A dialogue mechanism for public argumentation using conversation policies

Yuqing Tang
Dept. of Computer Science
Graduate Center, City University of New York
365 Fifth Avenue
New York, NY 10016, USA
ytang@gc.cuny.edu

Simon Parsons
Dept of Computer & Information Science
Brooklyn College, City University of New York
2900 Bedford Avenue
Brooklyn, NY 11210 USA
parsons@sci.brooklyn.cuny.edu

ABSTRACT

In this paper, we propose a flexible dialogue mechanism through which a set of agents can establish a coherent set of public beliefs. Flexibility and coherence are achieved by decomposing the dialogue mechanism into two parts, a backbone protocol and a set of conversation policies. The backbone protocol maintains the set of arguments put forward by the agents, and each agent uses a pre-agreed argumentation theory to extract the set of public beliefs from the context. The flexibility is achieved by distributing the other functions of the dialogue mechanism among a set of conversation policies, some of which are public and some of which are private to each agent.

1. INTRODUCTION

Multiagent systems need a mechanism by they can communicate in order to coordinate their efforts to achieve tasks that are assigned to the system [31]. Furthermore this mechanism should be flexible enough to enable a human designer to incrementally add more and more building blocks to the mechanism as understanding of the task evolves and the task itself changes. For the communication mechanism, many approaches have been proposed — see [18, 25] for surveys. Argumentation based dialogues [4, 23, 25] have proved to be a general approach to agent communication in which the agents exchange not only statements of what they believe and what they want but also the reason why. In this approach, a *protocol* or a *conversation policy* is used to govern the valid sequences of dialogue moves and then argumentation-based reasoning is used by individual agents to resolve the conflicts arising from the information that they hold privately and the messages they receive.

The set of beliefs held by the agent society as a whole can be implicitly induced from the common beliefs that all the agents have obtained by their own private argumentation. However, this causes two interrelated problems. Firstly, the specification of the protocol is not independent of the agent's internal specification, rather it is hard-wired into the agents. Secondly, it is hard to see the impact of the compositions of dialogue protocols on each individual agent's beliefs as well as on the implicit public belief set [18]. These problems will prevent the dialogues from achieving the desiderata and criteria of a good dialogue as suggested in [18] and [20], such as flexibility and verifiability.

In response to these problems, we propose a flexible dialogue mechanism through which the agents can establish a coherent public belief set. Flexibility and public coherence is achieved together by decomposing the mechanism into two parts — a backbone protocol and a set of conversation policies. The backbone protocol maintains a shared context of messages that have been exchanged between agents in the form of arguments and defeats, and each agent uses a pre-agreed argumentation theory to extract the public belief set from this set of messages. Flexibility is achieved by distributing the remaining functions of the dialogue mechanism among a set of conversation policies. Some of these policies will need to be obeyed publicly to regulate the kinds of argument and defeat that can be asserted into the dialogue context, and other policies will be private to each individual agent and be used to decide which arguments and defeats should be generated out of the agent's own private information base. The public aspect of conversation policies is to have the agents cooperate together to achieve the set of public beliefs. The private aspect of conversation policies is to offer freedom and flexibility for individual agents to solve the problems from different perspectives.

Cite as: A dialogue mechanism for public argumentation using conversation policies, Yuqing Tang and Simon Parsons, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. XXX-XXX.

2. RELATED WORK

Argumentation theory, which is used to establish the public set of beliefs in our mechanism, has been used in a range of ways in artificial intelligence in general [11, 10] and multiagent communications in particular [4, 23, 25]. In artificial intelligence in general, argumentation-based reasoning has been used to unify a number of approaches to nonmonotonic reasoning [11], to reason about uncertainty [17], to reason coherently from inconsistency [2], to perform decision making and practical reasoning [8], to handle conflicting desires [1] and so on [5].

These applications of argumentation based reasoning suggest that the approach is a solution for resolving conflicts arising from agent communication about issues involving uncertainty, inconsistency, decision making, practical reasoning and so on — all the things that researchers have shown that can be handled using argumentation. However, the most interesting property of argumentation based reasoning to us is the concept of "external stability" [11] through which a set of coherent beliefs is characterized by the relations between the arguments "internally" supporting the beliefs and the arguments "externally" supporting the contradictory beliefs. In an agent society, because of the diversity and the dynamics of the situated environment, messages from different agents and messages from the same agent at different times often convey conflicting information. In many cases, these conflicts can not be resolved by just looking at consecutive messages in a dilaogue. Therefore most dialogue protocols — for example [23] — spend a lot of effort in making rules about asserting statements into and retracting the

statements from the set of public beliefs in terms of commitments to maintain coherency of those beliefs. In contrast, in our approach, argumentation, with the "external stability" property, ensures the coherence of the public belief set almost for free by just choosing different semantics and different computation methods for different applications.

Furthermore, there are many recent advances in argumentation based reasoning such as the work of Cayrol and Lagasquie-Schiex [9] , Jakobovits and Vermeir [15], Besnard and Hunter [7], Pollock [22], which have expanded our understanding of what argumentation can be used for, and have created a bridge to possibility theory and plausibility theory in the field of reasoning about uncertainty [13]. The new systems that have emerged from this research are not usable in existing argumentation-based dialogue systems because of the way that the latter tightly couple the dialogue protocol with the underlying argumentation theory. The dialogue mechanism proposed in this paper paves the way to use these new argumentation reasoning theories freely in dialogues by decoupling dialogue from the underlying argumentation.

3. AN ARGUMENTATION FRAMEWORK

An *argumentation framework* is a pair $\langle \mathcal{A}, \mathcal{R} \rangle$ where $\mathcal{A}$ is a set of arguments, and $\mathcal{R}$ is a binary defeat relation over the set of arguments. The set of arguments are induced from an information base, denoted by Σ. Σ is represented in a logical language $\mathcal{L}$ with the standard connectives $\wedge, \vee, \neg, \equiv$. An entailment relationship $\vdash$ is required to be defined on $\mathcal{L}$. Inconsistent information is allowed in Σ to accommodate conflicting information in the information base. The defeat relation $\mathcal{R}$ will be induced from Σ to recapture the inconsistency of the information base at the level of arguments. Once we have the set of arguments and the set of defeats, we adopt a set of principles, principles drawn from the philosophical and linguistic study of human argumentation and fallacious reasoning [30], that we can use to analyze the outcome of the argument set and the defeat set.

