Agent dialogues with conflicting preferences

Leila Amgoud[1] and Simon Parsons[2]

[1] IRIT-UPS, 118 route de Narbonne 31062
Toulouse Cedex, France.
amgoud@irit.fr
[2] Department of Computer Science, University of Liverpool
Liverpool L69 7ZF, United Kingdom.
s.d.parsons@csc.liv.ac.uk

Abstract. An increasing number of software applications are being conceived, designed, and implemented using the notion of autonomous agents. These agents need to communicate in order to resolve differences of opinion and conflicts of interests, some of which result from differences in preferences. Hence the agents need the ability to engage in dialogues that can handle these communications while taking the differences in preferences into account. In this paper we present a general framework of dialogue where several agents can fulfill the above goals.

1 Introduction

An increasing number of software applications are being conceived, designed, and implemented using the notion of autonomous agents. These applications vary from email filtering, through electronic commerce, to large industrial applications. In all of these disparate cases, however, the notion of autonomy is used to denote the fact that the software has the ability to decide for itself which goals it should adopt and how these goals should be achieved. In most agent applications, the autonomous components need to interact. They need to communicate in order to resolve differences of opinion and conflicts of interest, work together to resolve dilemmas or find proofs, or simply to inform each other of pertinent facts. Often these needs require agents to engage in dialogues.

As a result of this requirement, there has been much work[1] on providing agents with the ability to hold dialogues. Typically these focus on one type of dialogue, either negotiation [14, 18, 19], where agents try to reach agreement on the division of some set of scarce resources, or persuasion [5, 7, 10, 12, 16, 20], where one agent is trying to change the opinion of another. In previous work [2, 3] we have presented a set of illocutions which can capture a number of different types of dialogue, An important limitation of much of this existing work is the fact that it either doesn't take into account the preferences of the agents (eg [10, 12, 16]), or, as in [2, 3], it assumes all agents have the same preferences. Another

[1] This is meant to be representative rather than an exhaustive survey. In particular, it omits the literature on natural language dialogues because we are interested in the much more restricted task of handling simple dialogues between software agents.

limitation is that while the way that the argumentation is handled is considered in great detail, much less attention is often paid to the way that arguments fit into dialogues, the way that these dialogues are constructed, and the overall context in which the dialogue takes place.

In this paper, we take some steps towards tackling these limitations. Firstly, we extend the argumentation system of [2, 3] to give the agents the ability to engage in dialogues while taking different sets of preferences into account. Secondly we show how this extended system fits into the wider context of agent dialogues by presenting a general framework which captures many of the essential features of such dialogues, and provide an instantiation of this framework which brings together elements of our previous work on this subject.

2 Arguing with conflicting preferences

Argumentation is an approach to handle reasoning about inconsistent information, based on the justification of plausible conclusions by arguments. Broadly speaking, any conclusion is initially entertained so long as an argument can be constructed in favour of it. This results in a set of arguments which, in general, conflict because they disagree about the truth of certain propositions. From this set some subset of conclusions are identified as acceptable, based on the relationships between the conflicting arguments.

In previous work [2, 3], we have described in detail a particular formal system for argumentation and shown how it can be used to underpin dialogues between agents under the assumption that all agents have the same set of preferences. Here we extend this system to take account of different sets of preferences, adopting the approach we suggested in [4]. We take the different preferences to be expressed as different preorderings over a set of propositions representing beliefs, and consider that different preferences arise because the propositions are considered from the viewpoint of different contexts. Thus the different preference orderings can be considered as *contextual preferences*, and these can change as agents update their knowledge.

We start with a single agent which has a possibly inconsistent knowledge base Σ with no deductive closure. We assume Σ contains formulae of a propositional language $\mathcal{R}$. $\vdash$ stands for classical inference and $\equiv$ for logical equivalence. The agent's beliefs are context-specific, and we denote this set of contexts by $\mathcal{C} = \{c_1, \ldots, c_n\}$. We assume that there is an order over these contexts, $\sqsupset$, so that for $c_1, c_2 \in \mathcal{C}$, $c_1 \sqsupset c_2$ means that any proposition in context c_1 is preferred over any proposition in context c_2. In addition to the ordering over contexts, there is also a set of preorderings $\gg_i, \ldots, \gg_n$ which give the preference over propositions within every context, $\gg_i$ giving the preferences in context c_i. We have:

Definition 1. An *argument* is a pair $A = (H, h)$ where h is a formula of $\mathcal{R}$ and H a subset of Σ such that:

1. H is consistent;

2. $H \vdash h$; and
3. H is minimal, so no subset of H satisfying both 1. and 2. exists.

H is called the *support* of A, written $H = \text{Support}(A)$, and h is the *conclusion* of A, written $h = \text{Conclusion}(A)$.

Since Σ is inconsistent, $\mathcal{A}(\Sigma)$, the set of all arguments which can be made from Σ, will contain arguments which undercut each other:

Definition 2. Let A_1 and A_2 be two arguments of $\mathcal{A}(\Sigma)$. A_1 *undercuts* A_2 iff $\exists h \in \mathit{Support}(A_2)$ such that $h \equiv \neg \mathit{Conclusion}(A_1)$.

In other words, an argument is undercut iff there exists an argument for the negation of an element of its support.

