local function f_i.

After this first processing phase is over, $SCF-Trees$ runs two phases:

- A *demand propagation phase* (line , *DemandPropagation* procedure), in which agents exchange demand messages up the tree.
- An *offer propagation phase* (line , *ValuePropagation* procedure), in which agents exchange offer messages down the tree.

When executing the demand exchange protocol, each agent a_i waits until receiving a demand message from each of its children

$a_j \in Ch_i$ (lines). The message that a_j sends to its parent a_i contains for each of the coalition variables x_S in X_{ij} the amount required for a_j and agents down T_j to join S. For example, in Figure **??**, agent a_0 waits until receiving the demand message from a_1 that contains a function over variables in their separator $\{x_{01}, x_{012}\}$. Upon receiving all demand messages, each agent a_i computes its payment function p_i as the combination of function f_i, that codifies its local utility and restrictions for variables $X_i; X_{ip}$, and the demand messages from the children, that substract the amount required for agents down a_i (line). Then, a_i computes its payment ρ_i as the highest payment a_i can get on any of these configurations (coalitions). After that, if agent a_i has a parent in PT (line), a_i sends a message to its parent a_p that summarizes its payment function p_i over X_{ip} variables, substracting its own payment ρ_i. The result of this summarization is for each coalition of X_{ip} the payment required from agents in $A(PT_i)$ to join that coalition. Thus, in Figure **??**, a_0 summarizes its payment function p_0 over variables $\{x_{01}, x_{012}\}$ (filtering out $X_1 = \{x_1, x_{12}\}$). Then, agents proceed to execute the offer propagation phase by executing the offer protocol (line).

During the offer exchange protocol, each agent a_i waits until receiving an offer message from its parent a_p specifying if a_p accepts for some coalition $x_S \in X_{ji}$ to pay the amount requested by agents in $A(PT_i)$ to join the coalition. Thus, in Figure , a_1 waits until receiving a message from a_0 with its decision with respect to coalitions x_{01}, x_{012}. Then, a_i computes the better coalition it can join given the decision of its parent a_p (line). If a_p decided to create a coalition $x_S \in X_{ip}$ agreeing on the payment required by a_i and other agents (S') down the tree to join S, then the best coalition for a_i is $x_{S'}$. In contrast, if a_p does not activate any variable $x_S \in X_{ip}$, a_i will select the best coalition a_i can join that includes itself and some agents down a_i in the tree. Then agent a_i sends an offer message to each of its children $a_j \in Ch_i$ that contains which coalition $x_S \in X_{ij}$ a_i accepted to create, if any, and the amount a_i can offer to agents down to join each coalition in X_{ij}.

4.1.1 Complexity and Correctness

Because the lack of space, in the following we just expose the results and sketching proofs. The interested reader can find far more detailed proofs in [**?**].

On the one hand, the complexity of SCF-Trees algorithm is linear to the size of the input (e.g. the number of feasible coalitions in G).

On the other hand, the correctness of SCF-Trees algorithm is assessed by the following theorem.

Theorem 3. *Given a game on a graph $CG = \langle A(G), v, F(G)\rangle$ where G is acyclic, the outcome produced by SCF-Trees(CG) belongs to the core of CG.*

PROOF. By Lemma 1 (*GDL state function equivalence*) the payment function $p_i(X_{C_i})$ computed by an agent $a_i \in A(G)$ during the execution of the DemandProtocol over the junction tree $\gamma(CG, PT)$ is equal to the state function $s_i(X_{C_i})$ computed during the execution of the single-vertex GDL algorithm over $\gamma(CG, PT)$ up to a normalisation constant (the sum of the payments of all the descendants of a_i in PT). Then, the set of optimal clique variables in both algorithms is the same and the value propagation phase in SCF-Trees computes a solution X^* that recovers the optimal coalition structure, CS^*, of CG ($\Omega(X^*) = CS^*$). Then, to prove Theorem we only need to show that the allocation ρ computed by agents $A(G)$ during the execution of the DemandProtocol satisfy the two core conditions, namely: (CS^*-Imputation) $\sum_{i \in A(G)} \rho_i = v(CS^*)$; and (Group rationality) $\forall S \subseteq F(G)$: $\rho_S \geq v(S)$. By lemma 2 ($CS^{*,i}$-*Imputation*), by setting a_i to the root agent a_r in PT, we have that ρ is an imputation of $CS^{*,r}$ where $CS^{*,r}$ is the optimal coalition structure of $CG^r = \langle A(PT), v, F(G_{A(PT)})\rangle$ and, since $A(PT) = A(G)$, ρ is an imputation of CS^*. By Lemma 3 (*Allocation on Trees is group rational*), ρ is group rational. Thus, the outcome of SCF-Trees $(\Omega(X^*), \rho)$ belongs to the core of CG and Theorem 3 holds.

Lemma 1 (*GDL state function equivalence*)**.** *Given a game on a graph $CG = \langle A, v, F(G)\rangle$ and a pseudotree PT of G over A, the payment function $p_i(X_{C_i})$ computed by any agent $a_i \in A$ satisfy that $\forall_{X_{C_i}} : p_i(X_{C_i}) = s_i(X_{C_i}) - \sum_{j \in D_i} \rho_j$ where ρ_j is the payment of agent a_j and $s_i(X_{C_i})$ is the state function of a_i in the execution of the single-vertex GDL algorithm over $\gamma(CG, PT)$.*

PROOF. $p_i(X_{C_i}) =_{\text{Proc.2}} \psi_i(X_{\psi_i}) \otimes \bigotimes_{j \in Ch_i} d_{j \to i}(Sep_{ji}) = \psi_i(X_{C_i}) \otimes \bigotimes_{j \in Ch_i} (\mu_{j \to i}(Sep_{ji}) - \sum_{k \in PT_j} \rho_k) =_{\text{Eq. 2}} s_i(X_{C_i}) - \sum_{j \in D_i} \rho_j$.

