Argumentation-Based Multi-Agent Decision Making with Privacy Preserved

Yang Gao
Institute of Software, Chinese Academy of Sciences
gaoyang@iscas.ac.cn

Francesca Toni
Imperial College London, UK
f.toni@imperial.ac.uk

Hao Wang, Fanjiang Xu
Institute of Software, Chinese Academy of Sciences
{wanghao, fanjiang}@iscas.ac.cn

ABSTRACT

We consider multi-agent decision making problems in which agents need to communicate with other agents to make socially optimal decisions but, at the same time, have some private information that they do not want to share. *Abstract argumentation* has been widely used in both single-agent and multi-agent decision making problems, because of its ability for reasoning with incomplete and conflicting information. In this work, we propose an abstract argumentation-based knowledge representation and communication protocol, such that agents can find socially optimal strategies by only disclosing the 'necessary' and 'disclosable' information. We prove that our protocol is sound, efficient, of perfect information security and guaranteed to terminate.

Keywords

Abstract Argumentation; Distributed Constraints Satisfaction; Privacy Preservation; Social Optimality

1. INTRODUCTION

In many *cooperative multi-agent decision making* problems, agents may have limited sensory capabilities and there may exist some restrictions on actions (e.g. some actions cannot be performed in certain states, and some actions need to be performed by a certain number of agents simultaneously); therefore, agents need to communicate with each other, so as to share some information and to coordinate their actions to meet the constraints [27]. Even though these agents are cooperative, they may have some private information that they do not want to share with other agents. This type of problem is interesting not only because many real problems can be viewed as instances thereof, e.g. time scheduling problems [22], freight scheduling problems [19] and sensor networks [34], but also because it involves information sharing, conflict resolution as well as privacy preservation at the same time.

Abstract argumentation frameworks (AFs, c.f. [8, 4]) are natural techniques to model and solve this type of problems in a novel way. Since AFs can naturally represent and reason with conflicting information, they have been widely used to model functionalities in both single-agent (e.g. [3]) and multi-agent (e.g. [14]) decision making problems. Furthermore, because of the dialectic nature of AFs, argumentation dialogue models [23, 9] have been proposed so that agents can jointly resolve conflicts or make decisions by exchanging only the necessary information (see, e.g., [12]). In this paper, inspired by algorithms for solving *distributed constraint satisfaction problems* (DisCSPs; see, e.g., [32]), we propose an AF-based cooperation protocol, which not only uses AFs to represent each agent's beliefs and observations, but also uses AF-based dialogues to regulate communication between agents. We prove that, by using our protocol, after finitely many rounds of communication, agents can find a strategy profile efficiently, by exchanging only the necessary information and limiting the disclosure of private information. The resulting strategy profile is **(i)** 'feasible', i.e. all actions in the strategy profile are 'doable' according to all agents' observations and beliefs; **(ii)** 'acceptable', i.e. meeting all constraints; and **(iii)** socially optimal, i.e. there does not exist another strategy profile that satisfies requirements **(i)** and **(ii)** and makes every agent 'better off'.

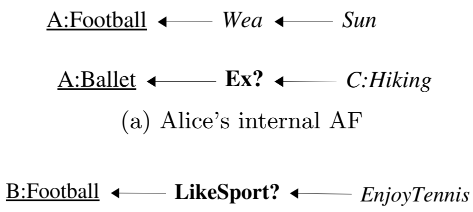
Throughout, we use, for illustration, the following motivating scenario, adapted from the *Battle of the Sexes* game in [25] and the meeting scheduling problem in [28].

Example 1. Alice and Bob want to decide on an activity for the day. Alice prefers going to a ballet to watching a football game, while Bob's preference is the opposite. Both Alice and Bob want to go to the same place rather than different ones. This is a simple DisCSP, but where both agents have private concerns they would rather not disclose.

As for Alice, she worries about Bob's ex-wife attending the ballet (we denote this concern by **Ex?**), but she does not want Bob to know about this concern. Caroline, Bob's and his ex-wife's daughter, told Alice earlier that she went hiking today with her mother (denoted by *C:Hiking*). Also, Alice worries about the weather for the day (denoted by *Wea*), but she found from a forecast that it will be sunny (denoted by *Sun*). Alice does not mind disclosing these two pieces of weather-related information. Alice's beliefs and observations can be modelled by the AF in Fig. 1(a), where arrows represent *attacks* (e.g., **Ex?** attacks <u>A:Ballet</u>, since Bob's ex-wife attending the ballet is a reason against Alice and Bob going, in Alice's mind). Note that this AF does not represent preferences (see Sect. 3.2 for more on this).

As for Bob, he worries about whether Alice likes watching sports games (denoted by **LikeSport?**): Alice has told him before, but he forgets and would rather Alice not be aware of this! However, they went to a tennis match last week and Alice enjoyed that (denoted by *EnjoyTennis*). Also, Bob

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016), J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.), May 9–13, 2016, Singapore.*



(a) Alice's internal AF

(b) Bob's internal AF

Figure 1: AFs that represent (a) Alice's and (b) Bob's beliefs and observations (respectively).

notices that his daughter Caroline has posted on Facebook 'In the ballet hall now!' (denoted by *C:Facebook*). Bob's beliefs and observations (but, again, not preferences) can be modelled by the AF in Fig. 1(b).

Note that *C:Facebook* is in conflict with *C:Hiking*. However, since Alice and Bob do not know each other's arguments a priori, no attack from *C:Facebook* to *C:Hiking* is present in Fig. 1. However, as soon as they share these arguments, an attack will naturally emerge.

We present a protocol for agents equipped with AFs with private and disclosable beliefs and observations, as in this example, to communicate and coordinate so as to solve DisCSPs, such as the one in this example, so as to guarantee social optimality, taking all agents' preferences into account, while preserving privacy in some sense.

2. BACKGROUND

We summarise abstract argumentation frameworks, argumentation dialogues and distributed constraints satisfaction problems in Sect. 2.1, 2.2 and 2.3, respectively. Finally, we give background on information security in Sect. 2.4. This will be used to characterise our protocol as 'privacy preserving'.

2.1 Abstract Argumentation Frameworks

An *Abstract argumentation Framework* (AF) [8] is a pair (Arg, Att) where Arg is a set of *arguments* and $Att \subseteq Arg \times Arg$ is a binary relation representing the *attacks* between arguments. An AF is typically visualised as a graph, as in Fig 1. A set $S \subseteq Arg$ attacks an argument $b \in Arg$ when some member of S attacks b. A set $S \subseteq Arg$ is *conflict-free* iff no member of S attacks a member of S.

Argumentation semantics are defined as criteria for selecting sets of 'winning' arguments in AFs. The *admissible semantics* is widely used: in $\mathsf{F} = (Arg, Att)$, $S \subseteq Arg$ is *admissible* iff S is conflict-free and attacks any argument in Arg that attacks an argument in S. We say that an argument is admissible (in F) iff it belongs to at least one admissible set.