Each preordering on the set Σ can be used to define a preordering on the set of arguments $\mathcal{A}(\Sigma)$. We can thus define a preference relation Pref_i over a context c_i based on the appropriate preordering $\gg_i$. In [1] several preference relations between arguments were proposed. Some of these assume that each $\gg_i$ is a partial preordering and others take it to be a total ordering. In this paper we adopt a preference relation which takes $\gg_i$ to be a total preordering. This is equivalent to considering the knowledge base to be stratified into non-overlapping sets $\Sigma_1, \ldots, \Sigma_n$ such that facts in Σ_i are all equally preferred and are more preferred than those in Σ_j where $j > i$. The preference level of a nonempty subset H of Σ, $\mathit{level}(H)$, is the number of the highest numbered layer which has a member in H.

Definition 3. Let A_1 and A_2 be two arguments in $\mathcal{A}(\Sigma)$. A_1 is *preferred* to A_2 according to Pref_i iff $\mathit{level}(\mathit{Support}(A_1)) \leq \mathit{level}(\mathit{Support}(A_2))$.

We denote by Pref_1, ..., Pref_n the different preference relations between arguments induced respectively from the preorderings $\gg_1$, ..., $\gg_n$. Note that since the preorderings on Σ may be conflicting, the preorderings on $\mathcal{A}(\Sigma)$ also may be conflicting, so that for two arguments A_1 and A_2, A_1 may be preferred to A_2 in a context c_i and A_2 may be preferred to A_1 in a context c_j such that $i \neq j$. We can now formally define the argumentation system we will use. We will denote such a system by CPAS:

Definition 4. An *argumentation system based on contextual preferences* is a tuple $\langle \mathcal{A}(\Sigma), \mathit{Undercut}, \mathcal{C}, \sqsupseteq, \gg_1, \ldots, \gg_n \rangle$ such that:

- $\mathcal{A}(\Sigma)$ is the set of arguments built from Σ;
- *Undercut* is a binary relation identifying which arguments undercut which other arguments, $\mathit{Undercut} \subseteq \mathcal{A}(\Sigma) \times \mathcal{A}(\Sigma)$;
- $\mathcal{C}$ is a set of contexts $\{c_1, \ldots, c_n\}$;
- $\sqsupseteq$ is a (total or partial) preordering on $\mathcal{C} \times \mathcal{C}$; and
- $\gg_i$ is a (partial or total) preordering on $\Sigma \times \Sigma$ in the context c_i.

The preference orderings $Pref_1, \ldots, Pref_n$ make it possible to distinguish different types of relations between arguments based on the way in which a set of arguments mutually undercut one another. Broadly speaking, an argument defends itself if it is stronger (in the sense of being based in a preferred context) than those arguments which seek to undercut it, and a set of arguments can defend a lone argument by undercutting all those arguments which the lone argument cannot defend itself against:

Definition 5. Let A_1, A_2 be two arguments of $\mathcal{A}(\Sigma)$.

- If A_2 undercuts A_1 then A_1 *defends itself* against A_2 iff:
  1. $\exists c_i \in \mathcal{C}$ such that A_1 $Pref_i$ A_2 and
  2. $\forall c_j$ such that A_2 $Pref_j$ A_1 then $c_i \sqsupset c_j$.

  Otherwise, A_1 *does not defend itself.*
- A set of arguments $\mathcal{S}$ *defends* A iff $\forall B$ such that B undercuts A and A does not defend itself against B then $\exists C \in \mathcal{S}$ such that C undercuts B and B does not defend itself against C.

Henceforth, $C_{Undercut,\sqsupset}$ will refer to all non-undercut arguments and arguments defending themselves against all their undercutting arguments. In [4], it was shown that the set $\underline{\mathcal{S}}$ of acceptable arguments of the argumentation system $\langle \mathcal{A}(\Sigma), Undercut, \mathcal{C}, \sqsupset, \gg_1, \ldots, \gg_n \rangle$ is the least fix-point of a function $\mathcal{F}$:

$$\mathcal{S} \subseteq \mathcal{A}(\Sigma)$$
$$\mathcal{F}(\mathcal{S}) = \{A \in \mathcal{A}(\Sigma) | \, A \text{ is defended by } \mathcal{S}\}$$

which then leads us to be able to define those arguments which are acceptable in the sense of either defending themselves or being defended:

Definition 6. The set of *acceptable* arguments for an argumentation system $\langle \mathcal{A}(\Sigma), Undercut, \mathcal{C}, \sqsupset, \gg_1, \ldots, \gg_n \rangle$ is:

$$\underline{\mathcal{S}} = \bigcup \mathcal{F}_{i \geq 0}(\emptyset) = C_{Undercut,\sqsupset} \cup \left[\bigcup \mathcal{F}_{i \geq 1}(C_{Undercut,\sqsupset}) \right]$$

An argument is acceptable if it is a member of the acceptable set.

In practice we don't need to calculate all the acceptable arguments from Σ in order to know whether or not a given argument is acceptable [2].

We can use this argumentation system to straightforwardly extend our previous work on argumentation-based dialogues [2, 3] to deal with agents that have different preferences.

3 A dialogue system

In this section we show how the system of argumentation introduced above may be used in inter-agent dialogues. We start from an abstract notion of a dialogue system which captures the basic elements required for such dialogues.

3.1 A general framework

One basic element in a dialogue system is the set of agents involved in a dialogue. Dialogues are often considered to take place between two agents, but here we deal with dialogues between arbitrary numbers of agents. Each agent has a name, a role in the dialogue and a knowledge base. The knowledge-base may contain information about other agents and we assume it is equipped with a (partial or total) preordering representing the preferences of the agent. The knowledge-base is written in some logic, and different agents can use different logics.