Lemma 2 ($CS^{*,i}$-*Imputation*)**.** *Given a game on a graph $CG = \langle A(G), v, F(G)\rangle$ and a pseudotree PT of G over $A(G)$, the allocation computed by agents $A(G)$ during the execution of the DemandPropagation (Procedure 2) over $\gamma(CG, PT)$ satisfy that $\forall a_i \in A$: $\sum_{j \in A(PT_i)} \rho_j = v(CS^{*,i})$, where $CS^{*,i}$ is the optimal coalition structure of $CG^i = \langle A(PT_i), v, F(G_{A(PT_i)})\rangle$.*

PROOF. $\rho_i =_{\text{Proc. 2}} max_{X_{C_i}} p_i(X_{C_i}) =_{\text{Lem. 1}} \max_{X_{C_i}} s_i(X_{C_i}) - \sum_{j \in D_i} \rho_j =_{\text{Eq. 7}} v(CS^{*,i}) - \sum_{j \in D_i} \rho_j$ and Lemma 2 holds.

Proposition 2. *Given a game on an* acyclic *graph $CG = \langle A(G), v, F(G)\rangle$ and a pseudotree PT of G over $A(G)$, the payment of any agent $a_i \in A$, ρ_i, computed by the DemandPropagation (Procedure 2) over $R(CG, PT)$ satisfy that:*

$$\begin{aligned}
\rho_i &= \max_{X_{C_i}} p_i(X_{C_i}) \underset{\textit{Obs. ??}}{=} \max_{X_i; X_{ip}} p_i(X_{C_i}) \\
&\underset{\textit{Obs. ??}}{=} \max_{X_i; X_{ip}} \psi_i(X_{\psi_i}) + \sum_{j \in Ch_i} \mu_{j \to i}(X_{ji}) - \rho_{A(PT_j)} \\
&\underset{\substack{\textit{Prop.??} \\ \textit{Lem.2}}}{=} \max_{S \in F_{\{i\}}(G)} v(S) + \sum_{j \in Ch_i} v(CS^{*,j \setminus S}) - v(CS^{*,j})
\end{aligned}$$

where, $CS^{*,j \setminus S}$ is the optimal coalition structure of $\langle A(PT_j) \setminus S, v, F(G_{A(PT_j) \setminus S})\rangle$ and $CS^{*,j}$ is the optimal coalition structure of $\langle A(PT_j), v, F(G_{A(PT_j)})\rangle$.

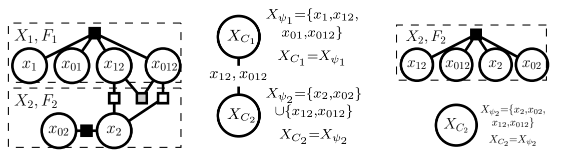
Notice that we can ignore the set X_{ip} in the computation of ρ_i because the value of activating any $x_S \in X_{ip}$ is lower than the value of the configuration that only activates its requested variable $x_{S'} \in X_i$ (when $x_S = 1$ the corresponding require function substracts the value of $x_{S'}$).

Lemma 3 (Allocation On Trees is Group Rational)**.** *Given a game over an* acyclic graph *$CG = \langle A(G), v, F(G)\rangle$ and a pseudotree PT of G over $A(G)$, the allocation computed by agents during the execution of the DemandPropagation (Procedure 2) over $\gamma(CG, PT)$ satisfy that: $\forall a_i \in A : \forall S \subseteq F(G_{A(PT_i)}) : \rho_S \geq v(S)$.*

PROOF. We prove Lemma 3 by induction on d, the rank of agent a_i in PT. In the base case $d = 1$ (a_i is a leaf) since the only coalition that a_i can compose by itself, $X_i = \{x_{\{a_i\}}\}$, and by Lemma 2 $\rho_i = v(\{a_i\})$, Lemma 3 holds. In the induction case we can split the pseudotree into the root agent and a set of subtrees of lower depth. Then, consider an agent a_i whose rank is $n + 1$ and assume that lemma 3 holds for all agents whose rank is less than or equal to n in the tree ($\forall a_j \in Ch_i$ $\forall S \in F(G_{A(PT_j)}) : \rho_S \geq v(S)$). Then, to prove Lemma 3, we only need to show that $\forall S \in F_{\{i\}}(G_{A(PT_i)}) : \rho_S \geq v(S)$. Recall that the payment of a_i computed in the DemandProtocol satisfies that $\rho_i = \max_{S \in F_{\{i\}}(G_{A(PT_i)})} v(S) + \sum_{j \in Ch_i} v(CS^{*,j \setminus S}) - \sum_{j \in Ch_i} \rho_{A(PT_j)}$ where $CS^{*,j\setminus S}$ is the optimal coalition structure in $\langle A(PT_j) \setminus S, v, F(G_{A(PT_j)\setminus S})\rangle$. Thus, when computing its payment on a coalition S, a_i: (i) substracts to the value of S the payment of all agents down a_i in PT; and (ii) since as we will show next $\forall_{j \in Ch_i} : v(CS^{*,j\setminus S}) == \rho_{A(PT_j)\setminus S}$, adds the payment of all the agents down a_i not in S. As a result, when computing its payment in each coalition S, a_i substracts to the value of S the payment of other agents in S, $\rho_i = \max_{S \in F_{\{i\}}(G_{A(PT_i)})} v(S) - \rho_{S\setminus\{i\}}$, and since the agent payment, ρ_i, is the maximum among payments on local coalitions, the group rationality condition $\forall S \in F_{\{i\}}(G_{A(PT_i)}) : \rho_{S\setminus\{i\}} + \rho_i \geq v(S)$ is satisfied.