The rest of this section will be devoted to describe the framework and its components formally. The framework is mostly drawn from the work of Amgoud and her colleagues [2, 3] with some slight modifications.

Definition 1. An argument based on Σ is pair (H, h) where $H \subseteq \Sigma$ such that

1. H is consistent with respect to $\mathcal{L}$,
2. $H \vdash h$,
3. H is minimal (for set inclusion).

H is called the support and h is called the conclusion of the argument. $\mathcal{A}(\Sigma)$ denotes the set of all arguments which can be constructed from Σ.

This definition of argument can be understood as constraints on how pieces of coherence information can be clustered as arguments. Condition (1) is to ensure that an argument is a coherent. The coherence of an agent's information is defined in terms of the consistency of the language $\mathcal{L}$ in which the information is written. Condition (2) can be understood as insisting that the conclusion of an argument should be supported by a set of information in the sense of inference in the language $\mathcal{L}$. Condition (3) can be understood as saying that no redundant information should appear in an argument. This definition of argument is chosen from Amgoud's work because its form is simple. Our proposed dialogue mechanism in Section 4 doesn't prevent the application from choosing another form of argument as long as there is a process to generate the arguments and check their validity.

Definition 2. (H', h') is a subargument of the argument (H, h) iff $H' \subseteq H$.

Definition 3. Let (H_1, h_1), (H_2, h_2) be two arguments of $\mathcal{A}(\Sigma)$.

1. (H_1, h_1) rebuts (H_2, h_2) iff $h_1 \equiv \neg h_2$.
2. (H_1, h_1) undercuts (H_2, h_2) iff $\exists h \in H_2$ such that $h \equiv \neg h_2$.
3. (H_1, h_1) contradicts (H_2, h_2) iff (H_1, h_1) rebuts a subargument of (H_2, h_2).

The binary relations *rebut*, *undercut*, and *contradict* gather all pairs of arguments satisfying conditions (1), (2) and (3) respectively.

The relations *rebut*, *undercut*, and *contradict* will be collectively referred to as *defeat* if no distinction is necessary.

The definition of these forms of defeat can be viewed as recapturing the inconsistency of the original information into a conflict relation among the arguments in terms of the fallacious reasoning recorded in the arguments. *rebut* means that the two arguments leads to conflicting conclusions in the sense of $\mathcal{L}$. *undercut* means that the one argument's conclusion is conflicting with another argument's premise. *contradict* means that one argument's conclusion is conflicting with a conclusion which can be extended, using the inferences in $\mathcal{L}$, from one or many segments of another argument's support. In contrast to *rebut* and *undercut*, *contradict* penetrates into arguments and explores various parts of the arguments to detect conflicting points with respect to $\mathcal{L}$.

These notions of defeat are close, but none is equivalent to, or subsumes, the other in general. If we define arguments of the form $(\{a\}, a)$, where the conclusion is also the support, to be *degenerate*, then we can easily show that:

Proposition 1. Let (H_1, h_1) and (H_2, h_2) be two arguments.

1. If (H_1, h_1) rebuts (H_2, h_2) then it also undercuts (H_2, h_2) iff (H_2, h_2) is degenerate.
2. If (H_1, h_1) undercuts (H_2, h_2) then it contradicts (H_2, h_2) iff (H_2, h_2) is degenerate,

PROOF. We can easily see the equivalence of rebut, contradict and undercut on a degenerate argument from an example. $(H_1, \neg a)$, rebuts, undercuts and contradicts $(\{a\}, a)$. In general, for rebuttal to entail undercut, the conclusion has to be in the support, and by the minimality condition on arguments, the undercut/rebutted argument must therefore be degenerate. Similarly, for undercut to entail contradiction, the element of the support that is attacked by the undercutter must also be the conclusion of the undercut argument. Hence it must be degenerate. □

[2] gives a detailed discussion on how these definitions of defeat will affect the behaviors of an argumentation framework, while [28, 29] provide a more detailed discussion on the concepts and forms of defeat. In later sections we will only use *undercut*.

Following Dung's work [11], we have the following component definitions of the theory.

Definition 4. An *argumentation framework* is a pair, $Args = \langle \mathcal{A}, \mathcal{R} \rangle$, where $\mathcal{A}$ is a set of arguments, and $\mathcal{R}$ is a binary defeat relation over the arguments.

Definition 5. Let $\langle \mathcal{A}, \mathcal{R} \rangle$ be an argumentation framework, and $S \subseteq \mathcal{A}$. An argument A is defended by S iff $\forall B \in \mathcal{A}$ if $(B,A) \in \mathcal{R}$ then $\exists C \in S$ such that $(C,B) \in \mathcal{R}$.

Definition 6. $S \subseteq \mathcal{A}$. $\mathcal{F}_{\mathcal{R}}(S) = \{A \in \mathcal{A} | A$ is defended by S with respect to $\mathcal{R}\}$.

Now, for a function $F : D \rightarrow D$ where D is the domain and the range of the function, a fixed point of F is an $x \in D$ such that $x = F(x)$. When the D is associated with an ordering P — for example, P can be set inclusion over the power set D of arguments — x is a *least fixpoint* of F if x is a least element of D with respect to P and x is a fixed point.

Definition 7. Let $\langle \mathcal{A}, \mathcal{R} \rangle$ be an argumentation framework. The set of acceptable arguments, denoted by $Acc^F_{\mathcal{R}}$, is the least fixpoint of the function $\mathcal{F}_{\mathcal{R}}$ with respect to set inclusion.

The least fixpoint semantics can be viewed as a mathematical translation of the principle such that an argument survives if it can defend itself and be defended by a set of arguments which can also survive all the attacks made upon them.

