Coalition Structure Generation for Self-Interested Agents in a Dialogue Game

Luke RILEY [a,1], Katie ATKINSON [a] and Terry PAYNE [a]

[a] *Department of Computer Science, University of Liverpool, UK.*

Abstract. Since the seminal work of Dung, Argumentation Frameworks have been shown to find solutions to *n*-person cooperative games. In multi-agent systems, decentralised methods for multi-agent system coalition structure generation have been proposed. This paper offers the first dialogue game that utilises argumentation frameworks to find a coalition structure and a payoff vector in a decentralised manner. The payoff vector found is in the *core* set of stable solutions if the *core* is non-empty. This dialogue game also puts restrictions on the payoff vectors that can be proposed so that the most unfair ones are discarded. Lastly an algorithm is described that allows the agents to find out if the *core* is empty.

Keywords. Dialogue games, argumentation, coalition structure generation, cooperative game theory

1. Introduction

Coalition formation is the process of agents recognising that cooperation between other agents can occur in a mutually beneficial manner and then choosing an appropiate set of agents, named a *coalition*, to collaborate with to achieve some goal (or complete some task).

Forming coalitions in multi-agent systems (MAS) has been shown to be an important topic, for example [11] details it has proved useful in: e-commence (where coalitions can take advantage of price discounts); e-business (where coalitions can form to satisfy market niches); distributed sensor networks (where coalitions can form to track targets); and distributed vehicle routing (where coalitions of delivery companies can form to reduce costs). The cost of finding the best coalitions that satisfy all agents can be high, both computationally, and in terms of the necessary communication overhead due to the exponential number of possible coalitions and the possible self-interested behaviour of agents.

Coalition formation takes place in *n-person cooperative games* originally defined in [18]. The payoff for a coalition is traditionally measured numerically in *characteristic function games* where the value of a coalition is not influenced by the other coalitions in the system [6,9,18]. In *transferable utility n-person cooperative games (TU games)*, a *payoff vector* x is then used to distribute the group's payoff to the individual agents. A *coalition structure* (*CS*) is a set of coalitions in a system and finding an *optimal coali-*

[1] Correspondence to: Department of Computer Science, University of Liverpool, L69 3BX, UK. Tel.: +44 (0)151 795 54271; E-mail: L.J.Riley@liverpool.ac.uk

tion structure (CS^*) that maximises social welfare is known as the *coalition structure generation* (CSG) problem [11].

As a general overview, the coalition formation process for a *n*-person cooperative game can be described as 3 stages [11]:

1. **Coalition value calculations** - This involves computing the expected payoff of each possible coalition (usually each subset of *n* agents - an exponential amount).
2. **Coalition structure generation** - Agents are then organised into a coalition structure (preferably an optimal one that maximises social welfare).
3. **Determining the payoff distribution** - If the *n*-person cooperative game is a TU game then the payoff of each coalition in the coalition structure is divided between the agents of the system in a stable manner (see Section 2.2).

In the MAS literature there are various decentralized algorithms proposed for benevolent agents to solve the first two stages of the coalition formation process (and so the CSG problem) while constraining the worst case run time (see [6] for a review). The algorithms distribute the coalition value calculations among the agents instead of performing all the computation in one place so that there is no system bottleneck. The agents then communicate in some form to find a CS^* (there could be multiple).

There has been less focus in MAS on decentralized protocols for CSG with self-interested agents. The field of cooperative game theory (CGT) is used to find the possible arising *CS*s when a system is full of self-interested agents, but many CGT solution concepts make the assumption that the coalitions have already formed and offer no methods to form them from a MAS perspective [16].

The use of dialogue games has been shown to be a valid method to constrain the communications of self interested agents in self-interested environments [17]. Dialogue games are based on the theory of speech acts [15] and are rule governed interactions where each player moves by making utterances (in the form of locutions) according to a defined set of rules but in a flexible manner [10]. In self-interested environments a protocol is needed to set the rules as to what agents can or cannot do. In [17], dialogue games are identified as a satisfactory method for describing to agents what is allowed and forbidden in self-interested environments. Without a protocol, [17] argues that agent communication in self-interested environments can become chaotic.

The dialogue game proposed in this paper, named the *CSG dialogue game*, allows agents to build, in a decentralised manner, argumentation frameworks (AFs) [8] that have previously been shown to enable CSG (e.g. [1]). The advantage of using AFs for CSG is that the preferred extension of an AF always holds the best coalition structure for the agents given the dialogue history (as the dialogue example in Section 3.6 demonstrates). Argumentation schemes, which are patterns of reasoning that when instantiated provide presumptive justification for the particular conclusion of the scheme [2], are also used in the CSG dialogue game. The argumentation schemes allow the agents in the CSG dialogue game to assert arguments for different coalitions and payoff vectors.

The main significance of this paper is to propose a dialogue game that finds a coalition structure and a payoff vector in a static TU game (a TU game where the coalition's payoffs do not change). The result of this dialogue will be an optimal coalition structure and a stable outcome for the participating dialogue agents when the *core* is non-empty (given enough time and utterences). To find a stable solution of a static TU game that has an empty *core* is outside the scope of this paper and left for future work.

The rest of the paper is structured as follows: Section 2 describes the relevent background of the CGT and argumentation fields. Section 3 details the CSG dialogue game including the AF, the argumentation schemes, the restrictions placed on the CGT solution concepts, an algorithm to find when the *core* is empty and an example. Section 4 discusses related work and Section 5 discusses future work and concludes.

2. Background

2.1. Transferable Utility Games

An n-person cooperative game is: $\mathcal{G} = \langle N, v \rangle$ where N is the set of agents and v is the *characteristic function* ($v(2^N) \rightarrow \mathbb{R}$), which assigns every possible coalition a real numeric payoff [18]. An **outcome** of a TU game is a CS and payoff vector pair, denoted: $\langle CS, x \rangle$ where CS is a set of coalitions, denoted $\{C^1, ..., C^k\}$,. Static TU games are TU games where a coalition's payoff does not change. In TU games an agent can only be a member of one coalition, so a TU game coalition structure takes the form [6]; $CS = \{C^1, ..., C^k\}$ such that:

- $\bigcup_{j=1}^{k} C^j = N$ and
- $C^i \cap C^j = \emptyset$ for any $i, j \in \{1, ..., k\}$, $i \neq j$.