We prove that $\forall j \in Ch_i \; v(CS^{*,j\setminus S}) == \rho_{A(PT_j)\setminus S}$ by observing that: (i) since G is acyclic, S forms a continuous subtree down a_i, so if we exclude agents S from each PT_j the result is a set of subtrees $\{PT_l\}$ where a_l stands for the agent with highest level in branch l not included in S and PT_l includes all the agents down a_l in PT; and (ii) by Lemma 2 $v(CS^{*,l}) = \rho_{PT_l}$.

4.2 Stable Coalition Formation on Graphs



(a) $R(CG_{\{1,2\}}, PT_1)$ (b) $\gamma(G_{\{1,2\}}, PT_1)$ (c) $R(G_{\{2\}}, PT_2)$

Figure 3: a) Representation of CG after eliminating a_0; b) γ-JT for (a); and c) and representation and γ-JT of CG after eliminating a_0, a_2.

High description of the algorithm In this section we introduce the SCF-Graphs algorithm that allows agents to . SCF-Graphs is based on extending , which performs n iterations on a tree decomposition. SCF-Graphs, operates on a variable ordering which is given by a DFS arrangement of the problem graph. In phase, a DFS traversal of the graph is done using Algorithm . The DFS tree thus obtained serves as a communication structure for the other 2 phases of the algorithm: UTIL propagation (UTIL messages travel bottom-up on the tree), and VALUE propagation (VALUE messages travel top-down on the tree).

SCF-Graphs extends the first three phases of the algorithm resembles the SFC-Trees algorithm. The first phase . As we will see .The second phase agents execute the DemandProtocol over a phase of the original graph. In the third phase they recover the optimal coalition structure. SCF-Graphs however then runs A. phases from.

First phase is composed of generate pseudoline, generate TreeDecomposition and DemandProtocol execution. Agents start by arranging them on an agent ordering which is given by a line arrangement of its graph.

The SCF-Graphs, whose pseudocode is outlined in Algorithm , has three phases.

Idea we can explain the algorithm saying that it runs on n iterations. The first iteration, run by all agents, executes three phases reesembling the operation of SCF-Trees: a junction tree is built from a pseudotree and a demandProtocol and a value propagation protocol are run. Then at the next n-1 iterations,

Given a coalitional game $CG = \langle A(G), v, F(G)\rangle$, SCF-Graphs runs on $|A|$ iterations. The first iteration, run by all agents, resembles the operation of SCF-Trees and has three phases: Preprocessing, Demand Propagation and Value Propagation. During the preprocessing phase agents distributedly arrange the coalitional game into a junction tree $\gamma(CG, L)$ where L is restricted to be pseudoline, a pseudotree in which all agents have a single child. This restriction easies the formulation of the algorithm and the corresponding correctness proofs presented in Section , while it does not limit its applicability since any graph G can be arranged into a pseudoline. Nevertheless, we point out here that this restriction may limit the efficiency of the Algorithm since the complexity of the demandProtocol depends on the induced treewidth of the junction tree, and we restrict to a subclass of junction trees that are built on a pseudoline. start First, agents start with a preprocessing phase (line) that arranges them into a line (a line is defined as an ordering among agents). Notice that this line ordering of agents defines a valid pseudotree of G over A (since all the agents are in the same branch). At completion, each agent knows its ancestor in the ordering (its parent in the pseudotree), a_p, as well as all the agents that are placed next to a_i in the line with its ordering (L_i). Then, the agent retrieves the next agent in the line, a_j, that stands for its (only) child in the pseudotree. For example, in Figure , agents in the game use a (line) pseudotree defined by an ordering $a_0 > a_1 > a_2$, that is rooted at a_0.

Then agents proceed to execute the first iteration composed of three phases. In the first phases, agents distributedly build a junction tree of the representation of the game, $\gamma(R(\langle A, v, F(G)\rangle), L)$ (, Procedure TreeDecomposition).

In this section we restrict to pseudotrees that place all agents in the same branch, and we call it pseudoline over A.

4.2.1 Complexity and Correctness

On the one hand, the complexity of SCF-Graphs algorithm is exponential to the treewidth of the junction tree over CG.

On the other hand, the correctness of SCF-Graphs algorithm is assessed by the following theorem.

Theorem 4. *Given a game on a graph $CG = \langle A(G), v, F(G)\rangle$, if the core of CG is not empty, the outcome produced by SCF-Graphs(CG) (Algorithm 5) belongs to the core of CG; otherwise SCF-Graphs(CG) outcomes the optimal coalition structure of CG detecting the emptiness of the core.*

Before proving Theorem 4, we define a set of characteristic functions, one per agent $a_i \in A$, $\{v_{PT_i} | a_i \in A\}$.