Following Hamblin [9] and Mackenzie [10], we suppose that the dialogue takes place using *commitment stores* that hold all the statements to which an agent has committed during the dialogue (broadly speaking all the propositions it has agreed are true). The different commitment stores are empty at the beginning of a dialogue and the rules for carrying out dialogues define how the commitment stores are updated. Thus the union of all the commitment stores can be viewed as the state of the dialogue at a particular time. Furthermore, the commitment stores provide additional knowledge, over and above that in an agent's knowledge-base, which an agent can use as the basis for its utterances.

We bring these ideas together in the notion of a dialogical agent:

Definition 7. A *dialogical agent* is a tuple $\langle A_i, Role_{A_i}, \mathcal{L}_{A_i}, \Sigma_{A_i}, \gg_{A_i}, CS_{A_i} \rangle$ where:

- A_i is the name of the agent;
- $Role_{A_i}$ denotes the role of agent A_i;
- $\mathcal{L}_{A_i}$ denotes the logic used by agent A_i;
- Σ_{A_i} is the knowledge base available to the agent A_i;
- $\gg_{A_i}$ represents the preference order over Σ_{A_i}; and
- CS_{A_i} stands for the commitment store of agent A_i.

We denote a single dialogical agent by $\mathcal{A}_i$ and define a *set of dialogical agents* as $\mathsf{A} = \bigcup_i \{\mathcal{A}_i\}$.

The role of an agent may affect the kind of logic used by that agent. In dialogues where a judge determines the outcome, some agents may use classical logic whereas the judge requires a more sophisticated non-classical logic in order to handle the contradictory arguments presented. In some applications, the strength of arguments are determined by roles. As discussed in [17], the hierarchy of a company may be reflected in the weight accorded to arguments made by agents playing certain roles. Hence, the set of agents may be equipped with a (partial or total) preordering $\succ$ to capture these differences in the strength of argument that an agent derives from its role.

Now, a dialogue may be viewed as a sequence of speech acts made by agents:

Definition 8.

A *move* in a dialogue is a tuple: $M = \langle S, H, Act \rangle$. S is the agent providing the act, written $S = Speaker(M)$. H is the agent or set of agents to whom the act is addressed, written $H = Hearer(M)$. When the act is addressed to all the agents, we denote A_Ω. $Act = Act(M)$ is the act itself.

A *dialogue* is a non-empty sequence of moves $M_1, \ldots, M_p$ such that:

- $Speaker(M_j) \neq Hearer(M_j)$
- $CS_{A_i}(0) = \emptyset, \forall i = 1, \ldots n$. Note that $CS_{A_i}(0)$ is the commitment store of agent A_i at step 0.
- For any two moves M_j, M_k, if $j \neq k$ then $Act(M_j) \neq Act(M_k)$
- For any $j < p$: $Speaker(M_j) \neq Speaker(M_{j+1})$.

The first condition prevents an agent from addressing a move to itself. The second says that commitment stores are empty at the beginning of the dialogue. The third prevents an agent from repeating an act already provided by another agent or by the agent itself (in other words, for example, it prevents the an agent repeatedly asking the same question). This guarantees non circular dialogues. The last condition prevents an agent from providing several moves at the same time and guarantees that there are no monologues.

Note the formal distinction between dialogue moves and dialogue acts. An act is a locution ($assert(p)$, $challenge(q)$ and so on), and makes up part of a move along with the agent generating the act and the agent receiving the act. When there is no chance of confusing the terms (or, as in most situations, it is possible to correctly use either), we will use "act" and "move" interchangeably and we frequently use M to denote a set of acts.

Agents are not free to make any move at any time—their moves are governed by a protocol, a set of rules governing the high-level behavior of interacting agents. A given protocol specifies the kinds of moves or acts the agents can make (assertions, requests, and so on) at any point in the dialogue.

Definition 9. A *dialogue protocol* is defined as a function $\pi : \mathsf{M} \mapsto 2^{\mathsf{M}}$, where M is a set of dialogue acts.

A dialogue protocol specifies the rules for interactions between agents and the different replies that are possible after a given move. In general there will be several such moves possible, and the exact way that an agent responds is the result of the agent's strategy. We can therefore think of a given strategy for an agent as being a function from the set of moves identified by the protocol to a single move which is then uttered by an agent:

Definition 10. A *dialogue strategy* is defined as a function $\mathcal{S} : 2^{\mathsf{M}} \mapsto \mathsf{M}$, where M is a set of dialogue acts, such that for $T \subseteq \mathsf{M}$, $\mathcal{S}(T) \in T$.

Given these definitions, we can define a dialogue system. Such a system will consist of a set of dialogical agents with an ordering over them, a protocol (which includes a set of speech acts), a set of strategies (one strategy for each agent), and a dialogue (which can be thought of as a tape recording of everything that has been said in the dialogue). Formally:

Definition 11. A *dialogue system* is a tuple $\langle \mathsf{A}, \succ, \mathsf{M}, \pi, \mathsf{S}, \mathcal{D} \rangle$ where A is a set of dialogical agents, $\succ$ is a preordering over the elements of A, M is a set of acts, π is a dialogue protocol, S is a set of strategies, and $\mathcal{D}$ is a dialogue.

In the rest of this section we will illustrate the idea of a dialogue system by instantiating each of these elements with a specific example.