Although the admissible (and other) semantics can identify sets of 'winning' arguments, it does not 'explain' why an argument belongs to a 'winning' set. To tackle this, the *related-admissible* semantics [13] has been proposed. In (Arg, Att), $S \subseteq Arg$ is *related-admissible* iff S is admissible and $\exists a \in S$ such that every argument in S *defends* a (such a is a *topic* of S), where argument b defends a iff:

- $b = a$; or
- $\exists c \in Arg$ s.t. c attacks a and b attacks c; or
- $\exists c \in Arg$ s.t. c defends a and b defends c.

A related-admissible set S is a *compact explanation of* a iff a is a topic of S and S is the smallest (wrt. $\subseteq$) among all related-admissible sets whose topics include a.

Example 2. Consider Bob's AF in Fig. 1(b). The set {**LikeSport?**, *EnjoyTennis*} is not admissible as it is not conflict-free. The set {<u>*B:Football*</u>, *EnjoyTennis*, *C:Facebook*} is admissible but not related-admissible since *C:Facebook* does not defend <u>*B : Football*</u> or *EnjoyTennis*. {<u>*B : Football*</u>, *EnjoyTennis*} is related-admissible and a compact explanation of <u>*B:Football*</u>. Also, {<u>*B : Ballet*</u>} is a compact explanation of <u>*B : Ballet*</u>.

Similarly, in Alice's AF (Fig. 1(a)), there are two compact explanations: {*C:Hiking*, <u>*A : Ballet*</u>} for argument <u>*A : Ballet*</u>, and {*Sun*, <u>*A:Football*</u>} for argument <u>*A : Football*</u>.

2.2 Argument Games and Disputes

Argumentation semantics define the 'winning' sets of arguments, but do not specify how to obtain them. Inspired by the dialectic nature of argumentation, *argument games* (also known as *dialogues*) are proposed as a proof theory for argumentation semantics (see, e.g., [18, 20]). An argument game involves two players[1]: a *defender* and a *challenger*. The defender starts with an argument to be tested, after which each player must attack the other player's arguments. This process is known as a *dispute*. The *two parties immediate response dispute* (TPI-dispute) [30, 9] is a widely used dialogue model. A player in a TPI-dispute is allowed to present an argument (only in the first move) or present an argument attacking one of the previous arguments of the other player. An argument is said to be *defended* by a TPI-dispute iff this argument is the first move in this TPI-dispute and the defender wins this dispute (as for the definition of 'win', see [30]). An argument is admissible iff it can be defended in every TPI-dispute. When there are finitely many arguments, TPI-disputes are guaranteed to terminate.

2.3 DisCSPs

DisCSPs are problems where knowledge (i.e., domains, variables and constraints) is distributed among communicating agents and cannot be centralized for various reasons (e.g. prohibitive costs of constraint translation or security/privacy issues) [32]. *Chronological synchronous backtracking* (SBT) is one of the simplest while most fundamental algorithms for DisCSPs; many modern algorithms for DisCSPs, e.g. asynchronous backtracking and its variants, have their roots in SBT [32]. SBT requires a static ordering of agents. Following this, agents try to extend a partial solution into a total one by adding consistent assignments for unassigned variables. Agents pass a token among them; the agent who has the token can extend the partial solution and then send the extended partial solution and the token to the next agent. SBT terminates either because all variables are included in the partial solution or because every value for the variable of the first agent has been discarded. In the former case, the partial solution constitutes a solution for the problem, while in the latter case, the problem is unsatisfiable. SBT is guaranteed to terminate and is sound and complete (i.e. it terminates only with correct answers, and it terminates for all problems) [17].

[1]Later in Sect. 5 we will see that the players are different from the agents in the problems we consider.

2.4 Information Entropy and Security

The extent to which a certain secret can be preserved, or the level of *information security*, has been rigorously studied in *information theory* [26, 7]. *Perfect security* is the highest privacy preserving level, in which the information disclosed by an agent does not provide any 'hints' to other agents about the private information of the agent, even when the other agents have unlimited computational resources.

The *(Shannon) entropy* [26] is arguably the most widely used measurement to quantitatively evaluate the information loss and the level of security. Formally, the entropy of a discrete random variable X with possible values $\{x_1, \cdots, x_n\}$ and probability mass function $P(X)$ is defined as

$$H(X) = -\sum_i p(x_i) \cdot \log_2 p(x_i)$$

where $p(x_i)$ is a shorthand for $P(X = x_i)$. Note that when $p(x_i) = 0$ for some i, the value of the corresponding summand is taken to be 0, i.e. $0 \cdot \log_2(0) = 0$. As an extension, the *conditional entropy* of two events X and Y is defined as

$$H(X|Y) = \sum_{i,j} p(x_i, y_j) \cdot \log_2[p(y_j)/p(x_i, y_j)]$$

where $p(x_i, y_j)$ is the probability that $X = x_i, Y = y_j$.

Perfect security can be defined using these notions of entropy, as follows. Suppose *Agent* has a secret message M, and *Agent* discloses information C according to some communication protocol Pr; Pr is of *perfect security wrt.* M iff $H(M) = H(M|C)$, i.e. the prior probability of M is the same as its posteriori probability given the disclosure of C.

3. PROBLEM DEFINITION

We first give a formal definition of the type of DisCSPs we consider in this paper (Sect. 3.1), and then introduce the AFs we use to model each agent (Sect. 3.2). Finally, we define the strategy profiles we are after (Sect. 3.3).

3.1 Problem Description

The problems we consider are tuples (Ag, Act, Con), where $Ag = \{Agent_1, \cdots, Agent_N\}$ is the set of agents ($N \in \mathbb{N}, N \geq 2$), $Act = \{a_1, \cdots, a_M\}$ is the set of available actions for each agent ($M \in \mathbb{N}, M \geq 1$), and $Con : Act \rightarrow 2^{\{0,\cdots,N\}}$ is the constraint function. Given an action $a \in Act$, $Con(a) \subseteq \{0, \cdots, N\}$ is the required number of agents to perform this action.[2] For example, if an action a can be performed by no agents or by two agents simultaneously, $Con(a) = \{0, 2\}$. In the reminder, unless stated otherwise, we let i and j range over all agents and all actions, respectively. A *strategy profile* S is a list $\langle ac_1, \cdots, ac_N\rangle$, such that $ac_i \in Act$ is the action of $Agent_i$.

We say that a strategy profile S is *acceptable* for a problem if it satisfies all constraints in the problem. We say that a problem is *satisfiable* if all its constraints can be satisfied by some (acceptable) strategy profile(s). Formally, for a strategy profile S and an action a, let $N(\mathsf{S}, a)$ be the number of agents assigned to perform a in S. Then:

[2] We only consider constraints on the number of agents simultaneously performing an action. More complex constraints in DisCSPs (see, e.g., [6]) are left for future work.