Definition 7. *Given a game $CG = \langle A, v, F(G)\rangle$ and a pseudoline PT over A in G, for any agent $a_i \in A$ $v_{PT_i} : F(G) \to \Re$ is a characteristic function inductively defined, starting from the root agent a_r in PT for which $v_{PT_r} = v$, as:*

$$v_{PT_i}(S) = \begin{cases} v_{PT_p}(S), & \text{if } a_p \notin S \\ v_{PT_p}(S) - \rho_p^p & \text{if } a_p \in S \end{cases}$$

where a_p is the parent of a_i in PT and ρ_p^p stands for the payment of a_p computed by the DemandPropagation (Procedure 2) over $R(CG_p, PT_p)$, $CG_p = \langle PT_p, v_{PT_p}, F(G)\rangle$.

Next, we provide the proof for Theorem 4.

PROOF. Theorem 4 follows from lemmas 2, 4 and 6. During the execution of SCF-Graphs(CG), agents execute $|A|$ iterations of the DemandPropagation procedure. At each iteration t, agents execute the DemandPropagation over $\gamma(R(CG_i, PT_i))$ where i stands for the index of the agent at rank t in PT and $CG_i = \langle A(PT_i), v_{PT_i}, F(G)\rangle$. In the first iteration, the execution of the DemandPropagation is over $\gamma(R(CG, PT))$ and it is followed by a ValuePropagation procedure in which each each agent a_i assesses

Algorithm 5 SCF-Graphs ($\langle A, v, F(G)\rangle$)

Each a_i knows $\langle v, F_{\{i\}}(G)\rangle$ and runs:
/*Preprocessing phase*/
Line ordering arrangement - run token based mechanism that arrange agents into a line L.
At completion, a_i knows a_p, L_i;
$a_j \leftarrow$ next agent in L_i;
First iteration
$F \leftarrow F_{\{i\}}(G_{A(PT_i)})$;
$\langle X_i, X_i^{\setminus r}, f_i\rangle \leftarrow$ TreeDecomposition$(a_p, \{a_j\}, PT_i, v, F)$;
/*Demand propagation phase*/
$\langle p_i, \rho_i\rangle \leftarrow$ DemandPropagation$(a_p, X_i, X_i^{\setminus r}, \{a_j\}, f_i)$;
/*Value propagation phase*/
$X^*_{C_i} \leftarrow$ ValuePropagation$(a_p, p_i, \{a_j\}, \bigcup_{j\in Ch_i} Sep_{ji})$;
loop
  if $a_p = \emptyset$ /*if a_i is the root*/ **then**
    if $p_i(X^*_{C_i}) \neq \rho_i)$ **then**
      $\rho_i \leftarrow -\infty$; /*Core is empty*/
    end if
    for $x_S \in X_i$ **do**
      $k \leftarrow \arg\min_{l\in S\cap A(PT_j)} level(a_l, PT_j)$ /*Find agent in S with highest position in PT_j*/
      Send $o_{i\to k} \leftarrow \langle S, v(x_S) - \rho_i\rangle$ to a_k;
    end for
    return $\langle \rho_i, X^*_{C_i}\rangle$
  else
    WaitForOfferMessages();
    /*Rebuild junction tree for that phase*/
    $\langle X_i, X_i^{\setminus r}, f_i\rangle \leftarrow$ TreeDecomposition$(a_p, \{a_j\}, PT_i, v, F)$;
    /*Demand propagation phase*/
    $\langle p_i, \rho_i\rangle \leftarrow$ DemandPropagation$(a_p, X_i, X_i^{\setminus r}, \{a_j\}, f_i)$;
  end if
end loop
Procedure WaitForOfferMessages()
On receipt of $o_{k\to i} = \langle S, v\rangle$
if $k = p$ /*Offer received from parent*/ **then**
  $a_p \leftarrow \emptyset$;
end if
$F \leftarrow F \cup S$; /*Add S to the set of local coalitions*/
$v(S) \leftarrow v$; /*Update the value for S*/
End Procedure

the optimal values for variables in its clique $X^*_{C_i}$. Since in CG, $A = A(G)$, analogously to the SCF-Trees(CG) algorithm, by observation **??**, these optimal solution computed during this phase recovers CS^*, the optimal coalition structure of CG ($\Omega(\bigcup_{a_i\in A} X_{C_i}) = CS^*$). Moreover, at each iteration t, each agent a_i, where a_i stands for the agent placed at rank t of the pseudoline PT, checks for equality $\rho_i = p_i(X^*_{C_i})$, and if not, detects the core as empty. By Lemma 2, $\rho^i_i = v_{PT_i}(CS^*_i) - \sum_{j\in Ch_i} \rho_{A(PT_j)}$ and by observation **??**, $p_i(X_{C_i}) = s_i(X_{C_i}) - \sum_{j\in Ch_i} \rho_{A(PT_j)}$ and hence, checking for $\rho_i = p_i(X^*_{C_i})$ is equivalent to checking for $v_{PT_i}(CS^*_i) = v_{PT_i}(CS^*)$, and by Lemma 6 the SCF-Graphs(CG) correctly detects the emptiness of the core. Otherwise, a_i fixes its payment ρ_i to the ones obtained at that iteration. Hence, the outcome of the SCF-Graph(CG), if $\forall a_i \in A : v_{PT_i}(CS^*_i) = v_{PT_i}(CS^*)$, is an allocation $\rho = \{\rho^1_1, \ldots, \rho^{|A|}_{|A|}\}$ where ρ^i_i stands for the payment of a_i computed by the DemandPropagation over $\gamma(R(CG_i, PT_i))$, $CG_i = \langle A(PT_i), v_{PT_i}, F(G)\rangle$. By Lemma 4 (when setting a_i to the leaf agent on PT), $(CS^*, \rho) \in Core(CG)$.