3.2 The set of agents

We consider a set of agents $\mathsf{A} = \{A_1, \ldots, A_n\}$, $n \geq 2$, where A_i is the name of the ith agent. For now we make no explicit use of the agent roles, noting only that, as in [17], roles can play a part in the formation of the preference order $\succ$ over the agents. Note that we only consider dialogues between at least two agents. As in [14], we assume that each agent has a set of beliefs, B, a set of desires, D, and a set of intentions, I. However, for the moment we deal with propositional knowledge and take each agent to use only classical propositional logic, modelling beliefs, desires and intentions by partitioning the knowledge base of each agent appropriately. In particular, we take the *basic knowledge base* of an agent A_i to be:

$$\Sigma^B_{A_i} = B_{A_i} \cup D_{A_i} \cup I_{A_i}$$

and assume that we know that a particular proposition p is, for example, one of A_i's intentions because it resides in I_{A_i}, not because it is explicitly denoted as such. We work under this restriction for notational simplicity, bearing in mind that the problem of how to deal with first-order argumentation in which beliefs, desires and intentions are explicitly denoted is considered in depth in [14].

In addition, the propositional language can usefully be extended to represent the type of information exchanged between agents in negotiation. As discussed in [17], negotiations often involve trade-offs with one agent accepting a request from a second agent provided that the second accepts its request. For example: "If you let me use your laptop, I'll let use my printer". To make it easier to represent this kind of information a new connective $\Rightarrow$ was introduced in [3]. Thus we have a new language $\mathcal{L}$ which contains propositional formulae and formulae $p \Rightarrow q$ such that p and q are propositional formulae. Note that for this instantiation of our general framework, all agents are assumed to have the same logic. As real multi-agent systems employ different logics, in [11] a formalism allowing reasoning with different rules of inference has been defined.

As described above, each agent A_i uses a commitment store CS_{A_i} in order to keep track of its dialogue commitments. At the beginning of the dialogue the different commitment stores are empty, and agents are assumed to have free access to any information stored in them. Since agents have to exchange two kinds of information—knowledge and preferences—the commitment stores will have two parts. The first part, denoted by $CS.Pref_{A_i}$, will contain preferences, and the second part, denoted by $CS.Kb_{A_i}$, will contain knowledge:

$$CS_{A_i} = CS.Pref_{A_i} \cup CS.Kb_{A_i}$$

Agent A_i knows everything in the commitment stores of all the agents in the dialogue, $\cup^n_{j=1} CS.Kb_{A_j}$, and also has some information about the beliefs of each agent, $\cup_{j \neq i} \Sigma^{B'}_{A_j}$ with $\Sigma^{B'}_{A_j} \subseteq \Sigma^B_{A_j}$. Thus the overall knowledge available to A_i is:

$$\Sigma_{A_i} = \Sigma^B_{A_i} \cup [\cup_{j \neq i} \Sigma^{B'}_{A_j}] \cup [\cup^n_{j=1} CS.Kb_{A_j}]$$

Each agent also has preferences over its knowledge base, and so is equipped with a (total or partial) preordering on $\Sigma^B_{A_i}$, denoted by $\gg^B_{A_i}$. These preferences are given as pairs (a, b) which denotes a is preferred to b.

A_i also knows some of the preferences of other agents A_j. These preferences, a subset of those in $\gg^B_{A_j}$, will be denoted by $\gg^{B'}_{A_j}$. A_i also knows the preferences that the other agents have stated, and are thus in commitment stores. Hence it knows $\gg_{j=1,n}\ CS.Pref_{A_j}$. So the overall set of preferences that A_i knows about are:

$$\gg_{A_i} = \gg^B_{A_i} \cup [\cup_{j \neq i} \gg^{B'}_{A_j}] \cup [\cup_{j=1,n} CS.Pref_{A_j}]$$

Each agent is equipped with an argumentation framework of the kind discussed above as its inference mechanism[2]. Using Σ_{A_i}, A_i can build arguments concerning beliefs, desires and intentions as discussed above. We adopt the system from Section 2 treating agents as contexts, so the set of contexts $\mathcal{C}$ is replaced by the set of agents A and the preordering $\sqsupset$ on the contexts replaced by the preordering $\succ$ on the set of agents. Thus each agent A_i will use the argumentation system:

$$\langle \mathcal{A}(\Sigma_{A_i}), \mathit{Undercut}, \mathsf{A}, \succ, (\gg^{B'}_{A_1} \cup CS.Pref_{A_1}), \ldots, (\gg^{B'}_{A_n} \cup CS.Pref_{A_n}) \rangle$$

The argumentation frameworks are used to help the agents to maintain the coherence of their beliefs, and ensure that they only assert justified arguments. In this sense the argumentation systems help to operationalize the *rationality* of the agents, and, as in [2], this in turn ensures that if dialogues end, they end with agents agreeing on arguments which are acceptable to all agents in the dialogue.

3.3 Dialogue acts and protocol

A set of dialogue acts and a protocol together these define the full set of legal moves available to an agent at any given time, and the mapping from one move in dialogue to the set of possible next moves. In this paper we combine these, as in [3], by giving each act an associated set of rules for using it—update rules, dialogue rules and rationality rules. The update rules simply say how the commitment stores of all the agents are updated as a result of the act. The rationality rules and the dialogue rules are more complex. The rationality rules are preconditions for an act, for instance saying that the act can only be made if the agent can build a certain argument. The dialogue rules define the acts which can be used to respond to a given act. Thus the rationality and dialogue rules together identify the set of possible next moves, and hence make up a protocol. Given a particular move, the dialogue rules for that move identify a set of possible replies and the rationality rules for these possible replies then identify which can be legally used.