The first condition states that the union of all the coalitions in the TU game is equal to the full set of agents in the game. The second condition states that coalitions should not share any agent, otherwise this would be an overlapping coalition game [6]. The payoff vector is then fully denoted [6]: $x = (x_1, ...x_n) \in \mathbb{R}^n$ where $x_i \geq 0$ for all $i \in N$ and so x_i denotes the payoff that agent i receives. Throughout the paper $x(C)$ is written to denote the total payoff $\sum_{i \in C} x_i$ of a coalition $C \subseteq N$ under x and $x\langle C \rangle$ is written to denote the part of x that has individual payoffs only for agents of C.

2.2. Cooperative Game Theory Solution Concepts

The most popular solution concept to find stability in the CGT field is known as the **core**, which corresponds to the set of feasible payoff vectors where no subset of agents of the system have an incentive to deviate from the current coalition game outcome [6,9]. The core for a set of agents N can be defined using the following [6,9]:

Definition 1: *The* **core***:- A payoff vector* $x = (x_1, ..., x_n)$ *and a coalition Structure* $CS = (C^1, ..., C^k)$ *is in the core iff:*

- $\forall C \subseteq N$, $x(C) \geq \nu(C)$ *and*
- $x(C^j) = \nu(C^j)$ *for any* $C^j \in CS$

The first condition is the *non-blocking* condition; if this condition does not hold for some group of agents, C, then they have a reason to deviate and divide the payoff of $\nu(C)$ between themselves. The second condition is the *feasibility* condition; in the core no side payments between coalitions are allowed, so all of the payoff of a coalition has to be divided between its members.

Two problems with the *core* are that it can sometimes be empty and some core outcomes can be classified as unfair [6,9], see the following example take from [19].

Example 1: *Given a coalition game* $\langle N, v \rangle$ *where* $N = \{1, 2\}$, $v(\{1\}) = v(\{2\}) = 5$ *and* $v(\{1, 2\}) = 20$*, the proposed* **core** *outcome is* $\langle \{\{1, 2\}\}, x(15, 5) \rangle$.

This outcome is in the core since every agent is receiving at least as much as it would get in every possible different coalition. The outcome can be seen as unfair because even though both agents earn the same amount by themselves, when joined together as a coalition, agent 1 gets **all** the additional payoff.

To tackle these issues a few stability concepts have been introduced that are never empty and have different fairness properties, for example the **kernel** and **nucleolus**.

The *kernel* and *nucleolus* were introduced to define tighter stability restrictions than the *core*. These restrictions use the idea of *coalitional excess*, which is the additional amount each potential coalition C currently earns over or under $v(C)$. Stability is then measured by comparing the *excesses*. The kernel is the set of payoff vectors where the maximum excess between the players are balanced and the nucleolus is the payoff vector that have the lexicographically smallest excess.

The relation between the stability concepts mentioned so far are [14]:

1. *nucleolus* $\subseteq$ *kernel*
2. *nucleolus* $\subseteq$ *core* (if non-empty)

The protocol detailed in this paper will find a solution in the *core* (if the *core* is non-empty) but this solution will sometimes overlap with the *kernel* and *nucleolus* depending on the values of the coalitions in the TU game. Any solution the protocol finds in the *core* will follow certain fairness principles (detailed in section 3.4.1) so that the most unfair *core* payoffs are not found.

2.3. Argumentation Frameworks

In the CSG dialogue game, an agent can propose arguments for a coalition and a payoff vector. To evaluate all the instantiated arguments in the CSG dialogue game to determine their *acceptability* they will be organised into an Argumentation Framework (AF). AFs are a means to represent and reason with different, possibly conflicting data. AFs use graphs of nodes and arcs, where the nodes represent abstract arguments, having no internal structure, and the arcs represent attacks between the arguments [8]. An AF is defined as:

Definition 2: *An Argumentation Framework is a tuple AF = (Args, R) where Args is a set of arguments and R is a binary attack relation $R \subseteq Args \times Args$.*

In an AF, where $Args = \{a_1, ..., a_n\}$, an attack $a_1 R a_2$ is said to **defeat** a_2 if a_1 has not been defeated by another argument in the AF. A set of arguments S is **acceptable** iff $\forall a_x \in Args$ if a_x attacks an argument a_y where $a_y \in S$ there $\exists a_z \in S$ where a_z defeats a_x. S is a **preferred extension** of the AF if S is the maximal acceptable set of arguments.

3. The CSG Dialogue Game

3.1. The CSG Dialogue Game Overview

The CSG dialogue game can be used by self-interested agents willing to partake in a static TU game. A static coalition game is used as they are the traditional, oldest and most-studied style of n-person cooperative games [6,9,18]. The locutions available to the players of the game, inspired by [2], are *join*, *assert reject*, *close* and *leave* (outlined in Table 1) and the pre/post conditions of the full set of dialogue moves are defined later in the paper in Table 2. The CSG dialogue game is designed to be turn based, so that an algorithm to check if the *core* is empty can be developed. This paper assumes that a

Table 1. The informal meaning of the moves

Move	Format	Meaning
join	$\langle i, join, a_n \rangle$	Agent i wants to be included in the dialogue and is willing to form the best coalition it can with any other agents in the dialogue. To join, the agent must assert its individual payoff.
assert	$\langle i, assert, a_n \rangle$	Agent i believes the coalition and payoff vector proposed in a_n will improve on i's current payoff.
reject	$\langle i, reject, a_n \rangle$	Agent i believes another agent has misrepresented the value of the coalition in a_n and believes this coalition currently cannot offer a fair payoff. Agent i can only reject its current best coalition (or this is wasted communication).
close	$\langle i, close \rangle$	An agent closes the dialogue if it does not believe it has any more moves available. If every agent's last move was a close move, without another move in-between, then the dialogue is over.
leave	$\langle i, leave \rangle$	As the agents are self-interested, agents may want to leave at any point. An agent utters a leave move if it wants to quit. An agent may only utter a leave move if it has joined the dialogue. Leaving is *final* and cannot be undone, to stop additional computational overheads. All arguments that include i are removed from the dialogue. Also all other arguments that use an argument that included agent i as evidence for an improved payoff are also removed.

suitable enforcement mechanism, in the form of a reject argumentation scheme is used to ensure the agent's assertions are truthful.