Observation 1. *Given a game on a graph $CG = \langle A, v, F(G)\rangle$ and a pseudoline PT over A in G, the payment of the root agent in PT_i, ρ_i, computed by the DemandPropagation over $R(CG_i, PT_i)$ where $CG_i = \langle A(PT_i), v_{PT_i}, F(G)\rangle$ satisfy that:*

$$\begin{aligned}\rho_i \; & \underset{Proc.2}{=} \max_{X_{C_i}} p_i(X_{C_i}) \underset{Obs.\,??}{=} \max_{X_i} p_i(X_i) \\ & \underset{Obs.\,??}{=} \max_{X_i} f_i(X_i) + \mu_{j\to i}(Sep_{ji}) - \rho_{A(PT_j)} \\ & \underset{\substack{Prop.??\\ Lem.2}}{=} \max_{S\in F_{\{i\}}(G)} v_{PT_i}(S) + v_{PT_i}(CS_i^{*,j\setminus(S\cap PT_i)}) - v_{PT_i}(CS_i^{*,j})\end{aligned}$$

where a_j is the (only) child of a_i in PT, $CS_i^{*,j\setminus(S\cap A(PT_i))}$ is the optimal coalition structure of $\langle A(PT_j)\setminus(S\cap A(PT_i)), v_{PT_i}, F(G_{\setminus(An_j\cup(S\cap A}$ and $CS_i^{*,j}$ is the optimal coalition structure of $\langle A(PT_j), v_{PT_i}, F(G_{\setminus An_j})\rangle$.

Lemma 4. *Given a game $CG = \langle A, v, F(G)\rangle$ and a pseudoline PT over A in G, if $\forall a_i \in A : v_{PT_i}(CS^{*,i}) = v_{PT_i}(CS^*)$ the allocation $\rho = \{\rho^1_1, \ldots, \rho^{|A|}_{|A|}\}$ where ρ^i_i stands for the payment of a_i computed by the DemandPropagation (Procedure 2) over $R(CG_i, PT_i)$, $CG_i = \langle A(PT_i), v_{PT_i}, F(G)\rangle$, satisfies that $\forall a_i \in A : v(CS^*) - \rho_{An_i} = \rho^i_{A(PT_i)}$ & $\forall S \in F(G_{An_i\cup\{i\}}) : \rho_S \geq v(S)$.*

PROOF. We prove Lemma 4 by induction on l, the level of a_i in PT. In case $l = 1$ a_i is the root and hence $a_i = a_r$, $A(PT_r) = A$, $CG^r = CG$ and $\{r\}$ is the only coalition that a_r can compose with its ancestors in PT (since a_r has no ancestor on PT). By observation **??**, $\rho^r_r = v(CS^*) - \rho^r_{A\setminus\{r\}}$. Moreover, by observation 1, $\rho^r_r = \max_{S\in F_{\{r\}}(G)} v_S(S) + v(CS^{*,\setminus S}) - v(CS^{*,\setminus\{r\}}) = max(v_S(\{r\}), \max_{S\in F_{\{r\}}(G)} v_S(S) + v(CS^{*,\setminus S}) - v(CS^{*,\setminus\{r\}}))$ and $p^r_r \geq v(\{r\})$ holds.