In the following descriptions, we suppose that agent A_i addresses an act to the other agents. The CPAS is therefore:

$$\langle \mathcal{A}(\Sigma_{A_i}), \mathit{Undercut}, \mathsf{A}, \succ, (\gg^{B'}_{A_1} \cup CS.Pref_{A_1}), \ldots, (\gg^{B'}_{A_n} \cup CS.Pref_{A_n}) \rangle.$$

[2] We view the meta-level inference provided by the argumentation system, along with the underlying propositional logic to be the $\mathcal{L}_{A_i}$ of an agent.

Basic dialogue acts

assert(p) where p is any formula in $\mathcal{L}$. This allows the exchange of information, such as "the weather is beautiful" or "It is my intention to hang a picture".

rationality The agent uses the CPAS to check if there is an acceptable argument A such that $p = Conclusion(A)$.
dialogue The other agents can respond with:
1. *accept*(p),
2. *assert*($\neg p$),
3. *challenge*(p).

update $CS.Kb_{A_i}(t) = CS.Kb_{A_i}(t-1) \cup \{p\}$ and $CS.Kb_{A_j}(t) = CS.Kb_{A_j}(t-1), \forall j \neq i$.

This assertion is added to the CS of the agent making the assertion. Note that an agent A_j can only make a response if the rationality rule for that response is satisfied. Thus it can only respond to *assert*(p) with *assert*($\neg p$) if it has an acceptable argument for $\neg p$. *assert*(S) where S is a set of formulae in $\mathcal{L}$ representing the support of an argument, is handled in a similar manner [3].

An agent is also allowed to present its preferences, requiring a new locution in addition to those in [3]:

prefer($(a_1, b_1), \ldots, (a_n, b_n)$) where a_i, b_i are formulae in $\mathcal{L}$.

rationality There is no rationality condition.
dialogue The other agents can play:
1. *prefer*($(a'_1, b'_1), \ldots, (a'_j, b'_j)$),
2. *assert*(S),
3. *question*(q),
4. *request*(q), where $q = b_i$,
5. *promise*($x \Rightarrow a_i$).

update $CS.Pref_{A_i}(t) = CS.Pref_{A_i}(t-1) \cup \{(a_1, b_1), \ldots, (a_j, b_j)\}$ and $CS.Pref_{A_j}(t) = CS.Pref_{A_j}(t-1), \forall j \neq i$.

Informally, this means that the responding agent can present its preferences, give an argument, ask a question, request something not preferred by the original agent, or simply promise something preferred by the other agent in exchange for another element. The next two moves allow an agent to ask questions.

challenge(p) where p is a formula in $\mathcal{L}$.

rationality There is no rationality condition.
dialogue The other agents can only *assert*(S) where S is the support of the argument (S, p), or S is the support of the argument (S, h) such that p belongs to S and h is one of A_i's intentions.
update $CS.Kb_{A_i}(t) = CS.Kb_{A_i}(t-1)$ and $CS.Kb_{A_j}(t) = CS.Kb_{A_j}(t-1), \forall j \neq i$.

question(p) [3] allows A_i to ask if p is the case. The other agents can answer either affirmatively (if they can show it to be the case) or negatively, if they can show it is not the case, or by asking another question, or by making a request.

Negotiation acts

The following acts are negotiation specific—while not strictly necessary for negotiation, they make it easier to capture some of the statements we wish our agents to make.

request(p) where p is any formula in $\mathcal{L}$.

rationality A_i uses the CPAS to identify a p in some $\Sigma_{A_j}^{B'}$ such that $p \in H$ and (H, h) is an acceptable argument for one of A_i's intentions.

dialogue The agent A_j can:

1. *accept*(p),
2. *refuse*(p),
3. *challenge*(p),
4. *promise*$(q \Rightarrow p)$.

update $CS.Kb_{A_i}(t) = CS.Kb_{A_i}(t-1)$ and $CS.Kb_{A_j}(t) = CS.Kb_{A_j}(t-1) \cup \{p\}$.

A request is stored in the CS of the receiving agent because, if accepted, it becomes a commitment on that agent. A *request* locution is invoked when an agent cannot, or prefers not to, achieve its intentions alone. The proposition requested differs from an asserted proposition in that it cannot be proved true or false—the decision on whether to accept it or not hinges upon the relation it has to other agents' intentions (see below).

promise$(p \Rightarrow q)$ where p and q are formulae in $\mathcal{L}$.

rationality A_i uses the CPAS to identify a p in some $\Sigma_{A_j}^{B'}$ such that $p \in H$ and (H, h) is an acceptable argument for one of A_i's intentions, and to check that there is no acceptable argument (H', h') for one of its intentions h' such that $q \in H'$.

dialogue The agent A_j can:

1. *accept*$(p \Rightarrow q)$,
2. *refuse*$(p \Rightarrow q)$,
3. *promise*$(s \Rightarrow p)$,
4. *challenge*(p),
5. *prefer*$((x, q))$.

update $CS.Kb_{A_i}(t) = CS.Kb_{A_i}(t-1) \cup \{q\}$ and $CS.Kb_{A_j}(t) = CS.Kb_{A_j}(t-1) \cup \{p\}$.