Definition 1. A strategy profile S is *acceptable* for problem (Ag, Act, Con) iff $\forall a \in Act$, $N(\mathsf{S}, a) \in Con(a)$. A problem (Ag, Act, Con) is *satisfiable* if there exists an acceptable strategy profile for it.

Example 3. In Example 1, $N = 2$ because there are two agents (Caroline and her mother are not counted as agents as they do not participate in the decision making). Also, $M = 2$ as there are two actions ('go to the ballet' and 'watch football'). Con('go to the ballet') $=$ Con('watch football') $= \{0, 2\}$ because both agents want to attend these activities together or not at all. This problem is satisfiable, as there are two acceptable strategy profiles: both agents going to the ballet and both agents watching football.

3.2 AFs for Knowledge Representation

We assume that each agent has a complete pre-order expressing a preference on all available actions, and we term $Agent_i$'s preference on actions the *ideal preference* of $Agent_i$, denoted by $\leq^i_{Ideal}$. $<^i_{Ideal}$ is a total order derived from $\leq^i_{Ideal}$, such that for $a_p, a_q \in Act$, $a_p <^i_{Ideal} a_q$ iff $a_p \leq^i_{Ideal} a_q$ and $a_q \not\leq^i_{Ideal} a_p$. This derived total order is used later in Sect. 3.3 when we define social optimality. Also, each agent has information (including beliefs and observations) for or against some actions or other information, represented by *internal AFs*. Agents will share elements of their internal AFs via our cooperation protocol. The *perfect-view internal AF* of an agent includes all arguments that other agents may disclose during cooperation. In this subsection, we define these concepts formally.

Arguments in internal AFs.

In line with some widely used AF-based decision making paradigms [3, 5], we distinguish two categories of arguments: *practical* and *epistemic* arguments. Practical arguments recommend actions to agents. For simplicity, for each action and agent, we allow for one and only one practical argument to recommend this action to the agent. We let A_{ij} denote the (one and only) practical argument recommending action a_j to $Agent_i$, and denote the set of all practical arguments belonging to $Agent_i$ by $pPra_i$ (p is short for 'private', and Pra stands for 'Practical'), i.e. $pPra_i = \{A_{i1}, \cdots, A_{iM}\}$. Practical arguments are *private* since, in our protocol, agents directly inform other agents about their action choices (in line with SBT, see later in Sect. 4) and, thus, they do not need to additionally communicate the reasons for these choices (expressed by the practical arguments).

As for the epistemic arguments, we distinguish two subcategories: *disclosable* and *private* arguments, which an agent is willing and unwilling to disclose to other agents, respectively. We denote $Agent_i$'s disclosable and private epistemic argument sets by $dEpi_i$ (d is short for 'disclosable') and $pEpi_i$, respectively.

We denote $Agent_i$'s argument set by $Arg_i = pPra_i \cup dEpi_i \cup pEpi_i$. We impose that $pPra_i$, $dEpi_i$ and $pEpi_i$ are mutually disjoint and that Arg_i is finite. In Fig. 1, practical arguments are underlined, private epistemic arguments are in boldface and disclosable epistemic arguments are in italic. How to build these arguments from beliefs and raw observations is not the focus of this paper; in principle, any structured argumentation technique, e.g. ABA [29] or ASPIC+ [21], can be used for this purpose.

Internal AFs.

We first discuss attacks within an agent's internal AF. In line with [3], we impose that practical arguments do not attack each other, so as to preserve all relevant information in the decision process (see Sect. 2.3 in [1] for a thorough explanation). Since a practical argument supports an agent to perform an action, while an epistemic argument justifies beliefs and observations on which practical arguments are built, in line with [3], practical arguments are not allowed to attack (private or disclosable) epistemic arguments.

As for attacks between epistemic arguments, since private arguments usually represent agents' private beliefs/concerns, whereas disclosable arguments usually represent observations/facts (see Example 1), and we believe that the latter are 'stronger' than the former, we do not allow private arguments to attack disclosable arguments; we leave the relaxation of this constraint for future work. Finally, each private epistemic argument is attacked by some disclosable (epistemic) arguments. This guarantees that, when negotiating with other agents, other agents can 'indirectly' defend or attack each private epistemic argument by (directly) attacking or defending some disclosable arguments attacking it. This will be further discussed in Sect. 5.

We now define the internal AF for each agent.

Definition 2. *$Agent_i$'s internal AF* is $AF_i = (Arg_i, Att_i)$, such that $Arg_i = pPra_i \cup dEpi_i \cup pEpi_i$ and

1. $\forall a, b \in pPra_i$, $(a, b) \notin Att_i$;
2. $\forall a \in pPra_i$, $\nexists b \in pEpi_i \cup dEpi_i$ such that $(a, b) \in Att_i$;
3. $\forall a \in pEpi_i$, $\nexists b \in dEpi_i$ such that $(a, b) \in Att_i$; and
4. $\forall a \in pEpi_i$, $\exists b \in dEpi_i$ such that $(b, a) \in Att_i$.

In line with our earlier discussion: condition 1 gives that the set of all practical arguments is conflict-free, condition 2 gives that practical arguments are not allowed to attack epistemic arguments, condition 3 gives that private arguments are not allowed to attack disclosable arguments, and condition 4 gives that each private argument is 'publicly defeasible' during coordination, as discussed earlier.

Example 4. The AF in Fig. 1(a) is an internal AF, as required by Def. 2. Indeed, there are no attacks between practical arguments (condition 1), no attacks from practical arguments to epistemic arguments (condition 2) and no attacks from private arguments to disclosable arguments (condition 3). Also, the only private epistemic argument, **Ex?**, is attacked by the disclosable epistemic argument *C:Hiking*, thus condition 4 holds too. Similarly, it is easy to check that the AF in Fig. 1(b) also satisfies the requirements of Def. 2.

Attacks across internal AFs/Perfect-view internal AFs.

Now we discuss attacks between arguments of different agents. Since an agent cannot know another agent's private arguments, we restrict attacks between agents' arguments only to their disclosable (epistemic) arguments, i.e. given a practical or private epistemic argument of $Agent_i$, this argument cannot attack or be attacked by another agent's arguments. In addition, we assume that agents have a consensus on the attack relation between (disclosable) arguments (irrespective of the agents they belong to). This assumption is in line, e.g., with [25]. Note that with this assumption we do not mean that an agent knows another agent's disclosable epistemic arguments a priori; instead, we mean that if two disclosable arguments are presented in front of an agent, this agent is able to decide the attack relation between them, and this attack relation is agreed by all other agents. We denote the attack relation between different agents' arguments by Att^*, such that $Att^* \subseteq \bigcup_i dEpi_i \times \bigcup_i dEpi_i$.

	Disclosable?	Allowed to attack which arguments?
$pPra_i$	No	$\emptyset$
$dEpi_i$	Yes	$dEpi_{-i} \cup Arg_i$
$pEpi_i$	No	$pPra_i \cup pEpi_i$

Table 1: Properties of different kinds of arguments (from $Agent_i$'s perspective).