In the induction case, consider an agent a_i whose level is $n+1$ and assume that Lemma 4 holds for all ancestors of a_i in PT (An_i). By induction hypothesis, $\forall S \in F(G_{An_i}) : \rho_S \geq v(S)$. Since $F_{\{i\}}(G_{An_i\cup\{i\}}) \cup F(G_{An_i}) = F(G_{An_i\cup\{i\}})$, to prove Lemma 4 in this case we need to prove that $v(CS^*) - \rho_{An_i} = \rho^i_{A(PT_i)}$ & $\forall S \in F_{\{i\}}(G_{An_i\cup\{i\}}) : \rho_{S\setminus\{i\}} + \rho^i_i \geq v(S)$. By observation **??**, $\rho^i_i = v_{PT_i}(CS^*_i) - \rho^i_{A(PT_i)\setminus\{i\}}$. Then, by Lemma $v_{PT_i}(CS^*_i) = v_{PT_i}(CS^*)$ and by definition of v_{PT_i}, $\rho^i_i = v(CS^*) - \rho_{An_i} - \rho^i_{A(PT_i)\setminus\{i\}}$. Moreover, by observation 1 $\rho^i_i = \max_{S\in F_{\{i\}}(G)} v_{PT_i}(S)$ $v_{PT_i}(CS_i^{*,j\setminus(S\cap A(PT_i))}) - v_{PT_i}(CS_i^{*,j})$ where a_j is the (only) child of a_i in PT, $CS_i^{*,j\setminus(S\cap A(PT_i))}$ is the optimal coalition structure of $\langle A(PT_j)\setminus(S\cap A(PT_i)), v_{PT_i}, F(G_{\setminus(An_j\cup(S\cap A(PT_i))))})\rangle$ and $CS_i^{*,j}$ the optimal coalition structure of $\langle A(PT_j), v_{PT_i}, F(G_{\setminus An_j})\rangle$. Thus, $CS_i^{*,j}$ is the best coalition structure among those that can be composed using coalitions that do not contain any agent up a_j in $A(PT_i)$ (since a_i is the root that means that do not contain agent a_i), while being exhaustive and disjoint with respect to agents in $A(PT_j)$ (the optimal coalition structure of $\langle A(PT_j), v_{PT_i}, F(G_{\setminus\{i\}})\rangle$) and $CS_i^{*,j\setminus(S\cap A(PT_i))}$ is the best coalition structure that can be composed of feasible coalitions in G that do not contain any agent up a_i in $A(PT_i)$ (that they do not contain a_i) or in $S\cap A(PT_i)$ disjoint and exhaustive with respect to agents in $PT_j \setminus (S \cap A(PT_i))$. If $S \in F_{\{i\}}(G_{An_i\cup\{i\}})$, then S is exclusively composed of $\{a_i\}$ and ancestors of a_i in PT ($S\cap A(PT_i) = \{i\}$) and hence, $v_{PT_i}(S) = v(S) - \rho_{S\setminus\{i\}}$, $\langle A(PT_j)\setminus(S\cap A(PT_i)), v_{PT_i}, F(G_{\setminus(An_j\cup(S\cap A(PT_i))))})\rangle = \langle A(PT_j), v_{PT_i}, F(G_{\setminus\{i\}})\rangle$ and $v_{PT_i}(CS_i^{*,j\setminus(S\cap A(PT_i))}) = v_{PT_i}(CS_i^{*,j})$. Thus, $\forall S \in F_{\{i\}}(G_{An_i\cup\{i\}})$: $\rho^i_i \geq v(S) - \rho_{S\setminus\{i\}}$ holds.

Lemma 5. *Given a game $CG = \langle A, v, F(G)\rangle$ and a pseudoline PT over A in G, the allocation $\rho = \{\rho^1_1, \ldots, \rho^{|A|}_{|A|}\}$ where ρ^i_i*

stands for the payment of a_i computed by the DemandPropagation (Procedure 2) over $R(CG_i, PT_i)$, $CG_i = \langle A(PT_i), v_{PT_i}, F(G)\rangle$, satisfies that $\forall a_i \in A$: if $Core(CG) \neq \emptyset$ & $v_{PT_i}(CS_i^*) = v_{PT_i}(CS^*)$ then $\exists \rho'(A(PT_i) \setminus \{i\}) : (CS^*, \rho'(A(PT_i) \setminus \{i\}) \cup \{\rho_i^i\} \cup \rho(A(PT_i))) \in Core(CG)$.

PROOF. We prove Lemma 5 by induction on l, the level of a_i in PT.

In case $l = 1$, a_i is the root ($a_i = a_r$), $A(PT_r) = A$, $CG_r = CG$. We prove this case by contradiction. Consider that the core exists for some allocation with $\rho_r'^r$, $\rho_r'^r \neq \rho_r^r$ but not for ρ_r^r. Then, we consider two cases: (i) the core exists for some $\rho_r'^r > \rho_r^r$ but not for ρ_r^r; or (ii) the core exists for some $\rho_r'^r < \rho_r^r$ but not for ρ_r^r.

Let's consider (i). By observation **??** $\rho_r^r = v(CS^*) - v(CS^{*,\setminus\{r\}})$ where $CS^{*,\setminus\{r\}}$ is the best coalition structure that agents down a_r ($A \setminus \{r\}$) can form without a_r. Then, since $\rho_r'^r > \rho_r^r$, $v(CS^*) - \rho_r'^r < v(CS^{*,\setminus\{r\}})$ leading to the contradiction that any imputation with $\rho_r'^r > \rho_r^r$ can not be in the core because exists a set of coalitions for which agents in $A \setminus \{r\}$ can deviate. Now consider case (ii). In this case that the core does not exists for an imputation with p_r^r means that $\nexists \rho'(A \setminus \{r\})$ such that $\rho'_{A\setminus\{r\}} = v(CS^*) - \rho_r^r$ & $\forall S \in F_{\{r\}}(G)$: $\rho_r^r \geq v(S) - \rho'_{S\setminus\{r\}}$. Next, we show that actually the set of payments $\rho^r(A \setminus \{r\}$ are a counterexample for this ($\rho'(A \setminus \{r\}) = \rho^r(A \setminus \{r\})$ these conditions are satisfied) leading case (ii) to a contradiction. By Lemma 2, $\rho^r_{A\setminus\{r\}} = v(CS^*) - \rho_r^r$ is satisfied. By observation 1, $\rho_r^r = \max_{S\in F_{\{r\}}(G)} v(S) + v(CS^{*,\setminus S}) - v(CS^{*,\setminus\{r\}})$. By Lemma 2, $\rho^r_{A\setminus\{r\}} = v(CS^{*,\setminus\{r\}})$. Moreover, $\forall S \in F_{\{r\}}(G) : v(CS^{*,\setminus S}) \leq \rho^r_{A\setminus S}$ because $v(CS^{*,\setminus S}) = \rho^r_{A\setminus S}$ only if agents $A \setminus S$ are placed in the lowest $|S|$ positions of PT; otherwise these payments are higher. Thus, $\forall S \in F_{\{r\}}(G) : v(CS^{*,\setminus\{r\}}) - v(CS^{*,\setminus S}) \geq p_S^r$ and case (ii) leads to a contradiction.