Broadly speaking, an agent will make a promise when it needs to request something, p, from another, and has something it does not need (because the thing is not needed to achieve any intentions), q, which it can offer in return. In replying to a promise, an agent can accept, refuse, question why the requested thing is required, or suggest an alternative trade (A_j replying with $s \Rightarrow p$ is equivalent to A_i retracting its initial promise and replacing it with $p \Rightarrow s$).

Responding acts

The following are acts which are made in response to requests and assertions.

accept(p) where p is a formula in $\mathcal{L}$. After an assertion or request, an agent can respond with an explicit acceptance.

rationality In response to an assertion, A_i uses its CPAS to check if there is an acceptable argument for p. If so, the move can be played. In response to a request, A_i has to check that there is no acceptable argument (H, h) for one of its intentions h, such that $p \in H$. In other words, it is only possible to accept a request if it doesn't invalidate the supporting argument for one of its intentions[3].
dialogue The other agents can make any move except *refuse*.
update $CS.Kb_{A_i}(t) = CS.Kb_{A_i}(t-1) \cup \{p\}$ and $CS.Kb_{A_j}(t) = CS.Kb_{A_j}(t-1), \forall j \neq i$.

Note that in case of a response to a request, p is already in the commitment store of the agent. An agent can also accept a set of formulae S, *accept*(S), dealing with each member s of S as for *accept*(s). An agent can also accept a promise, using *accept*($p \Rightarrow q$) [3].

refuse(p) where p is any formula in $\mathcal{L}$.

rationality A_i uses the CPAS to check if there is an acceptable argument (H, h) for one of its intentions h such that $p \in H$.
dialogue The other agents can make any move except *refuse*.
update $CS.Kb_{A_i}(t) = CS.Kb_{A_i}(t-1) \backslash \{p\}$ and $CS.Kb_{A_j}(t) = CS.Kb_{A_j}(t-1), \forall j \neq i$.

Thus A_i will refuse requests which are necessary to achieve its intentions. There is also a *refuse* for promises [3] which reverses the effect of the previous locution on the commitment stores. As discussed in [2, 3], these acts are sufficient to capture many types of dialogue, making the system more general than others which concentrate on just persuasion or negotiation. In addition, the acts closely relate the moves to the argumentation performed by an agent and hence, through the rationality conditions, to the information it has available to it.

This complete set of dialogue acts we denote $\mathcal{M}'_p$ after those developed in [2, 3], both of which are subsets of $\mathcal{M}'_p$. $\mathcal{M}_{\mathcal{DC}}$ from [2] contains *assert*, *accept*, *challenge* and *question*, and $\mathcal{M}'$ from [3] contains all the acts from $\mathcal{M}_{\mathcal{DC}}$ along with *request*, *promise* and *refuse*.

3.4 Dialogue strategy

Once the protocol has determined the set of legal moves available to an agent, the strategy selects one move from that set. For the moves we have here, we can identify a number of strategies which reflect broad classes of agent types:

[3] As in [14], an argument for an intention is essentially a plan for achieving it, so allowing p would invalidate this plan.

- Agreeable: *accept* whenever possible.
- Disagreeable: only *accept* when no reason not to.
- Open-minded: only *challenge* when necessary.
- Argumentative: *challenge* whenever possible.
- Elephant's Child [8]: *question* whenever possible.

Although we have termed these "strategies" each is only a partial definition of $\mathcal{S}$—a full definition would have to take into account the nature of the previous move and hence the overall protocol for the dialogue. For example, the agreeable strategy defines how to respond to $assert(p)$ when the agent does not have an argument against p. If the agent does have an argument, then a complete strategy would have to choose between $assert(\neg p)$ and $challenge(p)$. More work is required to define such complete strategies, and to work out the interaction between the strategies and rationality rules and their impact, and one approach to this is explored in [15].

The choice of strategy can have a big impact on the properties of the dialogue, especially on those related to length and termination. For example, as discussed below agents using the Elephant's Child strategy can cause a dialogue to spiral into an endless round of *question*s, while an argumentative agent has the potential to prolong any dialogue by constantly requiring other agents to assert arguments unnecessarily. Some initial work on this question is reported in [15] showing that termination is guaranteed in some kinds of dialogue using a subset of the dialogue moves discussed here.

3.5 Properties

Consider dialogues which start with one agent asserting some proposition (a typical start for dialogues which involve one agent trying to persuade another of the truth of a proposition). For such *assertion-led* dialogues we have [2][4] the following:

Theorem 12. *Given two agents P and C, equipped with argumentation systems AS_P and AS_C respectively, which hold an assertion-led dialogue using the set of dialogue acts $\mathcal{M}_{\mathcal{DC}}$ in which P moves first, then if S is the set of all arguments which the game can possibly generate,*

- *$\forall x \in S$, x is in the set of acceptable arguments of either AS_P or AS_C;*
- *If $x \in S$ is in the set of acceptable arguments of AS_P, it is in the set of acceptable arguments of AS_C.*

This theorem follows directly from the definition of the rationality conditions of the set of dialogue acts. As a result, it extends directly to $\mathcal{M}'$and $\mathcal{M}'_p$:

[4] In [2], the wording of the theorem is in terms of "justified arguments" which under the proof theory we use for constructing arguments (see [2] for details) are just the same as acceptable arguments, and the dialogues, though assertion-led, are not explicitly stated as such.