In static TU games, a coalition's value does not change during the coalition formation process. But an agent could attempt to assert an erroneous coalition value by deceitfully highering it. A rational agent would not assert an erroneous lower value as this could only negatively affect the agents final payoff. Deceitfully highering a coalition's payoff may help one agent but could hinder others as the CSG dialogue game may recommend that agents join the coalition with the misrepresented value. Truthful agents are allowed to assert a coalition's lower value. The lowest value asserted in the CSG dialogue game for each coalition is taken as its true value (see the *maxValue* later). Agents may use the *reject* locution if they disagree with a coalition's payoff and determine that a fair payoff vector cannot be found with the coalition's correct payoff.

Some of the recent MAS literature provides various algorithms to split up the coalition value calculations (e.g. the DCVC algorithm [11]), yet the only condition used to split up the coalition value calculations in the CSG dialogue game is: $i \in C$. This means that any agent of a coalition can assert an argument for that coalition. This is not the optimal method to compute coalitions structures as agents have to compute the value for 2^{n-1} coalitions instead of the optimal method which is where each agent computes the value of a distinct subset of all the coalitions. But the condition $i \in C$ is used as self-interested agents cannot be trusted to assert coalitions when only others would benefit[2]. So even though the outcome of this CSG dialogue consists of a CS and payoff vector pair, individual agents will only be able to argue over the subsets of both that concern it.

CSG dialogue games will only be feasible for small sets of agents or coalition games with restrictions such as: coalition games represented via a combinatorial structure (e.g. induced sub-graphs [6]); constrainted coalition games [12]; or CSG dialogue games with a predefined end time.

[2]Unless a sufficient enforcement mechanism is used, but this is outside the scope of this paper.

Before a dialogue commences the mechanism designer, as the overall controller of the dialogue game, has to set the following variables, which ensure finite execution time.

- The **timelimit** for agents to join the dialogue, denoted t, where $t \in \mathbb{N}$. This is the duration any agent has to announce if it wants to be in the dialogue.
- The **increment values**, set using the *inc* function of Defn. 5. They are the amounts proposed payoff vectors have to be raised by for the agents to consider the proposal. The *inc* function returns a real number from the set $\mathbb{R} \geq 0$.
- The dialogue **end time**, denoted et, where $et \in \mathbb{N}$ can be used if there are any time restrictions and the best CS found by that point will be returned. This is the only variable that does not have to be set for every game.

Once the mechanism designer has set these variables a new dialogue game can commence. The dialogue, based on [2], is denoted D_r^s where r is the timepoint of the first move of the dialogue and s the timepoint of the last, where $r, s \in \mathbb{N}$. The moves of the dialogue can be referred to individually as $m_r, ..., m_s$. Individual moves are abstractly defined as $m_n = \langle i^n, \alpha^n, a_m^n \rangle$, where i^n is the agent who asserted the move m_n, α^n is the name of the move m_n and a_m^n is the argumentation scheme of the move m_n. The following functions act over an individual move:

- $Utterer(m_n) = i^n$. This function will return the agent who uttered m_n.
- $Type(m_n) = \alpha^n$. This function will return m_n's name.
- $Argue(m_n) = a_m^n$. This function will return the argument of the move m_n. The empty set will be returned if the move does not have an argument.

Every time an agent makes an utterence of move m_n, the argumentation scheme returned by $Argue(m_n)$ is stored in a publicly readable commitment store ($CoSt$)[2]: An **individual commitment store** for a CSG dialogue (D_r^s), for every agent $i \in N$ a commitment store of agent i at time-point s is denoted $CoSt_i^s$. A **combined commitment store** for a CSG dialogue (D_r^s) with participants N and time-point s is denoted $CoSt_N^s$ where $CoSt_N^s = \bigcup_{\forall i \in N} CoSt_i^s$. If $s = 0$ then $CoSt_N^s = \emptyset$.

3.2. The Coalition Argumentation Schemes

Abstract arguments themselves are not always useful for representing instantiated arguments, that is, arguments with some internal structure or content. To reason on the best coalitions to form from a game theortic perspective, argumentation schemes are used, instantiated and then placed in an argumentation framework. Instantiations of argumentation schemes attack other instantiations of argumentation schemes under predefined conditions, which attack either the premises, inference rules or conclusions of the scheme. In the CSG dialogue game, all attack rules focus on the conclusions of the argumentation schemes (that is, whether to form the coalition or not). The two new argumentation schemes used in this paper are: $\mathscr{C}$**-Arguments** and $\mathscr{R}$**-Arguments**, which make use of the concepts defined earlier, in Section 2.

A $\mathscr{C}$-argument should be instantiated when an agent wants a coalition C to form. So the conclusion of a $\mathscr{C}$-argument is that C should form. It is informally described as:

$\mathscr{C}$**-Argument**: *Agent i asserts that coalition C* **should form**, *since given the current payoff of the agents of C, denoted $x^{s-1}\langle C \rangle$, and the coalition payoff of $v^s(C)$, then the payoff vector of $x^s\langle C \rangle$ should be implemented as $x^s\langle C \rangle$ offers an equal or better, fairer payoff for all of C.*

A $\mathscr{R}$-argument should be instantiated when an agent i refuses to join a coalition C. This can happen when i finds that the value given for a coalition C does not match the value expected as reported by the characteristic function and the real value would not allow a fairer payoff, as defined in section 3.4.1. So the conclusion of a $\mathscr{R}$-argument is that C should **not** form. This argumentation scheme is used to stop manipulations in the CSG dialogue game and it is informally described as:

$\mathscr{R}$-**Argument**: *Agent i asserts that coalition C* **should not** *form, as the coalition's value, previously asserted as $v^{s-1}(C)$, is wrong and with the correct value $v^{s}(C)$ a fair payoff cannot be found.*

The attack rule for the instantiated argumentation schemes is: A newly asserted argument a_s about a coalition C, attacks any other argument a_p previously asserted in the dialogue, that shares a member of C. This attack is used so that agents can only be part of one coalition.

The following functions operate over any a_n; $Coal(a_n)$ returns the coalition C proposed in a_n; $Vect(a_n)$ returns the section of the new payoff vector $x^n\langle C\rangle$ proposed in a_n; $Val(a_n)$ returns the value $v^n(C)$ of the coalition proposed in a_n.

The above argumentation schemes are for the traditional static TU game. In the future more information can be added to the schemes and the attack rules can be changed to make the CSG dialogue game applicable to different types of coalition games.