It is possible to provide alternative sematics for argumentation systems. For example we have the numerical characterization in [7], the string (or tree) characterization in [9], a characterization based on Dempster-Shafer theory [16], and the algebra based characterization [22]. Others are surveyed in [9].

In terms of engineering the reasoning system, given the language $\mathcal{L}$ it should be sufficient to describe the application domain. The concept of argument and defeat that are selected should be such that the logical property of arguments and the defeat defined on them should be strong enough to capture sufficient conflicting patterns of information in the application at the level of arguments. In addition, the argumentation semantics that are selected should have an appropriate power to produce a set of acceptable arguments which corresponds to the set of correct answers in the application domains. In the following sections, we propose a mechanism to lay out the backbone of a shared argumentation reasoning system and build different conversation policies on top of it.

4. A DIALOGUE MECHANISM

4.1 The backbone protocol

In this section, we define a flexible dialogue mechanism that decomposes a dialogue into a backbone protocol and a set of conversation policies. This dialogue mechanism serves a set of agents $\mathcal{T} = \{T_1, ..., T_n\}$ where each agent T_i is equipped with an information base $\Sigma(T_i)$. The set of conversation policies is a set of facilities, some of which are interrelated, to produce arguments and defeats out of the information base and feed them into the backbone protocol. The job of the backbone protocol is then to maintain a unified dialogue context of arguments and defeats between all the agents, and to provide an interface for the agents to query the public beliefs drawn from the context using a pre-agreed argumentation semantics. In contrast to the existing protocols, which enforce all the requirements on the structure of a conversation sequence, the backbone protocol only assures the integrity and validity of arguments and defeats exchanged and leaves the other requirements of the dialogue to conversation policies. As shown in Figure 1, the components of the mechanism can be divided into two layers — the public layer and the private layer — from the view of whether the components can be accessed and verified publicly by the the agents, and whether they require public cooperation among the agents to maintain their functions. The public layer is composed of a backbone protocol and a set of public conversation policies[1]; the private layer is composed of the agents' information bases and a set of private conversation policies. We will discuss conversation policies in section 4.2.

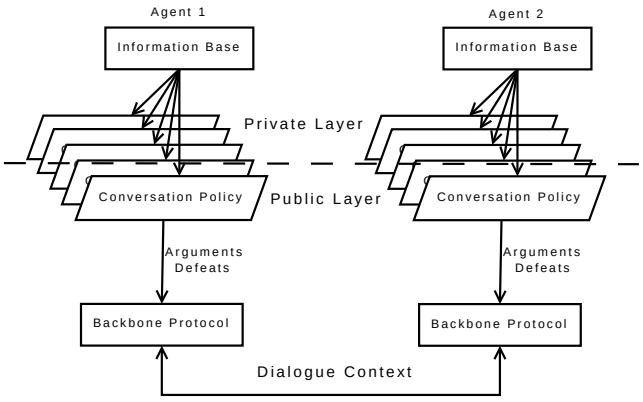

Figure 1: A dialogue between two agents

The backbone protocol organizes the messages exchanged by agents as a shared set of valid arguments and defeats, and then uses some agreed argumentation semantics to draw public beliefs out of the messages. The prerequisites for using the protocol are that all the agents share the same language $\mathcal{L}$, share the same definition of arguments and defeats, and share the same argumentation semantics. The central notion in the backbone protocol is the *dialogue context*, denoted by C, the shared set of arguments and defeats as well as their supports. C is triple

$$\langle C_\Sigma, C_\mathcal{A}, C_\mathcal{R} \rangle$$

where C_Σ is the set of formulae that have been exchanged; $C_\mathcal{A}$is the set of arguments that have been identified; and $C_\mathcal{R}$ is the set of defeat relations that have been identified. For convenience, we also write $C = C_\Sigma \cup C_\mathcal{A} \cup C_\mathcal{R}$.

The implementation of the dialogue context will depend heavily on how conversations between the agents are organized. Here we assume that the configuration only allows pair-wise agent communication. Under this assumption, one of many ways to implement the dialogue context C distributively is to have each agent T_i maintain a copy $C^i = C^i_\Sigma \cup C^i_\mathcal{A} \cup C^i_\mathcal{R}$ of the context C, and regulate the agents to access and modify the context only through a set of pairwise communication locutions:

Definition 8. Basic pairwise communication locutions:

send(T_i, T_j, φ)

- Precondition: none.
- T_i updates $C^i_\Sigma = C^i_\Sigma \cup \{\varphi\}$.
- T_j updates $C^j_\Sigma = C^j_\Sigma \cup \{\varphi\}$.

send$(T_i, T_j, (H,h))$

- Precondition: $H \subseteq C^i_\Sigma$, $h \in C^i_\Sigma$, and (H,h) is an argument according to Definition 1.

[1]We use the term, protocol, to name the set of rules that governs the overall structure of a dialogue instance, and use the term, conversation policy, to name the set of rules that governs, possibly partially, the local structure a segment of a dialogue instance following [19]. However, it is common to use the two terms interchangeably in the literature. See [18, 19] for more discussions.

- T_i updates $C^i_{\mathcal{A}} = C^i_{\mathcal{A}} \cup \{(H,h)\}$.
- T_j updates $C^j_{\mathcal{A}} = C^j_{\mathcal{A}} \cup \{(H,h)\}$.

send$(T_i, T_j, (H,h)$ *defeat* $(H',h'))$

- Precondition: $(H,h) \in C^i_{\mathcal{A}}$ and $(H',h') \in C^i_{\mathcal{A}}$, namely the arguments should already exist in the communication context, and (H,h) *defeat* $(H',h'))$ is a defeat according to Definition 3.
- T_i records updates $C^i_{\mathcal{R}} = C^i_{\mathcal{R}} \cup \{(H,h)$ *defeat* $(H',h'))\}$.
- T_j updates $C^j_{\mathcal{R}} = C^j_{\mathcal{R}} \cup \{(H,h)$ *defeat* $(H',h'))\}$.

query(T_i, T_j, h)

- Precondition: None
- T_i asks himself and T_j to stop sending formulae, arguments, and defeats into the context.
- The agents compute simultaneously whether there is an argument (H,h) in the set of acceptable arguments $Acc_{C^i_{\mathcal{A}}, C^i_{\mathcal{R}}}$ according to the argumentation framework $\langle C^i_{\mathcal{A}}, C^i_{\mathcal{R}} \rangle$.