Theorem 13. *Given two agents P and C, equipped with argumentation systems AS_P and AS_C respectively, which hold an assertion-led dialogue using the set of dialogue acts $\mathcal{M}'$ or $\mathcal{M}'_p$ in which P moves first, then if S is the set of all arguments which the game can possibly generate,*

- *$\forall x \in S$, x is in the set of acceptable arguments of either AS_P or AS_C;*
- *If $x \in S$ is in the set of acceptable arguments of AS_P, it is in the set of acceptable arguments of AS_C.*

These results give us some guarantees about the soundness of the dialogues (they only involve arguments that at least one agent finds acceptable) and about the way the dialogue can terminate (if the dialogue ends with an argument being acceptable to P, then it is also acceptable to C, so that P can be considered to have "persuaded" C). They can readily be extended to multi-party dialogues.

We can also give results relating to termination (supplementing those in [15]). For instance:

Theorem 14. *Given two agents P and C that both use the Elephant's Child strategy, any dialogue between the two agents using the set of dialogue acts $\mathcal{M}_{\mathcal{DC}}$, $\mathcal{M}'$ or $\mathcal{M}'_p$ will not terminate.*

Proof. Let us assume that P starts the dialogue. By definition P will *question* if possible, and since there are no dialogue conditions in force at the start of a dialogue will do so. The protocol allows C to respond with another question, and so, as an Elephant's Child, it will. By the same reasoning, P will then respond with a question, as will C, and so on. Even with a finite set of propositions available to them, each agent can generate an infinite number of questions, and the dialogue will never terminate. □

Thus, as we might expect, agents which follow the Elephant's Child strategy can disrupt dialogues, but the following result seems to suggest that it takes two such agents to cause non-termination:

Theorem 15. *Given two agents P, which uses the Elephant's Child strategy, and C, any dialogue using the set of dialogue acts $\mathcal{M}_{\mathcal{DC}}$ can be made to terminate if C chooses appropriate acts.*

Proof. Consider that P moves first, as above issuing a *question*. For P to be able to issue another *question*, and so keep the dialogue running, C must issue a *question* or an *accept*. For this proof it is sufficient to consider ways in which C can prevent this happening. Initially, therefore C will respond to the question with an *assert* rather than a *question*. Considering the dialogue conditions on *assert*, and those on every legal act that may follow *assert*, there is no way that C can follow its assertion with an *accept* and thus no way that P can force itself into a position in which it can *question*. Similarly, if C starts the dialogue, provided it does not *question* or *challenge*, there is no sequence of moves P can make in response to C's initial move (which with $\mathcal{M}_{\mathcal{DC}}$ has to be an *assert*) which will mean that P is in a position to ask even one question. □

It is currently an open question whether this latter result translates to $\mathcal{M}'$ and $\mathcal{M}'_p$.

	Σ_P	Σ_C
1	$\neg agr, pri, min$	$min, min \rightarrow \neg pri$
2	$pri \wedge \neg agr \rightarrow \neg pub$	pri
3	$min \rightarrow \neg pri$	

Table 1. The knowledge bases for the example

Move	CS
⟨P, C, $assert(\neg pub)$⟩	$CS.Kb_P = \{\neg pub\}$, $CS.Pref_P = \emptyset$ $CS.Kb_C = \emptyset$, $CS.Pref_C = \emptyset$
⟨C, P, $challenge(\neg pub)$⟩	$CS.Kb_P = \{\neg pub\}$, $CS.Kb_C = \emptyset$
⟨P, C, $assert(\{pri, \neg agr, pri \wedge \neg agr \rightarrow \neg pub\})$⟩	$CS.Kb_P = \{\neg pub, pri, \neg agr, pri \wedge \neg agr \rightarrow \neg pub\}$ $CS.Kb_C = \emptyset$
⟨C, P, $assert(\{min, min \rightarrow \neg pri\})$⟩	$CS.Kb_P = \{\neg pub, pri, \neg agr, pri \wedge \neg agr \rightarrow \neg pub\}$ $CS.Kb_C = \{min, min \rightarrow \neg pri\}$
⟨P, C, $prefer((pri, min \rightarrow \neg pri))$⟩	$CS.Pref_P = \{(pri, min \rightarrow \neg pri)\}$ $CS.Pref_C = \emptyset$
⟨C, P, $prefer((min \rightarrow \neg pri, pri))$⟩	$CS.Kb_P = \{\neg pub,$ pri, $\neg agr, pri \wedge \neg agr \rightarrow \neg pub\}$ $CS.Pref_P = \{(pri, min \rightarrow \neg pri)\}$ $CS.Kb_C = \{min, min \rightarrow \neg pri\}$ $CS.Pref_C = \{(min \rightarrow \neg pri, min)\}$

Table 2. The way the commitment stores change during the privacy dialogue

4 An example

In this section we present an example of a *persuasion dialogue* between two software agents P and C both of which use the instantiation of our general framework which was presented in the previous section. In this dialogue C is regarded as knowing more about the subject than P and so C $\succ$ P.

P: Newspapers can't publish the information X.
C: Why?
P: Because it's about A's private life and A doesn't agree.
C: But A is a minister, and any information concerning a minister is public.
P: I know, but that is less important than the private life of a person.
C: No, in politics the occupation is more important than anything else.

To handle this dialogue formally, we give the agents the knowledge-bases in Table 1 where *agr* denotes "A agrees", *pri* denotes "private", *min* denotes "A is a minister", and *pub* denotes "Newspapers can publish X").