3.3. Coalition Argumentation Framework

Now to find the *outcome* of the CSG dialogue game ($\langle CS, x\rangle$), firstly the preferred extension (PE) of the AF is found. Then the agents systematically consider the internal structure of the abstract arguments in the PE by looking at the instantiated versions of these abstract arguments. To find the *outcome*, firstly the preferred extension (PE) of the AF is found. Then the *CS* is the collection of all the coalitions proposed by the instantiated versions of the abstract arguments present in the PE that are $\mathscr{C}$-arguments and x is the conjunction of all the payoff vector subsets that are proposed by the same $\mathscr{C}$-arguments. Elements of the $\mathscr{R}$-arguments in the PE are not in the *outcome* of the coalition game since a $\mathscr{R}$-argument conclusion is for the proposed coalition to *not* form. The following functions are used to find the *outcome* of a coalition game, while only the *PayVect* and *CoalStruct* functions directly find the coalition game outcome, all other functions are used to help the agents work towards finding a *core* stable outcome if one exists:

- $PE(CoSt)$ = Φ. Where Φ is the preferred extension of *CoSt* given the attack relations taken from the instantiated argumentation schemes of the given *CoSt* followed by the removal of all $\mathscr{R}$-arguments.
- $CoalStruct(CoSt) = \lambda$. $\forall a_n \in PE(CoSt)$, $Coal(a_n) \in \lambda$. This function returns the set of all the coalitions in the instantiated argumentation schemes of the preferred extension (minus the $\mathscr{R}$-arguments) of the given CoSt.
- $PayVect(CoSt) = x$, where $x = (x_p, ..., x_q)$. $\forall a_n \in PE(CoSt), \forall x'_j \in Vect(a_n)$ then $x_j = x'_j$. This function returns as one tuple the payoffs of all the individual agents in the instantiated schemes of the preferred extension (minus the $\mathscr{R}$-arguments) of the given CoSt.
- $CoalStructVal(CoSt) = \sum_{a_n \in PE(CoSt)} Val(a_n)$. This function finds the value of the coalition structure of the given CoSt.
- $FairPayDist(CoSt, a_n)$. This function restricts the most unfair payoffs from being asserted. If a payoff that is deemed unfair is found in the given a_n then $\bot$ is returned, else $\top$ is returned. Further details in Section 3.4.1.

- $maxVal(CoSt, a_n) = coalVal(a_p)$ if $\exists a_p \in CoSt$ where $Coal(a_p) = Coal(a_n)$ and $\neg\exists a_q \in CoSt$ where $Coal(a_q) = Coal(a_n)$ and $CoalVal(a_q) < CoalVal(a_p)$ **else** $maxVal(CoSt, a_n) = coalVal(a_n)$. This function returns the maximum value that the coalition returned by $Coal(a_n)$ can be in the dialogue, which is the smallest value asserted in the dialogue so far. This is to stop agents manipulating the dialogue by inflating a coalition's worth.
- $BestCoal(CoSt, i) = C$. $\forall C' \in CoalStruct(CoSt)$, if $i \in C'$ then $C = C'$. This function returns the best coalition for the given agent i.
- $DiaAgs(D_r^s) = N$. $\forall m_p \in D_r^s$ where $Type(m_p) == join$ then $Utterer(m_p) \in N$ **iff** $\neg\exists m_q \in D_r^s$ where $Type(m_q) == leave$ and $Utterer(m_q) == Utterer(m_p)$.
This function returns the agents currently in the dialogue. To get a complete view of the agents in the system this function should only be called after the time limit for the agents to join the dialogue has passed.
- $DiaOpen(D_r^s) = \bot$ **if either** $(et < currentTime)$ **or** $\forall i \in DiaAgs(D_r^s)$, $\exists m_p \in D_r^s$ where $Type(m_p) = close$ and $p \geq s - |DiaAgs(D_r^s)|$ **else** $DialogueOpen(D_r^s) = \top$.
This function returns whether the dialogue is open ($\top$) or closed ($\bot$). For a dialogue to be closed either the time has run out or every agent has asserted a close move without a different move inbetween.

In the CSG dialogue game only one PE can exist, so no attack cycles can be made. This is according to the argumentation scheme attack rules, which state that no argument can ever attack another argument asserted after it. This rule is in place as newer arguments should have fairer payoff vectors and so will attack older arguments that have less fair payoff vectors (according to the fairness rules defined in section 3.4.1). Therefore computing the preferred extension of this CSG dialogue game takes time linear in the number of arguments [8].

3.4. Formalising the Argumentation schemes

The formal definition of the two argumentation schemes (informally described in section 3.2) at timepoint s in dialogue D_r^s are:

Definition 3: *A* $\mathscr{C}$**-argument** $\mathscr{C} = \langle i, C, x^{s-1}\langle C\rangle, v^s(C), x^s\langle C\rangle\rangle$ *s.t:* $i \in C$; $C \subseteq N$; $x^{s-1}\langle C\rangle \in \text{PayVect}(CoSt_N^{s-1})$; $\text{MaxVal}(CoSt_N^{s-1}, \mathscr{C}) \geq v^n(C)$; *and* $\text{FairPayDist}(CoSt_N^{s-1}, \mathscr{C}, i) = \top$.

A $\mathscr{C}$-argument can only be asserted if the *FairPayDist* function returns $\top$ else the argument is invalid because an unfair payoff has been suggested.

Definition 4: *A* $\mathscr{R}$**-argument** $\mathscr{R} = \langle i, C, x^m\langle C\rangle, v^n(C), x^n\langle C\rangle\rangle$ *s.t:* $i \in C$; $C \subseteq N$; $x^m\langle C\rangle \in \text{PayVect}(CoSt_N^{s-1})$; $\text{MaxVal}(CoSt_N^{s-1}, \mathscr{R}) > v^n(C)$; *and* $\text{FairPayDist}(CoSt_N^{s-1}, \mathscr{R}, i) = \bot$.

A $\mathscr{R}$-argument can only be asserted if the *FairPayDist* function returns $\bot$ else the $\mathscr{R}$-argument is invalid since there is no reason to object to the $\mathscr{R}$-argument's coalition, as the payoff of the coalition of $\mathscr{R}$ can be distributed in a fair manner. If an agent disagrees with a previously asserted coalition's value but still thinks that a fair payoff can be found with the new coalition's value then the agent should use a $\mathscr{C}$-argument, as the *maxVal* function was created to deal with these issues.

Now the formalised **attack rule** (described in Section 3.2) for an argument a_s is: a_s attacks every argument a_p in $CoSt_N^{s-1}$ **if** $Coal(a_s) \cap Coal(a_p) \neq \emptyset$.