Proposition 2. The contents of C^i for $i \in \{1,\ldots,n\}$ are identical if the context is only manipulated by the locutions defined in Definition 8 .

PROOF. Immediate by construction. □

The following are a set of macro locutions constructed from the pairwise locutions in Definition 8 and invoked as primitives (i.e. no other pairwise locutions can be invoked by any agent during each macro):

send(T_i, φ): Invoke *Send*(T_i, T_j, φ) for every agent T_j, $j \neq i$.

send$(T_i, (H,h))$: Invoke *Send*$(T_i, T_j, (H,h))$ for every agent T_j, $j \neq i$.

send$(T_i, (H,h)$ *defeat* $(H',h'))$: Invoke *Send*$(T_i, T_j, (H,h)$ *defeat* $(H',h'))$ for every agent T_j, $j \neq i$.

query(T_i, h): Invoke *Query*(T_i, T_j, h) for every agent T_j, $j \neq i$.

Proposition 3. For each query *query*(T_i, h), all the participating agents will obtain the same status for h.

PROOF. The status for h evaluated by T_i solely depends on C^i. For all $i \in \{1,\ldots,n\}$, C^i is the same according to Proposition 2, and all T_is share the same semantics of argumentation, therefore all the agents will obtain the same status for h. □

With these locutions, we get the backbone Protocol 4.1. This defines a set of prerequisites that all participating agents must satisfy before the execution of the protocol, and a loop of two execution steps: one to handle arguments and another one to handle defeats. The set of prerequisites is that all the agents must maintain a copy of the context, have a conversation policy *CP* (which will be defined below in section 4.2) plugged in, and pre-agree on a language $\mathcal{L}$, an argumentation system *AS* and its semantics. In the body, the protocol loops through two steps: 1) query the conversation policy for an argument, check its validity, make sure it is new to the context, and then send it into the context by sending its component formulae and the (argument) structure explicitly over these formulae using locutions *send*(T_i, φ) and *send*$(T_i, (H,h))$ respectively; 2) query the conversation policy for a defeat, check its validity, make sure that it is new to the context, and then sent it into the context by sending its explicit structure using the locution $(T_i, A_j$ *defeats* $A_k)$. The loop can be stopped at any time by any agent that needs the argument status of a belief represented by a formula h, then every agent will compute this status based on its own copy of the dialogue context. In this way, the protocol can guarantee that every agent will have the same answer for h.

Protocol 4.1 A Backbone Protocol

Require: (1) each agent T_i is equipped with a dialogue context C^i, (2) each agent T_i is equipped with a conversation policy *CP* (3) all the agents pre-agree on a language $\mathcal{L}$, an argumentation reasoning system *AS* and its semantics,

1: **repeat**

2: Query *CP* for an argument (H,h)

- Check whether (H,h) is a valid argument according to *AS*, if not go to next step
- Check whether $(H,h) \in C^i_{\mathcal{A}}$, if yes go to next step
- Invoke *Send*(T_i, p) for each $p \in H \cup \{h\}$
- Invoke *Send*$(T_i, (H,h))$

3: Query *CP* for a defeat A_j *defeats* A_k

- Check whether A_j and A_k are valid arguments and $A_j, A_k \in C^i$, if not continue the loop
- Check whether A_j *defeats* A_k is a valid defeat, if not continue the loop
- Invoke *send*$(T_i, A_j$ *defeats* $A_k)$

4: **until** *query*(T_k, h) is posted to the dialogue by some agent T_k; after every agent gets the answer, resume the loop

4.2 Conversation policies

In [12], the authors defined the concept of conversation policies as declarative specifications that govern communications between software agents using an agent communication language. We agree with this notion of conversation policy in general, but as we use the conversation policies on top of the backbone protocol defined in the previous section (which is actually also a conversation policy in this general definition), we will define conversation policies as declarative or procedural specifications that govern the production of arguments and defeats to feed into the backbone protocol with respect to different perspectives of the public argument argumentation and the applications on top of it.

There are several dimensions to look at the conversation policies on top of the backbone protocol: 1) the source and mechanism from which the arguments and defeats are generated; 2) whether it is a private policy which only requires an individual effort or whether it is a public policy which requires cooperation among agents; 3) whether it is verifiable; 4) whether it is concerned with general public argumentation, or with application-specific problem solving; and 5) some other considerations. In this paper, we only have room to deal with some of these dimensions.

Given two policies CP_1 and CP_2 we can combine them in the following ways, reminiscent of those suggested for dialogue game protocols in [18, 19].

- sequential: return the arguments (respectively, defeats) produced by CP_1 first; when no more arguments (defeats) can be produced by CP_1, then return those of CP_2
- alternate: one argument or defeat from CP_1, then one from

Policy 5.1 A basic conversation policy

Require: (1) a set of topics of interest $\{h_i\}$ shared by all the agents or held by individual agent, (2) each agent T_i is equipped with a dialogue context C^i and an information base $\Sigma(T_i)$, (3) all the agents pre-agree on a common language $\mathcal{L}$ and an argumentation system *AS* and its semantics,

1: Initialize $I = \{h_i\}$ and maintain a memory of I during the dialogue
2: On request for an argument,

- if I is empty, return *nil*
- select a formula h from I,
- construct an argument $A = (H, h) \notin C^i_{\mathcal{A}}$ based on $\Sigma(T_i) \cup C^i_{\Sigma}$ according to the proof theory of $\mathcal{L}$. If such an argument exists, then return it; otherwise return *nil*
- if all possible arguments for h have been exhausted, let $I = I - \{h\}$

3: Internally decide the defeating points: Select an argument $A = (H, h)$ in $\Sigma_{\mathcal{A}}$, select a formulae in $p \in H \cup \{h\}$, let $I = I \cup \{\neg p\}$ if $\neg p$ is not in I.
4: On request for a defeat, look for two arguments $A_1, A_2 \in C^i_A$ such that A_1 *defeats* $A_2 \notin C^i_{\mathcal{A}}$, if such a defeat exists then return it; otherwise return *nil*

CP_2, continuing to alternate until no further arguments or defeats can be produced.