Let $dEpi_{-i} = \bigcup_{p \neq i} dEpi_p$, for $p \in \{1, \cdots, N\}$. Namely, $dEpi_{-i}$ consists of all disclosable epistemic arguments belonging to agents other than $Agent_i$. Properties of different kinds of arguments are summarised in Table 1. We now define the perfect-view internal AF for each agent.

Definition 3. *$Agent_i$'s perfect-view internal AF* is $AF_i^* = (Arg_i^*, Att_i^*)$, where $Arg_i^* = dEpi_{-i} \cup Arg_i$ and $Att_i^* = Att_i \cup Att^*$.

Example 5. Alice's perfect-view internal AF (Arg_A^*, Att_A^*) has $Arg_A^* = Arg_A \cup \{EnjoyTennis, C{:}Facebook\}$ and $Att_A^* = Att_A \cup \{(C{:}Facebook, C{:}Hiking)\}$, where (Arg_A, Att_A) is the AF in Fig. 1(a). Bob's perfect-view internal AF can be obtained similarly.

Note that an agent may not be able to know the full contents of its perfect-view internal AF during cooperation. Nevertheless, this notation allows to define desirable properties of the protocol.

3.3 Desirable Properties

We want for the strategy profiles obtained by our protocol to meet the following requirements: **i)** *feasible*, i.e. if a strategy profile assigns action a_j to $Agent_i$, then a_j should be 'doable' for $Agent_i$; **ii)** *acceptable*, i.e. the strategy profiles satisfy all constraints; and **iii)** *socially optimal* wrt. agents' ideal preferences. We have already defined acceptability in Def. 1. Let us now define the other requirements.

Definition 4. An action a_j is *locally feasible* for $Agent_i$ iff argument A_{ij} (the practical argument that recommends a_j to $Agent_i$) is admissible in $Agent_i$'s internal AF.

If an action is not locally feasible for $Agent_i$, it is not supported by the information held by $Agent_i$ and, thus, $Agent_i$ has no incentive to propose it. We say that $Agent_i$'s internal AF is *feasible* iff all actions in Act are locally feasible for $Agent_i$. For example, we can see that every practical argument in Fig. 1 is admissible in the internal AF it belongs to. Thus, both Alice's and Bob's internal AFs are feasible. We assume that all agents' internal AFs are feasible; we leave relaxing this restriction for future work.

Definition 5. An action a_j is *(globally) feasible* for $Agent_i$ iff argument A_{ij} is admissible in (Arg_i^*, Att_i^*).

A strategy profile $\mathsf{S} = \langle ac_1, \cdots, ac_N \rangle$ is *feasible* iff, for each $i \in \{1, \cdots, N\}$, action ac_i is globally feasible for $Agent_i$.

Now we define the notion of 'social optimality' as follows:

Definition 6. $\mathsf{S} = \langle ac_1, \cdots, ac_N \rangle$ is *socially optimal* iff:
- S is acceptable and feasible; and
- there is no $\mathsf{S}' = \langle ac'_1, \cdots, ac'_N \rangle$ s.t. $\mathsf{S}' \neq \mathsf{S}$, S' is acceptable and feasible, and for any i $ac_i <^i_{Ideal} ac'_i$.

Example 6. As we mentioned in Example 3, there are two acceptable strategy profiles: both agents 'going to the ballet' or 'watching football'. We first check whether $\underline{A : Ballet}$ is feasible, i.e. whether it is admissible in Alice's perfect-view internal AF (given in Example 5). Easily, we can see that $\underline{A : Ballet}$ is not admissible, because it is attacked by **Ex?**, which, in turn, is defended by *C:Facebook*. Thus, the strategy profile wih both agents 'going to the ballet' is not feasible and not a desirable profile we want to obtain. Instead, we can easily verify that both agents 'watching football' is feasible; since this is the only strategy profile that meets the first two requirements, it is socially optimal by Def. 6.

Finally, we want to avoid agents' 'privacy leaks' during coordination. In this paper, this amounts to preventing that agents' private epistemic arguments are disclosed.[3] Thus, we require that **(iv)** our protocol is *of perfect security wrt. agents' private epistemic arguments* (as defined in Sect. 2.4). Next, we propose a protocol that can obtain strategy profiles with all desirable properties identified in this section.

4. THE COORDINATION PROTOCOL

The pseudo code of our protocol is presented in Alg. 1. The overall structure of this protocol is similar to that of SBT (see Sect. 2.3).[4] Without loss of generality, we assume that agents with smaller index numbers are higher in the static agent ordering required by SBT.

Let us walk through this protocol. This is activated by sending the token and a message **ok?**(S) to $Agent_1$. $Agent_1$ initialises S to an empty strategy profile (line 1). *my_act* stores $Agent_i$'s action choice, and is initialised to *null* (line 2). When an agent receives the token (line 3), it iterates over all actions that are equally or less preferred than its current action *my_act* by invoking function GetNextBestAct (line 4). When the input action of GetNextBestAct is *null*, it returns (one of) the agent's most preferred action(s). For the current action a, the agent checks whether it is *consistent*[5] with S and (globally) feasible using function Check (line 5), which is detailed later in Alg. 2.

After obtaining a consistent and feasible action a, the agent adds $(Agent_i, a)$ into S, updates *my_act* (line 6), and sends the updated S and the token to the next agent if the agent is not the last (line 8) or returns the obtained full strategy profile otherwise (line 10). If the action being checked is not consistent and feasible, the agent checks the next.

However, if the agent fails to find any consistent and feasible action (lines 14 to 18), it sends message **ngd** (short for 'not good') and the token to the previous agent so as to backtrack (line 15). In particular, if $Agent_1$ cannot find a feasible and consistent action and needs to backtrack, the protocol terminates and no strategy is obtained (line 17).

Algorithm 1 The protocol ($Agent_i$'s perspective).

1: Initialise S to an empty strategy profile if i is 1
2: $my\_act \leftarrow null$
3: **if** I have the token **then**
4:   **for** $a \leftarrow$ GetNextBestAct(my_act) **do**
5:     **if** Check$(a, S, i) == [consistent, feasible]$ **then**
6:       update S and *my_act*
7:       **if** $i < N$ **then**
8:         send **ok?**(S) and token to $Agent_{i+1}$
9:       **else**
10:         **return** S
11:       **end if**
12:     **end if**
13:   **end for**
14:   **if** $i > 1$ **then**
15:     send **ngd** and the token to $Agent_{i-1}$
16:   **else**
17:     **return** *null*
18:   **end if**
19: **end if**

Example 7. We now use Alg. 1 to coordinate Alice and Bob's actions. Suppose Alice is ranked higher in the static agent ordering; thus, she obtains the token first (line 3). Alice checks whether her most preferred action — going to the ballet — is consistent with S (note that S is empty at this stage) and globally feasible, by invoking Check (line 5). We omit the checking process here (see Examples 8 and 9), but its result suggests that going to the ballet is not feasible; Alice thus checks her next preferred action: watching football. By invoking Check again, Alice finds that watching football is consistent with S and is feasible. So she inserts (Alice,'watch football') into S and lets *my_act* be 'watch football' (line 6). Then she sends **ok?**(S) and the token to Bob (line 8).