3.4.1. Ensuring Fair Payoff Distributions

The *kernel* and *nucleolus* offer payoff vectors that are more stable and so can be classified as fairer than the standard definition of the *core*, but this CSG dialogue game does not focus on using these stability concepts for a few reasons relating to the communication and computation costs, which are important issues in coalition structure generation protocols [6]. Firstly an algorithm for computing one side payment for a payoff vector heading towards a *kernel*-stable payoff is in the *best case*: $O(n2^n)$ [16]. For the CSG dialogue game, even though it is true that the *worst case* complexity will still be exponential (as there can be an exponential amount of coalitions to check), the *best case* is significantly less. The CSG dialogue game finds *core* payoffs (with fairness constrictions) and so unlike [16], agents in the CSG dialogue game do not have to look at all the potential coalitions before computing one side payment for a new payoff vector. Instead in the *best case* they only have to look at one coalition and see if a payoff vector can be found that is fairer than the current payoff vector by satisfying Defn. 6 below.

The protocol for coalition structure generation provided by [16] also requires in the *best case* that all the agents: communicate all of the coalition's value to all of the other agents; communicate their personal optimal *CS*s to other agents; take part in a voting method to choose the final *CS**; and all agents involved in the final *CS** decision should transmit the details of the calculations that led to the decision of the *CS** to all other agents. All of these combined requirements leads to high communication costs, even in the *best case*. The CSG dialogue game's communication requirements are significantly smaller in the *best case*. For example, the *CS* requiring the least communication in a CSG dialogue game is the *CS* of singleton coalitions (e.g. for a 3-person game, the coalition structure of singletons would be: $CS = \{\{1\},\{2\},\{3\}\}$). For an n person game then $2n$ dialogue moves are required to form the *CS* of singletons (n join moves and n close moves) but [16] would require significantly higher amounts of communication (e.g. for all agents to communicate the entire exponential amount of coalition values to each other).

As the *nucleolus* is a subset of the *kernel*, the same problems in finding the *kernel* that are highlighted above are present also for finding the *nucleolus*. In fact, to the authors knowledge, there does not exist a decentralised method to find the *nucleolus* <u>and</u> an optimal coalition structure at the same time.

During the dialogue, an agent i will only be able to deviate from the current *CS* by asserting a new coalition C if $i \in C$ and all agents of C (including i) receive an incentive. The actual additional *incentive amount* agents must receive above their current payoff to defect to another coalition is set using Defn. 5, which ensures that the *CS** is found when no more moves are possible, as increasing the social welfare does not incur a cost.

Definition 5: $inc(CoSt, a_n, i) = 0$ *iff*

$$\begin{aligned}&\text{CoalStructVal}(CoSt \cup \{a_n\}) > \text{CoalStructVal}(CoSt)\ \textit{or}\\ &\quad \text{CoalStructVal}(CoSt \cup \{a_n\}) == \text{CoalStructVal}(CoSt)\ \textit{and}\\ &\quad |\text{Coal}(a_n)| > |\text{BestCoal}(CoSt, i)|\end{aligned}$$

else $inc(CoSt, a_n, i) > 0$

In the CSG dialogue game if an agent wants to assert a new argument a_n, then a_n needs to pass the following test (defined in Defn. 6), which checks if a_n results in the payoff vector converging towards *core* stability. The first bullet point of Defn. 6 ensures individual rationality and that the new payoff for all agents of C is greater than or equal to their previous payoff. The second bullet point ensures that no utility is lost out of the game. The third bullet point negates the unfair payoff criticism of the *core* by ensuring all agents get either an equal split of the coalition's payoff or each agent j that can get a greater payoff elsewhere is given at least that payoff in this coalition. The additional pay to agent j (above the equal split of the coalition's payoff) is taken equally from the remaining agents (unless this would motivate others to deviate).

Definition 6: *The* **fairPayDist**$(CoSt, a_n, i)$ *function, where* C = Coal(a_n), CS = Coal-Struct $(CoSt \cup \{a_n\})$, x = PayVect$(CoSt \cup \{a_n\})$ *and* x' = PayVect$(CoSt)$, *returns* $\bot$ *unless the following are satisfied, if so the function returns* $\top$*:*

- $\forall j \in C, x_j \geq \nu(\{j\})$ *and* $x_j \geq x'_j + \mathrm{inc}(CoSt, a_n, i)$
- $\sum_{j \in C} x_j = \mathrm{Val}(a_n)$
- double *split* $\leftarrow$ *split'* $\leftarrow$ *0*

  $C'' \leftarrow C' \leftarrow C$

  while $C'' \neq \emptyset$ **do**

      boolean *equal* $\leftarrow$ *true*

      $split \leftarrow \dfrac{split'}{|C''|}$

      for all $j \in C''$ *where* $x'_j + \mathrm{inc}(CoSt, a_n, i) \geq \dfrac{v(C)}{|C|} - split$ **do**

          $x_j \leftarrow x'_j + \mathrm{inc}(CoSt, a_n, i)$

          $split' \leftarrow split' - \dfrac{v(C)}{|C|} + x_j$

          $C' \leftarrow C' \setminus \{j\}$

          equal $\leftarrow$ *false*

      if *equal* == *true* **then**

          for all $k \in C''$ **do**

              $x_k \leftarrow \dfrac{v(C)}{|C|} - split$

              $C' \leftarrow C' \setminus \{k\}$

      $C'' \leftarrow C'$

3.5. Checking if the core is empty

This dialogue game can only find solutions to a coalition game if the *core* is **not** empty and so a valid question is: how do the agents know if the *core* is empty? To find the answer to the question the agents should communicate in stages: Firstly they should join the dialogue; secondly they should compute the coalition's values; and thirdly each agent i should assert either a $\mathscr{C}$-argument where the *inc* function returns 0 (if one can be found, as this coalition will improve social welfare) or assert a $\mathscr{C}$-argument where the *inc* function returns > 0 (as this coalition will improve the payoff for agent i but not the social welfare), else the agent should assert a close move.

As communication happens in these stages and agents assert social welfare improving arguments first, the core can be checked to be non-empty under a certain condition: for all arguments asserted in the dialogue for the same coalition C and with the same value for $v(C)$, if there does not exist an agent in these arguments that has a strictly increasing or decreasing payoff over time, then the *core* is empty. The reasons for this are the following:

1. If a coalition C is in the outcome of a CSG dialogue and a payoff has been found in the *core* then for all arguments in the dialogue for C that share the coalitions

final $v(C)$ value then these arguments will include at least one agent who has a strictly increasing/decreasing payoff.