In the induction case, consider an agent a_i whose level is $n + 1$ and assume that Lemma 5 holds for all ancestors of a_i in PT (if the core is not empty must exist some imputation in the core for $\rho(An_i)$). Then, to prove Lemma 5 requires to prove that ρ_i^i is a payment such that if the core is not empty must exist some imputation with $\rho(An_i) \cup \rho_i^i$. We prove this by contradiction. Consider that the core exists for some allocation with $\rho_i'^i \cup \rho(An_i)$, $\rho_i'^i \neq \rho_i^i$ but not for $\rho_i^i \cup \rho(An_i)$. We consider this in two separate cases: (i) the core exists for an imputation with $\rho_i'^i \cup \rho(An_i)$ such that $\rho_i'^i > \rho_i^i$, but not for $\rho_i^i \cup \rho(An_i)$; or (ii) the core exists for an imputation with $\rho_i'^i \cup \rho(An_i)$ such that $\rho_i'^i < \rho_i^i$, but not for $\rho_i^i \cup \rho(An_i)$. Let's consider case (i). By observation **??**, $\rho_i^i = v_{PT_i}(CS_i^*) - v_{PT_i}(CS_i^{*,\setminus\{i\}})$ where CS_i^* is the optimal coalition structure of CG_i and $v_{PT_i}(CS_i^{*,\setminus\{i\}})$ is the optimal coalition structure that agents in $A(PT_i)$ can obtain in CG_i without a_i (the optimal coalition structure in $\langle A(PT_i) \setminus \{i\}, v_{PT_i}, F(G_{\setminus\{a_i\}})\rangle$). By assumption $v_{PT_i}(CS^*) = v_{PT_i}(CS^*)$ and by definition of v_{PT_i}, $v_{PT_i}(CS_i^*) = v(CS^*) - \rho_{An_i}$. Then, since $\rho_i'^i > \rho_i^i$, $v(CS^*) - \rho_{An_i} - \rho_i'^i < v_{PT_i}(CS_i^{*,\setminus\{i\}})$ leading to the contradiction that any imputation with $\rho_i'^i \cup \rho(An_i)$ such that $\rho_i'^i > \rho_i^i$ can not be in the core because exists a set of coalitions for which agents in $A(PT_i) \setminus \{i\}$ can deviate. Now consider case (ii). In this case that the core does not exists for an imputation with p_r^r means that $\nexists \rho'(A(PT_i) \setminus \{i\})$ such that $\rho'_{A(PT_i)\setminus\{i\}} = v(CS^*) - \rho_i^i - \rho(An_i)$ & $\forall S \in F_i(G)$: $p_i^i \geq v(S) - \rho_{S\cap An_i} - \rho'_{S\cap(A(PT_i)\setminus\{i\})}$. Next, we show that actually the set of payments $\rho(A(PT_i) \setminus \{i\})$ are a counterexample for this ($\rho'(A(PT_i) \setminus \{i\}) = \rho^i(A(PT_i) \setminus \{i\})$ these conditions are satisfied) leading case (ii) to a contradiction. By Lemma 2, assumption $v_{PT_i}(CS^*) = v_{PT_i}(CS^*)$ and definition of v_{PT_i}, $\rho^i_{A(PT_i)\setminus\{i\}} = v(CS^*) - \rho_i^i - \rho_{An_i}$ is satisfied. By observation 1, $\rho_i^i = \max_{S\in F_{\{i\}}(G)} v_{PT_i}(S) + v_{PT_i}(CS_i^{*,\setminus S}) - v_{PT_i}(CS_i^{*,\setminus\{i\}})$. By definition of v_{PT_i}, $v_{PT_i}(S) = v(S) - \rho_{S\cap An_i}$. By Lemma 2, $\rho^i_{S\cap(A(PT_i)\setminus\{i\})} = v_{PT_i}(CS_i^{*,\setminus\{i\}})$. Moreover, $\forall S \in F_{\{i\}}(G) : v_{PT_i}(CS_i^{*,\setminus S}) \leq \rho^i_{PT_i\setminus S}$ because $v_{PT_i}(CS_i^{*,\setminus S}) = \rho^i_{A(PT_i)\setminus S}$ only if agents $A(PT_i) \setminus S$ are placed in the lowest $|S|$ positions of PT; otherwise these payments are higher. Thus, $\forall S \in F_{\{i\}}(G) : v_{PT_i}(CS^{*,\setminus\{i\}}) - v_{PT_i}(CS^{*,\setminus S}) \geq \rho_S^i$ and case (ii) leads to a contradiction.