Once Bob receives the token and Alice's message (line 3), he begins to propose his own action. Similarly, he starts from his favourite action: watching football (line 4), and he finds that this action is consistent with S (note that S now includes Alice's proposal) and is feasible by invoking Check (line 5). Since Bob is the last agent, after he updates S and *my_act* (line 6), he returns the full strategy profile and the whole protocol terminates (line 10).

Now we introduce the function Check (invoked in line 5 in Alg. 1), which checks an action's consistency and (global) feasibility. Its pseudo code is given in Alg. 2. It first checks the consistency of assignment $(Agent_i, a)$ wrt. the input partial strategy (line 2), as follows: it first counts the number of agents action a has been assigned to in S (we denote this number by $N(\mathsf{S}, a)$; see Sect. 3.1), and then compares $N(\mathsf{S}, a)$ with $\max Con(a)$, i.e. the maximal number of agents required to perform a (see Sect. 3.1 for the definition of $Con(a)$). When $i \neq N$ (i.e. the current agent is not the last agent), we say that assignment $(Agent_i, a)$ is *consistent* with S iff $N(\mathsf{S}, a) \leq \max Con(a)$; when $i = N$, the assignment is *consistent* with S iff adding this assignment into S generates an acceptable strategy profile (see Def. 1). The function updates *status* accordingly (line 3).

[3]Preserving other kinds of information may also be of interest in certain applications: for example, the agents may also want to avoid directly informing other agents about their proposed actions, or they may have private constraint functions that they do not want to disclose. Privacy preservation has been widely studied in DisCSPs (e.g. see [16, 6, 28, 31, 33]; more details are given in Sect. 7). We leave the study of further notions of privacy preservation as future work.

[4]More advanced algorithms for DisCSPs can be used as the basis for our protocol; we leave this for future work.

[5]Roughly speaking, assigning action a_j to $Agent_i$ is consistent with a partial strategy profile S if merging this assignment into S does not violate any constraints. Details are given later in this section when defining Check.

Algorithm 2 CHECK(action a, strategy S, agent $Agent_i$)

```
1: status ← [not_consistent, not_feasible]
2: if assignment (Agent_i, a) is consistent with S then
3:    let the first element in status be consistent
4: end if
5: Obtain all compact reasons E_1, ··· , E_K for a
6: for each E_k, k = 1, ··· , K do
7:    if Discussion suggests that E_k is successful then
8:       let the second element in status be feasible
9:    end if
10: end for
11: return status
```

Example 8. We illustrate how Alice checks the consistency of her first proposed action: going to the ballet. Note that the input strategy profile S is empty at this stage. Since $Con(\text{go to the ballet}) = \{0, 2\}$ (see Example 3), there can be at most two agents performing this action. Because Alice is not the last agent, she only needs to check whether there still exists any allowance for her to perform this action, i.e. she needs to check whether $N(\mathsf{S}, \text{go to the ballet}) \leq \max Con(\text{go to the ballet})$. Because $N(\mathsf{S}, \text{go to the ballet}) = 0$, her proposed action is consistent with S.

To illustrate the last agent's consistency check, we walk through how Bob checks whether his proposed action 'watch football' is consistent. Note that, in this case, the input partial strategy profile S includes Alice's proposal (watch football). After Bob inserts his proposal into S, the new strategy profile is $\mathsf{S}' = \langle \text{watch football}, \text{watch football} \rangle$, in which the first and second action are Alice's and Bob's proposed action, respectively. Thus, $N(S', \text{go to the ballet}) = 0$ and $N(S', \text{watch football}) = 2$, both satisfying the constraints (see Example 3); according to Def. 1, this strategy profile is acceptable and Bob's action proposal is thus consistent.

Function CHECK also checks the (global) feasibility of the input action (line 5 to 10). The intuition of this part is as follows: an agent obtains all '*compact reasons*' for the input action (line 5; here we suppose there are $K \in \mathbb{N}$ compact reasons), and '*discusses*' with other agents to see whether there exists any '*successful*' compact reason (line 7); the action is feasible if there exists (at least) one successful compact reason (line 8). In the next section, we detail the concept of 'compact reasons', 'discussion' and 'successful', and prove that the CHECK function is sound and it terminates.

5. EXPLAINING ACTION CHOICES WITH PRIVACY PRESERVED

In this section, we analyse which arguments need to be disclosed so as to check the feasibility of an action without leaking the private arguments. An intuitive method to find which arguments need to be disclosed is to compute the 'reasons' for an action; then the agent can 'discuss' these reasons with other agents, so that other agents only need to disclose their arguments 'relating to' these reasons. We use the related-admissible semantics (see Sect.2.1) to compute the reasons for actions. The next proposition shows that, for each related-admissible set that includes one and only one practical argument, the practical argument is the set's topic (see Sect. 2.1).

PROPOSITION 1. *Given $Agent_i$'s internal AF (Arg_i, Att_i), if $S \subseteq Arg_i$ is related-admissible and $S \cap pPra_i = \{A_{ij}\}$, then A_{ij} is the topic of S.*

PROOF. We prove this by contradiction. Suppose the topic of S is argument A, and $A \neq A_{ij}$. Since A_{ij} is the only practical argument in S, A is an epistemic argument. By condition 2 in Def. 2, practical arguments cannot attack epistemic arguments; thus, A_{ij} cannot defend A (for defend, see Sect. 2.1); however, by assumption, A is the topic of S and every element in S should defend S. Contradiction. □

Given $Agent_i$'s internal AF, if one of its related-admissible sets S includes only one practical argument A_{ij}, we say that *S recommends action a_j to $Agent_i$*. If S recommends a_j to $Agent_i$ and is a compact explanation (see Sect. 2.1) of A_{ij}, we say that $S \setminus \{A_{ij}\}$ is an *$Agent_i$'s compact reason for a_j*. In the next proposition, we prove that each non-empty compact reason includes disclosable arguments.

PROPOSITION 2. *For $Agent_i$'s internal AF (Arg_i, Att_i), if $S \subseteq Arg_i$, S is a compact reason for a_j and $S \neq \emptyset$, then $S \cap dEpi_i \neq \emptyset$.*

PROOF. We prove this by contradiction. Suppose $S \cap dEpi_i = \emptyset$; thus, all epistemic arguments in S are private. However, $\forall a \in S, \exists b \in dEpi_i$ s.t. $(b, a) \in Att_i$ (condition 4 in Def. 2). As private arguments are not allowed to attack disclosable arguments (condition 3 in Def. 2), S cannot attack b and S cannot be a compact reason. Contradiction. □

We say that a compact reason $E \subseteq Arg_i$ for action a_j is *successful* for $Agent_i$ (line 7 in Alg. 2) iff every disclosable argument in E is contained in (at least) one admissible set in $Agent_i$'s perfect-view internal AF (Arg_i^*, Att_i^*) (see Sect. 3.2). In the next proposition, we show that an action is feasible if it has (at least) one successful compact reason.