2. If the *core* is non-empty then the dialogue is finite.
3. If the *core* is empty then the dialogue is infinite, therefore certain conditions will be met.

Discussion for point 1: If agent 1 was the first agent to assert coalition C with the payoff $x\langle C\rangle$ where $C = \{1, i, ..., j\}$ then agents $i, ..., j$ will have an opportunity to object to this payoff by asserting an $\mathscr{C}$-argument, which will be called an *core*-objection to C. An agent i will object to coalition C if agent i can get a higher payoff from another coalition. So a *core*-objection means that the first condition of the *core* $(\forall C \subseteq N,\, x(C) \geq v(C))$ did not hold for the initial C and $x\langle C\rangle$ proposed by agent 1.

Every objection counts as a new proposal, so if agent i objected to agents 1's proposal by asserting $C' = \{i, p, ..., q\}$ and $x'\langle C'\rangle$ then all agents $p, ..., q$ now have a turn to object to this outcome. If coalition C can incorporate this objection then it will be eventually reasserted by an agent with the payoff vector $x''\langle C\rangle$ and this time; $\forall j \in C' \cap C$ then $x''_j\langle C\rangle \geq x'_j\langle C'\rangle$.

As each agent j always objects to a coalition by asserting a coalition that offers j the best payoff, given the conditions previously mentioned, then the agents that object the most, continue to raise their payoffs given the dialogue history and the agents that object the least continue to lose payoff given their dialogue history. If there exists a coalition where all the agents object to the payoff then this coalition cannot form as all the agents can get more payoff elsewhere. So there must always be at least one agent in every coalition in a *core* stable coalition game outcome that never objects to the coalitions payoff and so these agents will always lose payoff after every objection. As these agents always lose some payoff after every objection then there exists at least one agent i in every coalition C in a *core* stable game outcome with a strictly decreasing payoff in the arguments for C where $v(C)$ is equal to the final value of C (so there has been no deceitful values in these arguments).

As regards point number 2, as the objections increase the strongest agents payoffs, eventually a time will be reached when $(\forall C \subseteq N,\, x(C) \geq v(C))$ is satisfied as the value of all coalitions are finite and static. When this time is reached, then no more *core*-objections can be formed as a *core* payoff has already been found. The second condition of the *core* $(x(C) = v(C))$ is also satisfied for every dialogue assertion according to bullet point 2 of Defn. 6.

Regarding point number 3, when the *core* is empty then there will always exist an agent that can object to the current payoff as $(\forall C \subseteq N,\, x(C) \geq v(C))$ is never satisfied and so CSG dialogue games with an empty *core* become infinite unless stopped. As the dialogues become infinite, yet the payoffs of the coalitions are finite and the permutations of the payoff vectors are also finite if the *inc* function is set correctly, then an infinite dialogue can be identified when either a previous payoff vector is repeated or a series of payoff vectors that were converging in one direction, stop, then start converging to another direction (this happens when the potential *core* payoff vector set is non-existent).

So to identify when the core is non-empty, the agents should run in algorithm 1, which is a summary of the issues just discussed.

Algorithm 1. Algorithm to play the CSG dialogue game to find out if the core is empty

1. In the first stage, agents join the dialogue and assert their individual payoff.
2. In the second stage agents compute all their assigned coalition values.
3. In the third stage, agents may assert a previously unasserted coalition **or** assert a coalition that has previously been asserted **but** change its payoff vector. If an agent i is asserting a coalition, agent i should first check that no other coalition in the TU game exists that can offer a higher, fairer individual payoff for i given the history of dialogue utterances (if two coalitions C^j and C^k can be asserted and both can offer agent i the same payoff, than C^k should be choosen if $|C^k| > |C^j|$, if both coalition's have the same amount of agents then the choice can be random). A data structure maybe used here to order the coalitions, so that all coalitions do not have to be searched. Reject moves can also be performed.
4. Once a payoff vector of a coalition C has been changed via asserting the argument a_s, the agents should check:
   - $\forall a_n \in CoSt$ then $a_n \in \gamma$ if $C = Coal(a_s)$, $Val(a_n) = Val(a_s)$ and $\gamma =$ **acceptable**. This gathers all arguments for C that do not have a known deceitful value in.
   - **If** $\forall i \in C$, where $\forall a_p, a_q \in \gamma$ either $x_i^p > x_i^q$ or $x_i^p < x_i^q$ does not hold when $x^p \in Vect(a_p)$, $x^q \in Vect(a_q)$ and $p < q$ then the core **is empty**.

   If the core is empty then the game can be abandoned **or** a compromise can be agreed on, such as: the current CS can be taken.
5. In the fifth stage each agent should check if any additional utterences have been made since its last move. *If* yes than the agent should repeat the algorithm from stage 3 *else* the agent should utter a close move.

3.6. A Dialogue Example

Now that the formalism has been established, the dialogue pre and post conditions can be introduced in Table 2.

A full example can now be developed in Table 3 and Figure 1. As can be seen in Table 3, communication happens in stages: firstly the agents join the dialogue and assert their individual payoff; secondly the agents assert new coalitions or modify previously asserted coalitions' payoff vectors to gain fairer payoffs; finally when each agent can find no other variables that satisfy the argumentation scheme restrictions they utter a close move.

The AF generated from the example in Table 3 can be seen in Figure 1 where the argument number corresponds to the move number. Finding the outcome ($\langle CS, x\rangle$) of the CSG dialogue after move n requires computing the preferred extension of the AF of Figure 1 for the arguments: A1 to An. The preferred extension, after the CSG dialogue game of Table 2 has completed, is: $\{A1, A8\}$. When looking at the instantiated argumentation schemes of A1 and A8 we can see that the $CS = \{\{1\},\{2,3\}\}$ and the payoff vector is x(4,11,13). As all agents only uttered a close move when no more moves could be asserted, we find that this CS is the optimal CS of the game and x is <u>inside</u> the *core*.