Lemma 6. *Given a game on a graph $CG = \langle A, v, F(G)\rangle$ and a pseudoline PT over A in G then if $\exists a_i \in A : v_{PT_i}(CS_i^*) \neq v_{PT_i}(CS^*)$ where CS_i^* is the optimal coalition structure of $CG_i = \langle A(PT_i), v_{PT_i}, F(G)\rangle$ and CS^* is the optimal coalition structure of CG then the core of CG is empty.*

PROOF. Without lost of generality let's assume that a_i is the agent with highest level in PT for which $v_{PT_i}(CS_i^*) \neq v_{PT_i}(CS^*)$ ($\forall a_j \in An_i$ $v_{PT_j}(CS_i^*) = v_{PT_j}(CS^*)$ is satisfied). Then, we consider $v_{PT_i}(CS_i^*) \neq v_{PT_i}(CS^*)$ in its two separate cases: (i) $v_{PT_i}(CS_i^*) < v_{PT_i}(CS^*)$ and (ii) $v_{PT_i}(CS_i^*) > v_{PT_i}(CS^*)$.

Case (i) clearly leads to a contradiction: if CS_i^* is the optimal coalition structure in CG_i it can not exist another coalition structure CS^* whose value in v_{PT_i} is greater than those of CS_i^*. Now consider case (ii). If $v_{PT_i}(CS_i^*) > v_{PT_i}(CS^*)$ then $CS_i^* \neq CS^*$ and since $v_{PT_i}(CS^*) = v(CS^*) - \rho_{An_i}$ that means that exists a set of coalitions CS_i^* for which agents in $A(PT_i)$ can get better payments than not the best payments they can obtain in CS^* fixed the payments of $a_i's$ ancestors in PT (An_i) as in the allocation ρ. Such coalition structure can only exists iff: (i) payments fixed for a_i's ancestors (An_i) can not lead to any allocation in the core (even when $\forall a_j \in An_i$ $v_{PT_j}(CS_i^*) = v_{PT_j}(CS^*)$); or (ii) the core is empty. Because (i) leads to a contradiction with Lemma 5, the core is empty and Lemma 6 holds.

5. REFERENCES

[1] Grid 2030. a national vision for electricityï£¡s second 100 years. Washington DOE, 2003. http://www.oe.energy.gov/DocumentsandMedia/Electric_Vision_Docum

[2] S. M. Aji and R. J. McEliece. The generalized distributive law. *IEEE Transactions on Information Theory*, 46(2):325–343, 2000.

[3] Y. Bachrach, R. Meir, K. Jung, and P. Kohli. Coalitional Structure Generation in Skill Games. In *Proc. of AAAI-2010*, 2010.

[4] R. I. Brafman, C. Domshlak, Y. Engel, and M. Tennenholtz. Transferable utility planning games. In *Proc. of AAAI-2010*, 2010.

[5] V. Conitzer and T. Sandholm. Complexity of determining nonemptiness of the core. In *ACM Conference on Electronic Commerce*, pages 230–231, 2003.

[6] V. D. Dang and N. R. Jennings. Generating coalition structures with finite bound from the optimal guarantees. In *3rd International Conference on Autonomous Agents and Multi-Agent Systems*, pages 564–571, 2004.

[7] G. Demange. On Group Stability in Hierarchies and Networks. *Journal of Political Economy*, 112(4):754–778, August 2004.

[8] X. Deng and C. Papadimitriou. On the complexity of cooperative solution concepts. *Mathem. of Operation Research*, 19:257–266, 1994.

[9] G. Greco, E. Malizia, L. Palopoli, and F. Scarcello. On the complexity of compact coalitional games. In *IJCAI*, pages 147–152, 2011.

[10] S. Ieong and Y. Shoham. Marginal Contribution Nets: A Compact Representation Scheme for Coalitional Games. In *Proc. of ACM-EC'05*, pages 193 – 202, 2005.

[11] M. Jackson. A Survey of Models of Network Formation: Stability and Efficiency. Game Theory and Information 0303011, EconWPA, Mar. 2003.

[12] T. Michalak, J. Sroka, T. Rahwan, M. Wooldridge, P. Mcburney, and N. Jennings. A distributed algorithm for anytime coalition structure generation. In *Autonomous Agents And MultiAgent Systems (AAMAS 2010)*, pages 1007–1014, May 2010.

[13] R. Myerson. Graphs and Cooperation in Games. *Mathematics of Operations Research*, 2(3):225–229, 1977.

[14] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, Cambridge MA, USA, 1994.

[15] A. Petcu and B. Faltings. A scalable method for multiagent constraint optimization. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)*, pages 266–271, 2005.

[16] T. Rahwan, S. Ramchurn, N. Jennings, and A. Giovannucci. An anytime algorithm for optimal coalition structure generation. *Journal of Artificial Intelligence Research (JAIR)*, 34:521–567, April 2009.

[17] A. Rogers, A. Farinelli, R. Stranders, and N. R. Jennings. Bounded approximate decentralised coordination via the max-sum algorithm. *Artificial Intelligence*, 175(2):730–759, February 2011.

[18] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohmé. Coalition structure generation with worst case guarantees. *Artif. Intell.*, 111(1-2):209–238, 1999.

[19] R. van den Brink. On Hierarchies and Communication. Tinbergen Discussion Paper 06/056-1, Tinbergen Institute and Free University, 2006.

[20] M. Vinyals, J. A. Rodriguez-Aguilar, and J. Cerquides. Constructing a unifying theory of dynamic programming dcop algorithms via the generalized distributive law. *In Journal of Autonomous Agents and Multi Agent Systems (JAAMAS)*, 22(3):439–464, 2011.

[21] T. Voice, M. Polukarov, and N. R. Jennings. Graph coalition structure generation. *CoRR*, abs/1102.1747, 2011.