- filtering: filter the arguments and defeats from CP_1 and CP_2 with respect to some criteria
- preference selection: compare two arguments or defeats obtained from CP_1 and CP_2 respectively, then select one of them according to some preference

5. EXAMPLE POLICIES

5.1 A basic policy

Policy 5.1 is a basic policy, concerned with the general process of public argumentation. It generates arguments and defeats using the reasoning mechanism of $\mathcal{L}$, and it requires no cooperation among agents. It is not possible to verify whether an agent conforms with this policy by analyzing what the agent puts into the dialogue context. Since the criteria to select a formula h from I is unspecified, there is no guarantee that the arguments and defeats put into the dialogue are complete enough to generate a stable argument overall for the topic of interest in terms of the selected argumentation semantics and certainly no guarantee that this will occur within a given amount of time. We can, however, improve on the basic policy.

5.2 Iterative deepening dialogue

Policy 5.2 is an improvement on the basic policy. It will still generate arguments and defeats based on the information in agents' information bases and using the reasoning mechanism of the language $\mathcal{L}$, but it does this by generating arguments and defeats in a specific order in the spirit of iterative deepening search. The policy uses three search parameters to limit the resources that the agents can use to generate arguments and defeats: (1) reasoning depth R_D, which controls the maximum number of inference rules that can be used in building arguments, (2) defeat depth D_d, which controls the maximum number of defeats that may be chained together [2], (3)

[2]A defeat chain takes the form of argument A_1 defeats A_2 which defeats A_3.

Policy 5.2 An iterative deepening dialogue policy

Require: (1) A set of topics: $S = \{h_j\}$, (2) a set of agents T_i each with dialogue context C^i, (3) a pre-agreed reasoning depth increment Δ_R, (4) a pre-agreed defeat depth increment Δ_D, (4) a pre-agreed reasoning breadth increment Δ_B.

1: Initially set reasoning depth $R_D = 1$, reasoning breadth $R_B = 1$, defeat depth $D_d = 1$,
2: Initially set interest points $I = \{h_i\}$
3: Initially set defeat points $D = \emptyset$
4: On request for an argument

- if I is empty, return *nil*
- select a formula h from I,
- construct an argument $A = (H, h)$ based on $\Sigma(T_i) \cup C^i_{\Sigma}$ such that
  - $A \notin C^i_{\mathcal{A}}$
  - A uses at most R_D of the inference rules of $\mathcal{L}$
  - Other than A, there are less than R_B arguments supporting h
- if all possible arguments for h have been exhausted, let $I = I - \{h\}$ if such an argument exists, then return it; otherwise return *nil*

5: On request for a defeat, look for two arguments $A_1, A_2 \in C^i_A$ such that

- A_1 *defeats* $A_2 \notin C^i_{\mathcal{A}}$, and
- the length of any defeat path ended with A_1 is less than D_d,

if such a defeat exists then return it; otherwise return *nil*
6: Internally decide the defeating points: Select an argument $A = (H, h)$ in $\Sigma_{\mathcal{A}}$ such that the length of any defeat path ending with A is less than D_d, select a formulae in $p \in H \cup \{h\}$, let $I = I \cup \{\neg p\}$ if $\neg p$ is not in I.
7: If all the agents can not produce more arguments and defeats, they cooperatively increase the parameters: (1) reasoning depth $R_d \leftarrow R_d + \Delta_R$, (2) reasoning breadth $R_B \leftarrow R_B + \Delta_B$, (3) defeat depth $D_d \leftarrow D_d + \Delta_D$, (4) set interest points back to $I = \{h_i\}$

reasoning breadth R_B, which controls the maximum number of arguments an agent can provide for a conclusion. To apply the policy, we will need the agents to synchronize these parameters cooperatively (if not, this will become a specific case of Policy 5.1). This means that this conversation policy is a public one.

The advantage of this policy is that, since it is an exhaustive search through the arguments and defeats that the set of agents can generate, then if there is a set of acceptable arguments that can be distilled from the set of all arguments that each agent can construct on its own (namely $\cup_i \mathcal{A}(\Sigma(T_i))$, the iterative deepening search policy will reach this set after a finite number of iterations.

Using a similar scheme of maintaining some shared parameters, a more efficient policy than Policy 5.2 can be created based on AND-OR tree evaluation to decide whether an argument is acceptable. If Dung's grounded semantics is used, and the argumentation system is finitary — for every argument there are only finite number of defeat arguments — then we have the same policy as used in [3]). This policy is of polynomial complexity in terms of the number of arguments[3]. Alternatively, if we employ the argument schemes and defeat schemes approach of [26], we may be able to tailor the lan-

[3]This does not imply that the whole argumentation is polynomial in the size of the information base, Σ; the overall time complexity being dependent on checking the validity of an argument which, unless formulae in Σ and the reasoning mechanism are restricted in some way, will not in general be polynomial.

Policy 5.3 A policy to construct arguments cooperatively

Require: Requirements are those in Policy 5.1 or those in the Policy 5.2, and in addition that all the agents cooperatively maintain C_G

1: Function as Policy 5.1 or Policy 5.2, with the additional two steps:
2: For a formula $h \in I$, for which the agent cannot construct an argument, use backward chaining to obtain a proof for h, then select an open formula φ (i.e. not in $C_\Sigma \cup \Sigma(T_i)$) in the proof, and invoke *ask _help*(T_i, φ)
3: Select a $\varphi \in C_G$ such that $\varphi \in \Sigma(T_i)$ but $\varphi \notin C_\Sigma$, invoke *offer_help*(T_i, φ)

guage to generate a polynomial number of arguments for a specific domain, and then in total we will have a polynomial policy in terms of the number argument schemes and defeat schemes.