PROPOSITION 3. *Suppose E is an $Agent_i$'s compact reason for action a_j and it consists of l ($l \in \mathbb{N}, l \geq 1$) disclosable arguments $d_1, \cdots, d_l$. If $\exists S \subseteq Arg_i^*$ such that $\{d_1, \cdots, d_l\} \subseteq S$ and S is admissible, then action a_j is feasible, i.e. argument A_{ij} is admissible in $AF_i^* = (Arg_i^*, Att_i^*)$.*

PROOF. We denote $E \cup \{A_{ij}\}$ by E'. To prove this proposition, we show that $\exists S' \subseteq Arg_i^*$ s.t. $E' \subseteq S'$ and S' is admissible in AF_i^*. $\forall a \in Arg_i^*$ that attacks $b \in E'$:

- If $a \in Arg_i$, since E' is related-admissible, E' attacks a.
- Else, $a \in Arg_i^* \setminus Arg_i$, i.e. $a \in dEpi_{-i}$, then b must be a discloable argument (i.e. $b \in \{d_1, \cdots, d_l\}$) because another agent's arguments are not allowed to attack $Agent_i$'s private arguments (see Table 1). Because S is admissible in AF_i^* and $b \in S$, there exists $c \in S$ such that c attacks a. We prove that $E' \cup \{c\}$ is conflict-free.
  - If $c \in dEpi_i$, suppose $E' \cup \{c\}$ is not conflict-free. Because E' is related-admissible, $\exists d \in \{d_1, \cdots, d_l\}$ such that d attacks c. Because both d and c are in S, S is not conflict-free. Contradiction. Thus, If $c \in dEpi_i$, $E' \cup \{c\}$ is conflict-free.
  - Else, $c \in Arg_i^* \setminus Arg_i$, i.e. $c \in dEpi_{-i}$. Because c is in S, $\{d_1, \cdots, d_l\} \cup \{c\}$ is conflict-free; also, since c is not allowed to be in an attack relation with $Agent_i$'s private arguments, $\{c\} \cup (E' \cap Pri)$, for Pri the set of all $Agent_i$'s private (epistemic and practical) arguments, is conflict-free. Thus, when $c \in dEpi_{-i}$, $\{c\} \cup E'$ is conflict-free.

Thus, for every a that attacks $b \in E'$, there exists some argument c attacking a and $E' \cup \{c\}$ is conflict-free. Therefore, A_{ij} is admissible in AF_i^*. □

Proposition 3 explains lines 6 to 9 in Alg. 2: by checking whether there exists a successful compact reason for action a_j, we can check the feasibility of a_j. However, to check whether a compact reason is successful, an agent needs to obtain some necessary disclosable arguments from other agents. We term this process *Discussion* (line 7 in Alg. 2).

We use a variant of the TPI-dispute (see Sect. 2.2) to implement Discussion. There are two (opponent) players in the dispute: a *defender* and a *challenger*. Note that these two players are different from the agents; instead, they are only used in the dispute to ask agents to provide some of their disclosable arguments, so as to find whether a compact reason is successful. Also, note that even inactive agents (i.e. agents that do not have the token) can disclose their arguments to the players, upon players' requests. Our variant of TPI-dispute, called *compact-reason-oriented TPI-dispute*, is formally defined as follows.

Definition 7. Suppose E is an $Agent_i$'s compact reason for action a_j, and E contains l ($l \in \mathbb{N}, l \geq 1$) disclosable arguments $d_1, \cdots, d_l$. An *E-oriented TPI-dispute* satisfies the following requirements:

- In the first move, the defender puts forward arguments $d_1, \cdots, d_l$.
- After that, players take turns to ask all agents except $Agent_i$ whether they have disclosable arguments attacking one of the arguments presented earlier by the opponent player; if they do, the asking player presents one of these arguments (following the rules given below); otherwise, we say that the player *runs out of moves*.
- Both players are allowed to *backtrack*, i.e. present an argument attacking some earlier argument of the other player, called the *backtracked* argument.
- Both players are allowed to repeat arguments earlier presented by the defender.
- The challenger is allowed to repeat arguments presented earlier by itself iff those repeated arguments are in different *dispute lines*[6] from their earlier uses.
- No other moves are allowed.
- The challenger wins the dispute iff it does an *eo ipso* move (i.e. uses a previous non-backtracked argument of the other player) or the defender runs out of moves. The defender wins iff the challenger cannot win.

Example 9. We illustrate the discussion between Alice and Bob (line 7 in Alg. 2) that checks the feasibility of Alice's actions. Since Alice has two compact explanations (see Example 2), we can easily obtain her compact reasons: $\{C{:}Hiking\}$ is the compact reason for $\underline{A : Ballet}$, and $\{Sun\}$ is the compact reason for $\underline{A : Football}$ (since each action has one compact reason, $K = 1$ in line 6, Alg. 2). We walk through the $\{C{:}Hiking\}$-oriented TPI-dispute as follows (**D1**/**C1** labels the first step of the defender/challenger, respectively.):

- **D1**: The defender puts forward *C:Hiking*.
- **C1**: The challenger asks Bob whether he has argument(s) attacking *C:Hiking*, and Bob reports *C:Facebook*. Thus, the challenger presents *C:Facebook*.

After step **C1**, since the defender runs out of moves, the challenger wins the dispute and, thus, the compact reason $\{C{:}Hiking\}$ is not successful and $\underline{A : Ballet}$ is not feasible.

Instead, the compact reason for $\underline{A : Football}$, i.e. $\{Sun\}$, is successful, as the challenger cannot make any moves after the defender puts it forward. Thus, $\underline{A : Football}$ is feasible.