4. Related Work

So far in the argumentation and dialogue literature, a few attempts have been made to detail coalition formation techniques such as [1,8,3,4,5]. Yet the majority of the argumentation based coalition formation papers refer to the idea of stability but do not further

Table 2. The formal definitions of the moves available to the agents in the Dialogue D_r^s. All moves are described in Table 1 of section 3.2

Move	Format	Pre-conditions	Post-conditions
join	$\langle i, join, a_n\rangle$	$(DiaOpen(D_r^s) = \top) \wedge (a_n = \mathscr{C}) \wedge (currentTime < t) \wedge (i \notin DiaAgs(D_r^s)) \wedge (Coal(a_n) == \{i\}) \wedge (\neg\exists m_p \in D_r^s$ where $Type(m_p) ==$ leave and $Utterer(m_p) == i)$	The instantiated argumentation scheme, a_n, of the join move is added to the commitment store: $CoSt_i^s = CoSt_i^{s-1} \cup a_n$.
assert	$\langle i, assert, a_n\rangle$	$(DiaOpen(D_r^s) = \top) \wedge (a_n = \mathscr{C}) \wedge Coal(a_n) \notin CoalStruct(CoSt_n^{s-1})$	The instantiated argumentation scheme, a_n, of the assert move is added to the commitment store: $CoSt_i^s = CoSt_i^{s-1} \cup a_n$.
reject	$\langle i, reject, a_n\rangle$	$(DiaOpen(D_r^s) = \top) \wedge (a_n = \mathscr{R}) \wedge (i \in Coal(a_n)) \wedge (bestCoal(i, CoSt) == Coal(a_n)) \wedge (CoalVal(a_n) \leq MaxVal(CoSts, a_n))$	The instantiated argumentation scheme, a_n, of the reject move is added to the commitment store, then all the arguments in the commitment store that have the same coalition as a_n are removed as they have an invalid coalition value: $CoSt_i^s = CoSt_i^{s-1} \cup a_n$. **If** $a_n == \mathscr{R}$ then $\forall a_m \in CoSt_n^{s-1}$, $a_m \in \gamma$ iff $a_n \mathrm{R} a_m$ and $Coal(a_n) == Coal(a_m)$. $CoSt_n^s = CoSt_n^{s-1} \setminus \gamma$.
close	$\langle i, close\rangle$	$(DiaOpen(D_r^s) = \top) \wedge (i \in DiaAgs(D_r^s))$	If DialogueOpen$(D_r^s) == \bot$ then the dialogue has finished and the outcome of the CSG dialogue game is $\langle CoalStruct(CoSt_N^s), PayVect\ (CoSt_N^s)\rangle$
leave	$\langle i, leave\rangle$	$(DiaOpen(D_r^s) = \top) \wedge (i \in DiaAgs(D_r^s))$	As agents are self interested, they can leave whenever they want. To allow for this, once a leave move is uttered by agent i, then the committment store must update and remove all coalitions from which agent i was a member. Also if an utterance of a coalition with agent i has changed the payoff of another coalition, then the other coalition needs to be removed also, for fairness issues: $\forall a_p \in CoSt_n^{s-1}$ where $i \in Coal(a_p)$ then $a_p \in \gamma$ and $\forall a_q \in CoSt_n^{s-1}$ where $a_q R a_p$, $Coal(a_q) \cap Coal(a_p) \neq \emptyset$ and $\exists x_j, x_k \in Vect(a_q)$ where $x_j \neq x_k$ then $a_q \in \gamma$. $CoSt_n^s = CoSt_n^{s-1} \setminus \gamma$.

clarify exactly what cooperative game theory solution concepts they are using and so are ambiguous on exactly what type of coalition game their frameworks can be used for. An interesting exception to this is [8], where Dung used n-person games to demonstrate the correctness of his argumentation frameworks. He showed that argumentation frameworks can be used to represent *von Neumann-Morgenstein* and *core* stable games. But Dung did not investigate how these solutions can be collaboratively built by multi-agent systems in a decentralised manner. This issue is investigated here.

Amgoud [1] again showed that argumentation frameworks could be used to solve the coalition structure generation problem and detailed different argumentation semantics so that a solution can always be found. But these semantics and formalism do not show a complete link to cooperative game theory, as [1] leaves out the details of the payoff vectors. Also [1] does not offered a n-person dialogue game to find an optimal coalition structure (*CS**), but does detail a two person dialogue game to find if one coalition is in the *CS**. Again all of these issues are dealt with in this paper.

Table 3. This table details a CSG dialogue for a $\mathscr{G} = \langle N, v\rangle$ where $N = \{1,2,3\}$, $v(\{1\}) = v(\{2\}) = 4$, $v(\{3\}) = 5$, $v(\{1,2\}) = 8$, $v(\{1,3\}) = 18$, $v(\{2,3\}) = 24$, $v(\{1,2,3\}) = 12$. A move at timepoint s is of the following form: the agent's identifier; the move type; and (if not a close move) an argumentation scheme instantiation i.e a $\mathscr{C}$-argument of the form $\langle i, C, x^{s-1}\langle C\rangle, v^s(C), x^s\langle C\rangle\rangle$. The attack rule is $a_p R a_q$ iff $Coal(a_p) \cap Coal(a_q) \neq \emptyset$ and argument a_p is asserted after a_q. For the dialogue to finish, n close moves need to be asserted consecutively.

Move No.	Move	CS	x
1	$\langle 1, join, \langle 1, \{1\}, (-), 4, (4)\rangle\rangle$	$\{\{1\}\}$	$x(4)$
2	$\langle 2, join, \langle 2, \{2\}, (-), 4, (-,4)\rangle\rangle$	$\{\{1\},\{2\}\}$	$x(4,4)$
3	$\langle 3, join, \langle 3, \{3\}, (-,-), 5, (-,-,5)\rangle\rangle$	$\{\{1\},\{2\},\{3\}\}$	$x(4,4,5)$
4	$\langle 1, assert, \langle 1, \{1,3\}, (4,-,5), 18, (9,-,9)\rangle\rangle$	$\{\{2\},\{1,3\}\}$	$x(9,4,9)$
5	$\langle 2, assert, \langle 2, \{2,3\}, (-,4,9), 24, (-,12,12)\rangle\rangle$	$\{\{1\},\{2,3\}\}$	$x(4,12,12)$
6	$\langle 3, assert, \langle 3, \{1,3\}, (4,-,12), 18, (5,-,13)\rangle\rangle$	$\{\{2\},\{1,3\}\}$	$x(5,4,13)$
7	$\langle 1, close\rangle$	$\{\{2\},\{1,3\}\}$	$x(5,4,13)$
8	$\langle 2, assert, \langle 2, \{2,3\}, (-,4,13), 24, (-,11,13)\rangle\rangle$	$\{\{1\},\{2,3\}\}$	$x(4,11,13)$
9	$\langle 3, close\rangle$	$\{\{1\},\{2,3\}\}$	$x(4,11,13)$
10	$\langle 1, close\rangle$	$\{\{1\},\{2,3\}\}$	$x(4,11,13)$
11	$\langle 2, close\rangle$	$\{\{1\},\{2,3\}\}$	$x(4,11,13)$