5.3 Constructing arguments cooperatively

If we use Policies 5.1 and 5.2 then there will be some arguments and defeats which can not be constructed. These are arguments that are constructed using information that is held by different agents, and so is not all available to any single agent. Some of these arguments *might* be constructed by Policies 5.1 and 5.2 — the necessary information being revealed by other arguments that the agents put forward — but there is no guarantee that this will be the case. In general we will need some mechanism to help the agents construct arguments cooperatively, especially in the information seeking, inquiry and deliberation dialogues [30]. The following are the basic constructs for this purpose. To better organize the policy, we decompose some primitive functions of the policy as those in the backbone Protocol 4.1. We need the agents to cooperatively maintain a set of goals C_G, the set of goals waiting for additional information (we can think of this as part of the dialogue context). The content of C_G is different from C_Σ in the sense that it is not the information held by any participating agents, but rather a set of symbols indicating the intention of asking agents to provide information. C_G is maintained by the following locutions:

- *ask_help*(T, φ)
  - precondition: $\varphi \notin \Sigma(T)$ and $\varphi \notin C_\Sigma$,
  - T updates $C_G = C_G \cup \{\varphi\}$.
- *offer_help*(T, φ)
  - precondition: $\varphi \in C_G$, and $\varphi \in \Sigma(T)$
  - T updates $C_\Sigma = C_\Sigma \cup \{\varphi\}$
  - T updates $C_G = C_G - \{\varphi\}$

With this set of locutions we can define Policy 5.3. This policy can be used to gain and provide help in constructing arguments, and works by delegating all the other functions to policies like those we discussed above. In Policy 5.3, we do not specify the conditions under which the agents can ask for help and should offer help. Such conditions will be application specific, and will result in specializations of Policy 5.3 that are used in specific situations.

5.4 A policy for multiagent planning

Our final example, Policy 5.4, is a conversation policy that is application specific, and deals with multiagent planning. The policy handles part of the generation of the arguments and defeats using its knowledge about specific problem — formalised in terms of state transitions, plans, resource conflicts and resource reconfiguration — and delegates the other functions to Policy 5.1 or 5.2.

To demonstrate the policy, we consider a simple multiagent planning problem, concerning two agents T_1 and T_2. Making common assumptions from the AI planning literature [21], both of them characterize the world as a set of precisely observable states S; T_1 and T_2 are capable of performing two sets of actions, A_1 and A_2 respectively. The evolution of the world is modeled as three mutually exclusive state transition functions

$$\begin{aligned} \gamma_1 : S \times A_1^* &\rightarrow S \\ \gamma_2 : S \times A_2^* &\rightarrow S \\ \gamma_3 : S \times (A_1 - A_1^*) \times (A_2 - A_2^*) &\rightarrow Sn \end{aligned}$$

where $A_1^* \subseteq A_1$ and $A_2^* \subseteq A_2$, γ_1 models the state transitions which can be totally controlled by T_1, γ_2 models the state transitions which can be totally controlled by T_2, and γ_3 models the state transitions which can only be controlled cooperatively by the two agents. Two sets of states $G_1 \subseteq S$ and $G_2 \subseteq S$ express the goals of T_1 and T_2 respectively. We denote the set of all possible state transitions as

$$\begin{aligned} \Gamma = \{(s, a, s') | \gamma_i(s, a) = s' \text{ with } i = 1, 2\} \\ \cup \quad \{(s, a_1, s_2, s') | \gamma_3(s, a_1, a_2) = s'\} \end{aligned}$$

A plan p is a pair $\langle \Gamma_p, \pi_p \rangle$ where $\Gamma_p \subseteq \Gamma$ is a set of interesting state transitions and $\pi_p \subseteq S \times A$ is a set of state-actions pairs. π_p is a policy, in the planning sense, prescribing what action to take in each state encountered. We assume that there is a carefully designed language $\mathcal{L}$ such that one argument type is a pair $\langle (\Gamma_p, \pi_p), G \rangle$ where Γ_p and π_p together characterize a plan p, and G characterizes the set of states that can be experienced by the policy π_p. Another argument type is pair of the form $\langle H, \neg(s, a) \rangle$ or $\langle H, \neg(s, a_1, a_2) \rangle$ which means that H is a set of formulae in the language that characterize the resource conflicts which prevent the action a or cooperative action pair (a_1, a_2) from being performed in the state s, and one more argument type is a pair of the form $\langle H, (s, a) \rangle$ or $\langle H, (s, a_1, a_2) \rangle$ which means that H characterizes a configuration of resources which enables the action a or (a_1, a_2) in the state s. Therefore we have two types of defeat: (1) an argument $\langle H, \neg(s, a) \rangle$ defeats another argument $\langle (\Gamma_p, \pi_p), G \rangle$ when $(s, a) \in \pi_p$, and (2) an argument $\langle H, (s, a) \rangle$ or $\langle H, (s, a_1, a_2) \rangle$ defeats another argument $\langle H, \neg(s, a) \rangle$. For simplicity, we assume that nothing can defeat the argument of the form $\langle H, (s, a) \rangle$ or $\langle H, (s, a_1, a_2) \rangle$.

