A new semantics for ACL based on commitments and penalties

Leila Amgoud, Florence Dupin de Saint-Cyr

A new semantics for ACL based on commitments and penalties*

Leila Amgoud Florence Dupin de Saint-Cyr

Institut de Recherche en Informatique de Toulouse
Université Paul Sabatier, 118 route de Narbonne
31062 Toulouse Cedex 9, France
{amgoud, bannay}@irit.fr

Abstract. In complex multi agent systems, the agents may be heterogeneous and possibly designed by different programmers. Thus, the importance of defining a standard framework for agent communication languages (ACL) with a *clear semantics* has been widely recognized. The semantics should be *verifiable*, *clear* and *practical*. Most classical proposals (for instance, *mentalistic semantics*) fail to meet these objectives.
This paper proposes a *logic-based* semantics which is *social* in nature. The basic idea is to associate with each speech act a clear meaning in terms of a *commitment* induced by that speech act, and a *penalty* to be paid in case that commitment is violated. A *violation criterion* based on the existence of arguments is then defined per speech act. We show that the proposed semantics satisfies some key properties that ensure that the approach is well-founded. The logical setting makes the semantics verifiable. Moreover, it is shown that the new semantics is practical since it captures the dynamic of dialogues, and shows clearly how isolated speech acts can be connected for building dialogues.

Keywords: Agent communication languages, Commitments, Penalties

1 Introduction

When building multi-agent systems, we take for granted the fact that the agents which make up the system will need to communicate and to engage in the different types of dialogues identified by Walton and Krabbe in [33], using a communication language (ACL).
The definition of an ACL from a syntactic point of view (the different *speech acts*[1] that agents can perform during a dialogue) poses no problems. The situation is different when semantics is taken into account. Given that agents in a multi-agent system may be independently designed by different programmers,

* This is a draft version, the article was published In International Journal of Intelligent Systems, Wiley periodical Inc., Vol. 23(3), p286-312, 2008.

[1] The speech acts are also called elsewhere *illocutionary* acts or *performatives*.

a clear understanding of semantics is essential. Indeed, any speech act should have a *unique interpretation*. Moreover, it should be *verifiable*, i.e. it should be possible to check whether a system conforms to a particular ACL or not [35]. Although a number of significant agent communication languages have been developed, for instance [1, 7, 13, 14, 10, 30, 31], obtaining a suitable formal semantics for ACLs which satisfies the above objectives remains one of the greatest challenges of multi-agent theory.

There are mainly three categories of semantics: *mentalistic* semantics, protocol-based semantics and *social* ones. A mentalistic semantics, used for instance in KQML [13] and FIPA [14], is based on a notion of speech act close to the concept of illocutionary act as developed in speech act theory [9, 27]. The basic idea behind this semantics is to define the conditions under which a given speech act can be played. Unfortunately, the conditions are based on the mental states (beliefs and intentions) of the interacting agents. This makes the semantics not verifiable as shown in [35], and consequently violates one of the important properties of a semantics. In the second category of semantics, the notion of dialogue protocol plays a crucial role when defining the meanings of speech acts. In [1, 24], for instance, the meaning of a speech act is given in terms of the allowed responses to that speech act by a protocol. Thus, the meaning of the same speech act may change from one protocol to another. This makes this kind of semantics not global and not suitable. The most popular category of semantics is the social one. In this kind of approach, as developed in [10, 29, 31], primacy is given to the interactions among the agents. The semantics is based on social *commitments*. A commitment is an engagement taken by an agent, called the *debtor* of the commitment, toward a set of agents, called the *creditors* of the commitment. Thus, by uttering speech acts, commitments are induced and need to be satisfied by their debtors. For example, by affirming a data, the agent commits on the truth of that data. After a promise, the agent is committed to carrying it out. While this approach overcomes the limitation of the mentalistic approach by being verifiable, itsuffers from some weak points. In fact, the concept of commitment is *ambiguous* and its semantics is not clear. According to the performative act, the semantics of the commitment differs. Another important drawback of this approach is the fact that it is not practical. Indeed, it is not clear how such an approach can be used in order to capture the dynamics of agents interactions, or how isolated speech acts can be connected for building complete and coherent dialogues.

This paper is a substantially expanded and revised version of our previous works [4, 5].The basic idea behind our semantics is to clarify the origin of each commitment and the link between a speech act and the induced commitment. Indeed, each speech act has a *goal* in a dialogue. For instance, behind a question, one expects an answer. Hence, during a dialogue, as soon as a speech act is uttered, a *commitment* for achieving its goal is *created*. Depending on the speech act, the debtor of the commitment may be either the sender of the move, or its hearer. In the case of a question, by uttering such a speech act, a commitment for giving

an answer is created, and the debtor is the hearer. Note that this does not mean at all that the hearer should necessarily give an answer. A dialogue protocol may impose such a condition, but the problem of dealing with protocols is beyond the scope of this paper.
Once the goal of each speech act is clearly stated, and the commitments induced from them are specified, we propose to associate with each commitment a *penalty* to be paid in case that commitment is violated by its debtor. This notion of penalty allows us to check at any step of a dialogue whether new commitments are created, and whether commitments are fulfilled or withdrawn. Thus, the semantics associates with each speech act a meaning in terms of the *commitment* induced by it, and a *penalty* to be paid in case that commitment is violated. The new semantics is grounded on a computational logic framework, thus allowing automatic verification of compliance by means of proof procedures. More precisely, a *violation criterion* based on the existence of arguments is defined per speech act. From a *syntactic* point of view, utterances are stored in *commitment stores* as in [20]. Each agent is supposed to be equipped with a commitment store visible to all agents.
The contribution of this paper can be summarized in four main points:

1. To clarify the origin of each commitment induced from a speech act. Moreover, the link between the two notions is established.
2. To propose a new semantics in terms of commitments to be satisfied and a penalty that needs to be paid if the commitments are violated. A violation criterion is given for each speech act. All the criteria are based on what has been exchanged during a dialogue (and not on what is in the bases of the agents), and this makes the semantics *verifiable*.
3. To propose a simple alternative of commitment-based semantics. Instead of using more complicate logics such temporal logic for handling commitment, we propose here the use of a simple penalty logic. This makes the semantics *simple*.
4. Contrarily to existing social semantics that focus only on isolated speech acts, and do not worry about how these semantics can be integrated in a concrete dialogue system, this paper proposes a semantics that is defined on the basis of moves uttered during a dialogue. This ensures that the proposed semantics is *practical*.

The paper is organized as follows: Section 2 introduces the logical language used throughout the paper. Section 3 introduces the basic concepts of our semantics. Namely, it introduces the different speech acts that will be studied, defines the notion of commitment as well as the notion of penalty. Finally, it defines for each speech act a violation criterion for its associated commitment. Section 4 studies the logical properties of the new semantics. The semantics is then illustrated through two examples in Section 5. Section 6 compares our approach with existing approaches of ACL and Section 7 is devoted to some concluding remarks and perspectives. All the proofs are given in an appendix at the end of the document.

2 The logical language

Throughout the paper, let us consider a *propositional language* $\mathcal{L}$. $\vdash$ denotes classical inference and $\equiv$ logical equivalence. A *knowledge base* Σ is a set of formulas of $\mathcal{L}$. *Arguments* can be built from any knowledge base Σ. By argument we mean a reason of believing a piece of information, or of making a choice, etc. In argumentation literature, different definitions of an argument have been proposed. In what follows, we will opt for the one suggested by Simari and Loui in [28].

Definition 1 (Argument) *An* argument *is a pair* (S, c) *where* c *is a formula of* $\mathcal{L}$ *and* $S \subseteq \Sigma$ *such that:*

1. S *is consistent,*
2. $S \vdash c$,
3. S *is minimal for set inclusion among the sets satisfying 1) and 2)*

S *is called the* support *of the argument and* c *its* conclusion. $\texttt{Arg}(\Sigma)$ *denotes the set of all the arguments that can be built from a knowledge base* Σ.

$\texttt{Arg}(\mathcal{L})$ denotes then the set of all arguments that can be built from the logical language $\mathcal{L}$.
Given that a knowledge base Σ may be inconsistent, arguments may be conflicting too. In [3], different conflict relations between arguments have been studied. In what follows we will use the relation "Undercut" which is the most suitable in our case.

Definition 2 (Undercut) *Let* $A_1 = (S_1, c_1)$, $A_2 = (S_2, c_2) \in \texttt{Arg}(\Sigma)$.
A_1 undercuts A_2 *if* $\exists\ h_2 \in S_2$ *such that* $c_1 \equiv \neg\ h_2$.

Since arguments are conflicting, it is important to know among all these arguments which are the good ones in order to be able to decide which conclusion to infer from Σ. In [11], Dung has presented a powerful argumentation framework that takes as input a set of arguments and the different conflicts which may exist between them, and returns among all the arguments the "good" ones, called the *acceptable* arguments. This notion of acceptability will not be presented here since it is not required for the definition of our semantics.

Let $\mathcal{A} = \{ag_1, \ldots, ag_n\}$ be a set of variables denoting *agents* identifiers. Each agent is assumed to have a *role* allowing it to have the control over a subset of formulas in $\mathcal{L}$. By having a control over a formula, we mean that the agent is allowed to alter the truth value of that formula.

$$\texttt{Role} : \mathcal{A} \longmapsto 2^{\mathcal{L}}$$

The roles are supposed to be visible to all the agents. Thus, each agent is aware about the formulas that it can control, and also the formulas under the control of each interacting agent. The roles are thus not private.

A communication language is based on a set of *speech acts*. Let $\mathcal{S}$ denote that set. From $\mathcal{S}$ and $\mathcal{L}$, different moves can be built and uttered during a dialogue.

Definition 3 (Move) *If $a \in \mathcal{S}$ and either $x \in \mathcal{L}$ with $x \nvdash \bot$, or $x \in Arg(\mathcal{L})$ then $m = a{:}x$ is a* move.