Proposition 4. *If E is an $Agent_i$'s compact reason for action a_j, E is successful iff the defender wins in every E-oriented TPI-dispute.*[7]

Proof. (Sketch) We first prove that if the defender wins in every E-oriented TPI-dispute, E is successful. Because the challenger cannot win, the arguments presented by the defender are conflict-free (otherwise, the challenger can perform eo ipso) and can attack every arguments presented by the challenger (otherwise, the defender runs out of moves). Conversely, if E is successful, suppose S is an admissible set and $E \subseteq S$; then the defender can win every dispute by using arguments in S only. □

Since all moves other than the first in our disputes are the same as in standard TPI-disputes, and there are finitely many arguments presented in the first move, we have:

Proposition 5. *If E is an $Agent_i$'s compact reason for action a_j, every E-oriented TPI-dispute terminates.*

6. PROPERTIES OF THE PROTOCOL

Theorem 1. *Alg. 1 is guaranteed to terminate.*

Proof. (Sketch) Function Check employs compact-reason-oriented TPI-disputes (Def. 7) to check whether compact reasons are successful, and the disputes are guaranteed to terminate (Proposition 5); thus, Check terminates. Also, because the basic structure of Alg. 1 is the same as that of SBT, and all functions in Alg. 1 terminate, Alg. 1 is guaranteed to terminate. □

Theorem 2. *Alg. 1 is sound, i.e. the strategy profile obtained by this protocol is socially optimal (see Def. 6).*

Proof. (Sketch) The SBT structure of Alg. 1 ensures that the obtained strategy profile is acceptable. Propositions 3 and 4 ensure that all obtained strategy profiles are feasible. Since agents propose their actions according to their ideal preferences (line 5 in Alg. 1), the obtained strategy is the most preferred feasible and acceptable strategy wrt. agents' ideal preferences. Thus, the strategy profile obtained by Alg. 1 is socially optimal. □

However, note that our current protocol is not complete: it may return no solutions when there exist some. For example, consider a problem consisting of two agents and one action a_1. The constraint is that a_1 should be performed by both agents. For $Agent_1$'s internal AF (Arg_1, Att_1), $Arg_1 = \{A_{11}, d_1, d_2\}$, $dEpi_1 = \{d_1, d_2\}$, $pPra_1 = \{A_{11}\}$ and $Att_1 =$

[6] A dispute line is a dispute where each move replies to the immediately preceding move.

[7] Due to the space limit, we sketch or omit some proofs.

$\{(d_1, A_{11}), (d_2, d_1)\}$[8]. For $Agent_2$'s internal AF (Arg_2, Att_2), $Arg_2 = \{A_{21}, d_3, p_1\}$, $pPra_2 = \{A_{21}\}$, $dEpi_2 = \{d_3\}$, $pEpi_2 = \{p_1\}$ and $Att_2 = \{(p_1, A_{21}), (d_3, p_1)\}$. Also, we let d_3 attacks both d_1 and d_2. Thus, $Agent_1$ and $Agent_2$'s compact reason for their practical arguments are $\{d_2\}$ and $\{d_3\}$, respectively. We can see that $Agent_1$'s compact reason is not successful (line 7 in Alg. 2) as d_2 is attacked by d_3, but A_{11} is actually admissible in $Agent_1$'s perfect-view internal AF. Thus, our protocol will return no strategy profile in this example, while 'both agents perform a_1' is actually a strategy profile that meets all our requirements (see Sect. 3.3). We leave improving completeness for future work.

THEOREM 3. *Alg. 1 is of* perfect security *wrt. agents' private epistemic arguments.*

PROOF. For $Agent_i$ and $Agent_h$, $i, h \in \{1, \cdots, N\}$, $i \neq h$, let $\mathbb{P}_h^i = \{a | Agent_h$ believes a may be in $pEpi_i\}$. Because $Agent_h$ has no information about the cardinality of $pEpi_i$ (denoted by $|pEpi_i|$), $|\mathbb{P}_h^i|$ is $+\infty$. Similarly, we let $\mathbb{D}_h^i = \{a | Agent_h$ believes a may be in $dEpi_i\}$; then $|\mathbb{D}_h^i| = +\infty$.

Thus, for any $a \in \mathbb{P}_h^i$, the probability that a is contained in $pEpi_i$ (denoted by $p_h(a \in pEpi_i)$) is $|pEpi_i|/|\mathbb{P}_h^i|$, because each argument in $\mathbb{P}_h^i$ has the same probability to be included in $pEpi_i$. Thus, $p_h(a \in pEpi_i) = 0$. Although $Agent_j$ may be aware of some argument $b \in dEpi_i$, since $p(a \in pEpi_i, b \in dEpi_i) = 0$ and $p(b \in dEpi_i) = |dEpi_i|/|\mathbb{D}_h^i| = 0$, by Eq. (2.4), $H(a \in pEpi_i) = H(a \in pEpi_i | b \in dEpi_i) = 0$. The protocol is thus perfect secure wrt. private arguments. $\square$

Since, in our compact-reason-oriented TPI-disputes, only arguments attacking earlier-presented arguments can be presented, we can easily see that the communication between agents is 'efficient', i.e. only arguments that *defend* (see Sect. 2) the practical arguments are communicated.

THEOREM 4. *For $Agent_i$ to check the feasibility of action a_j using function* CHECK, *if an argument $d \in dEpi_{-i}$ is presented by the defender in a compact-reason-oriented TPI-dispute (Def. 7), d defends argument A_{ij} in AF_i^*.*

7. RELATED WORK

Our work is related to the *distributed constraint optimisation* problems (DisCOPs) [10], in which variables and constraints are distributed among a set of agents who must communicate to find an optimal assignment of values for the variables. The acceptability and social optimality requirements can be represented in DisCOPs, but the feasibility requirement cannot, because agents' domain knowledge (including beliefs and observations) about the feasibility of actions is not considered in DisCOPs.

Some work has investigated the types of privacies and methods to preserve private information in DisCSPs and DisCOPs, e.g. [16, 6, 28, 31, 33]. Two types of privacies are widely identified and studied: i) *constraint privacy*, i.e. the constraints are partially known by each involved agent, and ii) *assignment privacy*, i.e. each agent does not directly let other agents know its own assignment. The privacy we consider is not covered by either of these two types, because we consider the agents' private beliefs information (beliefs and observations). Methods to preserve privacies fall into two categories: i) *privacy enforcement*, i.e. only disclose the necessary information so as to minimise privacy loss; and ii) *encrypted message*, i.e. encrypt some messages so that only certain agents can understand them. Our protocol falls into the privacy enforcement category.

Grant et al. [15] consider a variant of Boolean games, in which agents cannot observe some *environment variables*; a human operator can disclose certain environment variables' values and preserve the others, so as to influence agents' decisions, in the hope of obtaining (socially) optimal strategy profiles. Their work is not as generic as ours, as they focus on Boolean games and propositional logic-based knowledge representation. Also, the privacy they considered is centralised (pertaining only to the environment) while the privacy we consider is distributed (pertaining to every agent).

Amgoud and Devred [2] represented AFs as DisCSPs, and used solvers for DisCSPs to compute 'winning' sets of arguments (wrt. some different semantics). Our work integrates argumentation and DisCSPs in a different way: we use algorithms from DisCSPs to guarantee the acceptability of the strategy profiles, and use argumentation to represent agents' domain knowledge on the feasibility of actions and to regulate communication between agents.

Some work has been devoted to efficient argumentation-based negotiation. Fan and Toni [11] introduced a dialogue protocol based in ABA [29], which does not require agents to put forward all their domain knowledge. Pasquier et al. [24] introduced *interest-based negotiation*, in which agents not only communicate their requirements, but also the reasons (i.e. interests) behind the requirements, so that agents can cooperate to find better alternative solutions. However, none of these works considers agents' private information, e.g. agents' private arguments.