We further assume that the set of all possible arguments is $\mathcal{A}$. There is a set of arguments $A_G \subseteq \mathcal{A}$ such that all the arguments in A_G have the same conclusions $G_1 \cup G_2$. For every argument $a \in A_G$, there is an argument in $b \in A_C$ with conclusions saying that some action in a is not performable because of resource conflict; there is subset $A'_C \subset A_C$ such that for every argument $a \in A'_C$ there is an argument $b \in A_R$ saying there is a resource configuration which will make sure that the action in a's conclusion can be performed. Therefore the set of acceptable arguments A_{acc} is the set of all $a \in A_G$ such that if there is some $b \in A_C$ that defeats a then another $c \in A_R$ defeats b. In this setting, A_{acc} corresponds to the set of plans that can achieve the two agents' goals G_1 and G_2 simultaneously. Assume that T_i $(i = 1, 2)$ can constructs four nonempty sets of arguments $A_{G,i}, A_{C,i}, A'_{C,i}, A_{R,i}$ (it is possible that two sets of the same argument type from different agents intersect) where $A_G = A_{G,1} \cup A_{G,2}$, $A_C = A_{C,1} \cup A_{C,2}$, $A_C = A'_{C,1} \cup A'_{C,2}$, and $A_R = A_{R,1} \cup A_{R,2}$.

If all the arguments only use a finite number of inferences in the language $\mathcal{L}$, then the backbone protocol 4.1 and the deepening search conversation policy 5.2 can collect the set of A_{acc} in the dialogue at some point. After that point, the two agents enter the stable state in which no matter what the two agents say the set of acceptable arguments, and in turn the acceptable plans, will not change. With this kind of domain knowledge, the agent can employ Policy

Policy 5.4 A planning policy

Require: Requirements are those of Policy 5.1 or 5.2

1: Function as Policy 5.1 or 5.2, with the following modified request handler:
2: On request for an argument

- if there is an argument a from $A_{R,i}$ (assume there is an additional mechanism to detect this with errors), return a
- if there is an argument b from $A_{G,i}$ (assume there is an additional mechanism to detect this with errors), return b
- otherwise function as Policy 5.1 or 5.2.

5.4. In the policy, two additional mechanisms are used to generate the arguments from $A_{R,i}$ and $A_{G,i}$ as early as possible so that the system can reach the stable state as soon as possible. As we can see in the policy, we allow errors in the policy for generating these arguments, but it won't greatly affect the outcome of the dialogue for two reasons: 1) the argumentation semantics will still characterize the major part of the acceptable arguments as acceptable on the fly if the errors are restricted to a small range, and 2) the iterative deepening conversation policy will eventually generate these arguments using the inference power of $\mathcal{L}$ although it may take a long time for the dialogue to reach the stable stage.

As we can see, the major efforts in coming up with the dialogue for multiagent planning are: 1) to represent the problem domain in the language $\mathcal{L}$ and the problem solving schemes in terms of the argumentation system chosen, 2) to figure out shortcuts to generate the most important arguments to have the dialogue reach the stable stage as soon as possible. In this way, the dialogue becomes much easier to engineer than the other dialogue approaches for similar problems [6, 14, 27] in which the problem domain, the problem solving schemes, the underlying logic, the dialogue moves, the dialogue protocols, and the dialogue conversation policies need to be considered all together.

6. RESPONSES TO DESIGN DESIDERATA

This section briefly compares our framework against the set of 13 desiderata for argumentation protocols proposed by [20]. 1) In response to stated dialogue purpose, our mechanism does not impose any restriction on the purpose of the dialogue, different applications can choose their purposes freely by agreeing on a set of interest points represented in the language. 2) In response to the need for diversity of individual purposes, different agents can have different points of interest and these will be subjected to the public argumentation to resolve conflicts. 3) In response to the need for inclusiveness, our mechanism allows any agent to participate into the dialogue, and how new agents contribute to the dialogue will depend on the quality of the arguments they can make with respect to the public argumentation semantics. 4) In response to transparency, our mechanism decomposes the functions so that as many components as possible are public. An agents' commitments to the external world should be represented in the language and subjected to public argumentation. 5) In response to fairness, our mechanism advocates fairness in terms of the public argumentation semantics: every agent can influence the outcome if it can provide a good arguments. 6) In response to the clarity of argumentation theory, most of the argumentation theory is captured explicitly by the shared public argumentation semantics. 7) In response to the separation of syntax and semantics, the semantics of the dialogue is mainly defined by the public argumentation semantics. The conversation policies and the backbone protocols are independent of the semantics. 8) In response to rule-consistency, our mechanism in practice allows any conversation policy, but the backbone protocol will rule out invalid arguments and defeats, and the argumentation semantics will further rule out ultimately defeated arguments to maintain the consistency of the public belief set. 9) In response to encouragement of resolution, our mechanism can output results at any time. Whether the status of the public belief benefits a given agent at that time will depend on the quality of the arguments it has put into the dialogue context. This can be viewed as an incentive to encourage agents to provide the best arguments to resolve conflicts. 10) In response to discouragement of disruption, the argumentation semantics prevent behaviors such as repeatedly uttering the same argument from having effect on the public belief. 11) In response to the enablement of self-transformation, there are two aspects. From the view of how an agent influences the public belief, all agents are allowed to change their opinions or preference using different arguments, whether these changes will be sanctioned by the public beliefs will depend on how good an argument the agent can make for its most recent interest. From the view of how an agent changes its views to fit in with the new public belief, we leave this for future work. 12) In response to system simplicity, the decomposition of the dialogue mechanism helps to modularize and thus simplify the major components. 13) In response to computational simplicity, our mechanism allows the private and public conversation polices to be made efficient in order to have public argumentation reach the stable state as early as possible.

7. CONCLUSIONS

In this paper, we propose a flexible dialogue mechanism built on top of a public argumentation system. In this mechanism, we decompose the functions of dialogue into two parts — a backbone protocol which maintains the dialogue context regarding the set of public beliefs, and a set of conversation policies which handle the other aspects of the dialogue regarding the application and further regulation of the public argumentation. In this way, we are free to choose different argumentation theories to maintain the set of public beliefs, we can incrementally construct and combine conversation policies for the computation of argumentation semantics as well as making effective arguments for the specific applications without concerning the other parts of the dialogue. The publicly accessible part of the dialogue, that is the backbone protocol and the public conversation policies, are in general verifiable because they are using only publicly available information. The private part of the dialogue is open for an individual agent to choose with respect to their individual needs. In this way, we balance between the need for public specification and verification and the need for flexibility.