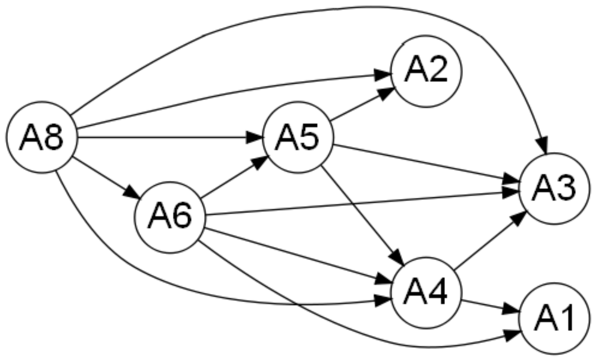

Figure 1. The AF generated via the CSG dialogue of Table 3

The idea of using dialogue games for coalition formation can be traced back to [7]. In [7] a dialogue game for agents to form a coalition is detailed but [7] does not show how a *CS* can be generated, which can be found with the CSG dialogue game.

Lastly, distributed coalition structure generation methods in multi-agent systems for benevolent agents have received a lot of recent attention [6]. Distributed coalition structure generation methods for self-interested agents have received significantly less attention. The main exception is [16] which looks for *kernal* stable coalitions; some of the differences of this paper are detailed in Section 3.4. More differences include; deceitful values should be detected earlier in the CSG dialogue game; and the whole protocol will not have to restart if a deceitful/erroneous coalition value is found.

5. Conclusion and Future Work

In this paper, an argumentation-based dialogue to find coalition structures has been proposed. The novel contribution of this paper is therefore: an n-person dialogue game that can find game theoretic stable coalition structures with explicit individual payoff for each agent if the *core* is non-empty; and a coalition structure that maximises social welfare can be found given certain conditions detailed in the paper. Additionally the paper discusses the restrictions placed on the *core* so that the most unfair *core* payoffs can never be suggested and an algorithm is detailed that shows how the agents can find out if the *core* is empty. Lastly the CSG dialogue offers the MAS community another way to solve the CSG problem for self-interested agents in TU games with computation and communication costs that are less than [16] in the best case. This paper outlines the framework that will be built on by future research to be applicable to different types of coalition games.

In the future I will aim to implement the ε-core solution concept so that a game theoretic stable solution can be found in every static TU game.

Lastly creating an enforcement mechanism for the agents so that the search space of coalitions values can be more efficiently divided between them would lead to minimising the computation time of finding the CS^* and a stable payoff. This problem of minimising the computation time has stimulated lots of research with benevolent agents (e.g. [11]) yet so far little research has been conducted on efficiently computing self-interested agents' optimal coalition structures.

References

[1] L. Amgoud. An argumentation-based model for reasoning about coalition structures. *The second International Workshop on Argumentation in Multi-Agent Systems, ArgMAS'2005*, pages 217–228, 2005.

[2] E. Black and K. Atkinson. Dialogues that account for different perspectives in collaborative argumentation. In *Proc. of the 8th Conf. on Autonomous Agents and Multi-Agent Systems*, pages 867–874, 2009.

[3] G. Boella, D. Gabbay, A. Perotti, and S. Villata. Coalition formation via negotiation in multiagent systems with voluntary attacks. In *Proceeding of the 22nd Benelux Conference on Artificial Intelligence (BNAIC)*, 2010.

[4] N. Bulling and J. Dix. Modelling and verifying coalitions using argumentation and atl. *Inteligencia Artificial*, pages 45–73, 2010.

[5] C. Cayrol and M. Lagasquie-Schiex. Coalitions of arguments: A tool for handling bipolar argumentation frameworks. *Intelligent Systems*, pages 83–109, 2010.

[6] G. Chalkiadakis, E. Elkind, and M. Wooldridge. *Computational Aspects of Cooperative Game Theory*. Morgan & Claypool Publishers, 2011.

[7] F. Dignum, B. Dunin-Keplicz, and R. Verbrugge. Agent theory for team formation by dialogue. In *Proc. of the 7th Int. Workshop on Agent Theories, Architectures and Languages*, pages 141 –156, 2000.

[8] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. *Artificial Intelligence*, 77:321–357, 1995.

[9] D. Gillies. *Some theorems on n-person games*. PhD thesis, Princeton University, 1953.

[10] P. McBurney and S. Parsons. Dialogue games for agent argumentation. *Argumentation in Artificial Intelligence*, pages 261–280, 2009.

[11] T. Rahwan. *Algorithms for Coalition Formation in Multi-Agent Systems*. PhD thesis, University of Southampton, 2007.

[12] T. Rahwan, T. Michalak, E. Elkind, P. Faliszewski, J. Sroka, M. Wooldridge, and N. Jennings. Constrained coalition formation. In *The Twenty Fifth Conference on Artificial Intelligence (AAAI)*, 2011.

[13] L. Riley, K. Atkinson, and T. Payne. Coalition structure generation for self interested agents in a dialogue game. Technical Report ULCS-12-004, University of Liverpool, 2012.

[14] T. Rombouts. *Solutions Of Argumentation In Cooperative Game Theory*. PhD thesis, University of Utrecht, 2004.

[15] J. Searle and D. Vanderveken. *Foundations of Illocutionary Logic*. Cambridge University Press, 1985.

[16] O. Shehory and S. Kraus. Feasible formation of stable coalitions among autonomous agents in non-super-additive environments. *Computational Intelligence*, 15(3):218–251, 1999.

[17] C. Sierra, J. A. Rodriguez-Aguilar, P. Noriega, M. Esteva, and J. L. Arcos. Engineering multi-agent systems as electronic institutions. *European Journal for the Informatics Professional*, 5:33 –39, 2004.

[18] J. von Neumann and O. Morgenstern. *The Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[19] M. Wooldridge. *An Introduction to MultiAgent Systems Second Edition*. John Wiley & Sons, 2009.