8. CONCLUSION AND FUTURE WORK

We consider a class of cooperative multi-agent decision making problems, and propose an abstract argumentation based knowledge representation and communication protocol that is sound, efficient, guaranteed to terminate and privacy-preserving. Our work suggests that abstract argumentation can be used to represent agents' domain knowledge that includes private information; upon this knowledge representation, agents can communicate and coordinate their actions 'efficiently' and 'securely' (without leaking personal information) by using argumentation-based dialogues. Our work connects argumentation, distributed constraint satisfaction/optimisation and game theory in a novel manner, suggesting that several multi-agent coordination techniques can be used together to tackle complex problems.

We identify two major lines for extending this work: **i)** stick to cooperative multi-agent decision-making problems, and improve the current protocol, e.g. relax some restrictions, improve the completeness of the protocol, or use more advanced DisCSPs algorithms as the basis of the protocol; **ii)** consider more complex problems, and adjust the current framework accordingly; for example, allowing agents to have different action sets, or tuning the 'selfishness' of each agent, so that agents cooperate and compete simultaneously.

Acknowledgements

F. Toni was supported by the EPSRC TRaDAr project *Transparent Rational Decisions by Argumentation*: EP/J020915/1.

[8]For $a, b \in Arg_i$, $(a, b) \in Att_i$ is read 'a attacks b'.

REFERENCES

[1] L. Amgoud and C. Cayrol. Inferring from inconsistency in preference-based argumentation frameworks. *Journal of Automated Reasoning*, 29(2):125–169, 2002.

[2] L. Amgoud and C. Devred. Argumentation frameworks as constraint satisfaction problems. In S. Benferhat and J. Grant, editors, *SUM, LNAI*. Springer, 2011.

[3] L. Amgoud and H. Prade. Using arguments for making and explaining decisions. *Artificial Intelligence*, 173(3):413–436, 2009.

[4] T. J. M. Bench-Capon and P. E. Dunne. Argumentation in artificial intelligence. *Artificial Intelligence*, 171:619–641, 2007.

[5] T. J. M. Bench-Capon and H. Prakken. Justifying actions by accruing arguments. *Frontiers in Artificial Intelligence and Applications*, 144:247, 2006.

[6] I. Brito, A. Meisels, P. Meseguer, and R. Zivan. Distributed constraint satisfaction with partially known constraints. *Constraints*, 2008.

[7] T. M. Cover and J. A. Thomas. *Elements of information theory, 2nd edition*. Wiley-Interscience, 2006.

[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(2):321–357, 1995.

[9] P. E. Dunne and T. J. M. Bench-Capon. Two party immediate response disputes: Properties and efficiency. *Artificial Intelligence*, 149(2):221–250, 2003.

[10] B. Faltings. Distributed constraint programming. In T. Walsh, editor, *Handbook of Constraint Programming*. Elsevier, 2006.

[11] X. Fan and F. Toni. Assumption-based argumentation dialogues. In *Proceedings of International Joint Conferences on Artificial Intelligence*, 2011.

[12] X. Fan and F. Toni. A general framework for sound assumption-based argumentation dialogues. *Artificial Intelligence*, 2014.

[13] X. Fan and F. Toni. On computing explanations in argumentation. In *Proceedings of AAAI Conference on Artificial Intelligence*, 2015.

[14] X. Fan, F. Toni, A. Mocanu, and M. Williams. Multi-agent decision making with assumption-based argumentation. In *Proceedings of Thirteenth International Conference on Autonomous Agents and Multiagent Systems*, 2014.

[15] J. Grant, S. Kraus, M. Wooldridge, and I. Zuckerman. Manipulating games by sharing information. *Studia Logica*, 102:267–295, 2014.

[16] T. Grinshapoun and T. Tassa. A privacy preserving algorithm for distributed constraint optimisation. In *Proceedings of International Conference on Autonomous Agents and Multiagent Systems*, 2014.

[17] P. Harvey. *Solving Very Large Distributed Constraint Problems*. PhD thesis, University of Wollongong, 2009.

[18] H. Jakobovits and D. Vermeir. Dialectic semantics for argumentation frameworks. In *Proceedings of International Conference on Artificial Intelligence and Law*, 1999.

[19] T. Leaute and B. Faltings. Distributed constraint optimization under stochastic uncertainty. In *Proceedings of AAAI Conference on Artificial Intelligence*, 2011.

[20] R. Loui. Process and policy: resource-bounded nondemonstrative reasoning. *Computational Intelligence*, 14(1):1–38, 1998.

[21] S. Modgil and H. Prakken. The ASPIC+ framework for structured argumentation: a tutorial. *Argument and Computation, Special Issue: Tutorials on Structured Argumentation Computation*, 5(1):31–62, 2014.

[22] J. Modi and M. Veloso. Multiagent meeting scheduling with rescheduling. In *Distributed Constraint Reasoning Workshops, DCR*, 2004.

[23] S. Parsons and P. McBurney. Argumentation-based communication between agents. In *Communications in Multiagent Systems*. Springer, 2003.

[24] P. Pasquier, R. Hollands, I. Rahwan, F. Dignum, and L. Sonenberg. An empirical study of interest-based negotiation. *Journal of Autonomous Agents and Multi-Agent Systems*, 2010.

[25] I. Rahwan and K. Larson. Argumentation and game theory. In *Argumentation in Artificial Intelligence*. Springer, 2009.

[26] C. E. Shannon. *The Mathematical Theory of Communication*. University of Illinois Press, 1949.

[27] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009.

[28] M. Silaghi and D. Mitra. Distributed constraint satisfaction and optimization with privacy enforcement. In *Proceedings of IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT)*, 2004.

[29] F. Toni. A tutorial on assumption-based argumentation. *Argument and Computation, Special Issue: Tutorials on Structured Argumentation*, 5(1):89–117, 2014.

[30] G. Vreeswijk and H. Prakken. Credulous and sceptical argument games for preferred semantics. In *Proceedings of European Conference on Logics in Artificial Intelligence*, 2000.

[31] R. J. Wallace and E. C. Freuder. Constraint-based reasoning and privacy/efficiency tradeoffs in multi-agent problem solving. *Artificial Intelligence*, 161:209–227, 2005.

[32] M. Yokoo and K. Hirayama. Algorithms for distributed constraint satisfaction: A review. *Journal of Autonomous Agents and Multi-Agent Systems*, 2000.

[33] M. Yokoo, K. Suzuki, and K. Hirayama. Secure distributed constraint satisfaction: Reaching agreement without revealing private information. *Artificial Intelligence*, 161:229–245, 2005.

[34] W. Zhang, Z. Xing, G. Wang, and L. Wittenburg. Distributed stochastic search and distributed breakout: properties, comparison and applications to constraints optimization problems in sensor networks. *Artificial Intelligence*, 16:55–87, 2005.