There are a number of ways to extend this work. One future direction is to complete the mechanism with another set of private conversation polices which revise the agent's individual information base in the light of the set of public beliefs. The model used in [4] is a candidate for this set of revision conversation policies, a model that decides how to revise by looking at the status of the arguments in an argumentation system that uses a combination of public and private information. As the maintenance of the public dialogue context is costly, another future direction is to devise distributed algorithms and data structures to efficiently maintain the public beliefs especially in a cooperative society of agents so that we can extend the dialogue mechanism to be a general multiagent coordination mechanism. A third direction that we want to pursue is to formally verify the properties of the backbone protocol and the conversation policies. In addition, more formal treatments of how to combine conversation policies similar to the approach of dialogue game protocols (e.g. those mentioned in [19]) will be

needed, especially to formalize the way in which agents can reach an agreement on the public conversation policy. One candidate is to represent the concern of conversation policy into a multiagent planning problem similar to those defined in section 5.4, then use the dialogue mechanism defined in this paper as a meta-dialogue for the agents to come to an agreement on the conversation polices. In combination with the research directions proposed above, the properties of this mechanism should also be explored in the future.

Acknowledgments

This work was partially supported by the US Army Research Laboratory and the UK Ministry of Defence under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

8. REFERENCES

[1] L. Amgoud. A formal framework for handling conflicting desires. In *Proceedings of the 7th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, 2003.

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

[3] L. Amgoud and C. Cayrol. A reasoning model based on the production of acceptable arguments. *Annals of Mathematics and Artificial Intelligence*, 34(1-3):197–215, 2002.

[4] L. Amgoud, N. Maudet, and S. Parsons. Modeling dialogues using argumentation. In *Proceedings of the Fourth International Conference on Multi-Agent Systems*, 2000.

[5] ASPIC. D1.1 - review on argumentation technology: State of the art, technical and user requirements. Technical report, ASPIC, ASPIC, 2004.

[6] K. Atkinson, T. J. M. Bench-Capon, and P. McBurney. A dialogue game protocol for multi-agent argument over proposals for action. In Rahwan et al. [24], pages 149–161.

[7] P. Besnard and A. Hunter. A logic-based theory of deductive arguments. *Artifical Intelligence*, 128(1-2):203–235, 2001.

[8] B. Bonet and H. Geffner. Arguing for decisions: A qualitative model of decision making. In *Proceedings of the 12th Annual Conference on Uncertainty in Artificial Intelligence*, 1996.

[9] C. Cayrol and M.-C. Lagasquie-Schiex. Graduality in argumentation. In *Journal of Artificial Intelligence Research*, volume 23, pages 245–297. 2005.

[10] C. Chesñevar, A. Maguitman, and R. Loui. Logical models of argument. *ACM Computing Surveys*, 32(4):337–383, 2000.

[11] 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):321–358, 1995.

[12] M. Greaves, H. Holmback, and J. Bradshaw. What is a conversation policy? In *Issues in Agent Communication*, pages 118–131, London, UK, 2000. Springer-Verlag.

[13] J. Y. Halpern. *Reasoning about Uncertainty*. MIT press, Cambridge, MA, 2003.

[14] D. Hitchcock, P. McBurney, and S. Parsons. The eightfold way of deliberation dialogues. *International Journal of Intelligent Systems*, 2004.

[15] H. Jakobovits and D. Vermeir. Robust semantics for argumentation frameworks. *Journal of Logic and Computation*, 9(2):215–261, 1999.

[16] J. Kohlas. Probabilistic argumentation systems: A new way to combine logic with probability. *Journal of Applied Logic*, 1(3-4):225–253, 2003.

[17] P. Krause, S. Ambler, M. Elvang-Gøransson, and J. Fox. A logic of argumentation for reasoning under uncertainty. *Computational Intelligence*, 11:113–131, 1995.

[18] N. Maudet and B. Chaib-Draa. Commitment-based and dialogue-game-based protocols: new trends in agent communication languages. *Knowledge Engineering Review*, 17(2):157–179, 2002.

[19] P. McBurney and S. Parsons. Dialogue game protocols. In M.-P. Huget, editor, *Communication in Multiagent Systems*, volume 2650 of *Lecture Notes in Computer Science*, pages 269–283. Springer, 2003.

[20] P. McBurney, S. Parsons, and M. Wooldridge. Desiderata for agent argumentation protocols. In *Proceedings of the 1st International Conference on Autonomous Agents and Multiagent Systems*, 2002.

[21] D. Nau, M. Ghallab, and P. Traverso. *Automated Planning: Theory & Practice*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[22] J. L. Pollock. Defeasible reasoning with variable degrees of justification. *Artificial Intelligence*, 133(1-2):233–282, 2001.

[23] H. Prakken. Coherence and flexibility in dialogue games for argumentation. *Journal of Logic and Computation*, 15(6):1009–1040, 2005.

[24] I. Rahwan, P. Moraitis, and C. Reed, editors. *Argumentation in Multi-Agent Systems*, volume 3366 of *Lecture Notes in Computer Science*. Springer, 2005.

[25] I. Rahwan, S. D. Ramchurn, N. R. Jennings, P. Mcburney, S. Parsons, and L. Sonenberg. Argumentation-based negotiation. *The Knowledge Engineering Review*, 18(4):343–375, December 2003.

[26] C. Reed and D. Walton. Towards a formal and implemented model of argumentation schemes in agent communication. In Rahwan et al. [24], pages 19–30.

[27] Y. Tang and S. Parsons. Argumentation-based dialogues for deliberation. In *Proceedings of the 4th International Conference on Autonomous Agents and Multiagent Systems*, 2005.

[28] B. Verheij. *Rules, Reasons, Arguments. Formal studies of argumentation and defeat*. PhD thesis, University of Maastricht, 1996.

[29] G. Vreeswijk. The feasibility of defeat in defeasible reasoning. In *Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning*, 1991.

[30] D. N. Walton and E. C. W. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. State University of New York Press, Albany, NY, USA, 1995.

[31] G. Weiss, editor. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.