This formal dialogue proceeds as in Table 2. The dialogue begins when P

presents an argument, $A = (\{pri, \neg agr, pri \wedge \neg agr \rightarrow \neg pub\}, \neg pub)$, in favour of not publishing. That argument is initially acceptable in P's argumentation framework since it defends itself against the unique undercutting argument, $B = (\{min, min \rightarrow \neg pri\}, \neg pri)$. Later C gives its preferences and these conflict with P's. Since C is regarded as more reliable than P in this domain, P's argumentation framework will find that the argument A is not longer acceptable since in the most preferred context (that of agent C), B is preferred to A.

5 Conclusion

This paper makes two main contributions to the study of inter-agent dialogues. First, it presents a way of handling dialogues between agents with different preferences, which relaxes one of the more restrictive constraints imposed by much previous work in this area ([5] being an honourable exception). Second, it presents both a general framework which identifies and defines some of the important components in any agent dialogue, and a detailed description of an instantiation of this framework. This work is also novel, going further than previous attempts (including the work which underpins much of this paper [2, 3]) in identifying all the parts of the computational framework necessary to generate dialogues between agents. We note that it leaves unresolved the way in which the illocutions we use relate to agent communication languages, and this something we aim to clarify in the future.

In some respects the work presented here follows on from the dialogical framework of Noriega and Sierra [13] and the framework for argumentation-based negotiation of Sierra *et al.* [17]. This paper starts at the same high level of abstraction, but goes into much greater detail about the precise way in which the agents carry out the argumentation which underpins the dialogue. This is not to say that the current work subsumes the ideas of dialogical frameworks and, especially, the related notion of electronic institutions [6]. Indeed, relating what we have here to the notion of institutions is the subject of ongoing work.

Acknowledgments This work was funded by the EU SLIE Project (IST-1999-10948). Many thanks to Peter McBurney for helpful comments.

References

1. L. Amgoud, C. Cayrol, and D. LeBerre. Comparing arguments using preference orderings for argument-based reasoning. In *Proceedings of the 8th International Conference on Tools with Artificial Intelligence*, pages 400–403, 1996.
2. L. Amgoud, N. Maudet, and S. Parsons. Modelling dialogues using argumentation. In *Proceedings of the International Conference on Multi-Agent Systems*, pages 31–38, Boston, MA, 2000.
3. L. Amgoud, S. Parsons, and N. Maudet. Arguments, dialogue, and negotiation. In *Proceedings of the Fourteenth European Conference on Artificial Intelligence*, pages 338–342, Berlin, Germany, 2000.

4. L. Amgoud, S. Parsons, and L. Perrussel. An argumentation framework based on contextual preferences. In *Proceedings of the International Conference on Formal and Applied and Practical Reasoning*, pages 59–67, 2000.
5. G. Brewka. Dynamic argument systems: a formal model of argumentation process based on situation calculus. *Journal of Logic and Computation*, 11(2):257–282, 2001.
6. M. Esteva, J. A. Rodríguez-Augilar, J. L. Arcos, C. Sierra, and P. Garcia. Institutionalising open multi-agent systems. In *Proceedings of the 4th International Conference on Multi Agent Systems*, pages 381–382, 2000.
7. T. F. Gordon. The pleadings game. *Artificial Intelligence and Law*, 2:239–292, 1993.
8. R. Kipling. *Just So Stories.* Everyman Library, 1992.
9. C. L.Hamblin. *Fallacies.* Methuen, London, 1970.
10. J. MacKenzie. Question-begging in non-cumulative systems. *Journal of Philosophical Logic*, 8:117–133, 1979.
11. P. McBurney and S.Parsons. Tenacious tortoises: A formalism for argument over rules of inference. In *Worshop of Computational Dialectics: Models of argumentation, Negotiation, and Decision Making. 14th European Conference on Artificial Intelligence*, 2000.
12. R. McConachy and I. Zukerman. Dialogue requirements for argumentation systems. In *Proceedings of IJCAI'99 Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, 1999.
13. P. Noriega and C. Sierra. Towards layered dialogical agents. In J. P. Muller, M. J. Wooldridge, and N. R. Jennings, editors, *Intelligent Agents III*, pages 173–188, Berlin, Germany, 1997. Springer Verlag.
14. S. Parsons, C. Sierra, and N. R. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261—292, 1998.
15. S. Parsons, M. Wooldridge, and L. Amgoud. An analysis of formal inter-agent dialogues. Technical report, Department of Computer Science, University of Liverpool, 2001.
16. H. Prakken. On dialogue systems with speech acts, arguments, and counterarguments. In *7th European Workshop on Logic for Artificial Intelligence*, Malaga, 2000.
17. C. Sierra, N. R. Jennings, P. Noriega, and S. Parsons. A framework for argumentation-based negotiation. In *Proceedings of the 4th International Workshop on Agent Theories, Architectures and Languages*, 1997.
18. K. Sycara. Persuasive argumentation in negotiation. *Theory and Decision*, 28:203–242, 1990.
19. F. Tohmé. Negotiation and defeasible reasons for choice. In *Proceedings of the Stanford Spring Symposium on Qualitative Preferences in Deliberation and Practical Reasoning*, pages 95–102, 1997.
20. S. Zabala, I. Lara, and H. Geffner. Beliefs, reasons and moves in a model for argumentation dialogues. In *Proceedings of the Latino-American Conference on Computer Science*, 1999.