- *The function* `Act` *returns the speech act (*$\texttt{Act}(m) = a$*),*
- *The function* `Content` *returns the content of the move (*$\texttt{Content}(m) = x$*).*

Let $\mathcal{M}$ denote the set of all the possible moves that can be built from $\mathcal{S}$ and $\mathcal{L}$.

An example of a move is $\texttt{Question}{:}\phi$ where ϕ encodes "the sky is blue". This move means that one asks whether the sky is blue or not. Here `Question` is a speech act and ϕ is a propositional formula which can be either true or false.

Note that the exchanged information is assumed to be *consistent*. The reason is that we consider that agents are *rational* (i.e., they cannot assert absurdities).

3 Semantics

The basic idea behind our semantics is that each speech act has a *goal* in a dialogue. For instance, by asking a question, one expects to get an answer from its receiver, by making a promise one expects to see that promise realized in the future. Another important speech act in dialogues is `Assert`, through which agents exchange knowledge. The goal of `Assert` is to explicit the truth value of the exchanged information. For instance, by uttering the move $\texttt{Assert}(\phi)$ where ϕ stands for "the weather is beautiful", one may think that the sender believes ϕ, and it is ready to defend this position if challenged. Thus, during a dialogue, when a given speech act is uttered through a move, a kind of commitment for achieving the goal of that speech act is *created*. As we will see later, depending on the speech act, the debtor of the commitment may be either the sender or the receiver of the move in which the speech act is involved.
Our semantics associates with each speech act a meaning in terms of the *commitment* induced by that speech act, and a *penalty* to be paid by the agent concerned by the commitment in case that commitment is violated. For each speech act, we define a *criterion* pointing out when the corresponding commitment is violated. These criteria are all based on the existence of arguments. Note that penalties are computed in the same way for each agent and must have a unique understanding (if it is money then the money unit must be the same for each agent). The various moves uttered during a dialogue are stored in *commitment stores* which are visible to all agents.

3.1 Speech acts

We consider the following set of basic speech acts that are used in the literature (for instance [6, 8, 15, 23, 25, 36]) for modeling the different types of dialogues identified by Walton and Krabbe in [33]. Most of these speech acts are studied in existing ACL semantics.

$\mathcal{S} = \{\texttt{Assert}, \texttt{Argue}, \texttt{Declare}, \texttt{Question}, \texttt{Request}, \texttt{Challenge}, \texttt{Promise}\}$.

`Assert` allows agents to inform each other about the state of the world. Its goal is then to explicit an information and to defend it against any attack. Thus, the debtor of the commitment is the sender of the speech act. This speech act is considered as an *assertive* act according to the classification of Searle [26]. Moves using this speech act have the following form: `Assert`:x where x is a proposition ($x \in \mathcal{L}$) like "the weather is beautiful" or "it is my intention to hang a mirror".

`Argue` allows agents to support their claims by arguments. The goal of this speech act is to defend the claim supported by the argument. Thus, this argument is expected to be defended by its sender during a dialogue. Here again, the debtor of the induced commitment is the sender. This act is also an *assertive* one according to the classification of Searle [26]. A move involving this speech act is defined as follows: `Argue`:x, where x is an argument ($x \in Arg(\mathcal{L})$) (i.e., a pair $(Support, conclusion)$ where the support is itself a set of propositions and the conclusion is a proposition).

`Declare` allows to change the state of the world. Indeed, a move involving this speech act brings about a state of affairs that makes its content true. Such a move is defined as `Declare`:x, where x is a proposition, thus $x \in \mathcal{L}$. Examples of declarations are "the auction is open", and "John and Mary are husband and wife". This speech act is a *declarative* act according to the classification of Searle [26]. Its goal is then to change the state of the world. However, the agent who makes a move using this speech act should have the necessary authority to make such a change. For instance, during an auction, only the auctioneer is allowed to open the auction. It is then clear that the debtor of the induced commitment is the sender of the speech act.

`Question` is an act that incites the agent which receives it to give an answer. This act is considered as *directive* acts according to the classification of Searle [26]. A move involving this act is defined as `Question`:x, where x is a *proposition*, thus $x \in \mathcal{L}$. Through such a move, an agent asks for the truth value of x. Then, the debtor of the induced commitment is the receiver of the move involving this speech act.

`Request` like a question, a request incites the agent which receives it to give an answer. Through a move `Request`:x, with x is a *proposition* ($x \in \mathcal{L}$), an agent asks another agent to alter its value to true. `Request`:x has then a character more imperative than `Question`:x which does not ask the other agent to act on the world but only to give some information. A `Request` is used when an agent cannot, or prefers not, to achieve one of its goals alone. For instance, if the agent ag_2 utters `Request`:$ag_2\_is\_paid$ then it means that ag_2 asks for being paid. As for a question, after a request one expects that its receiver will change the value of the content x.

`Challenge` is used when an agent wants to get an explanation or an argument in favor of a given information. Its goal is then to incite its receiver to present an argument. Of course, during a dialogue, an agent which receives a challenge is not obliged to answer with an argument unless the protocol enforces it to do so. Note that the commitment concerns the receiver of the speech act. The move involving a challenge is defined as follows: `Challenge`:x where x is a *proposition*, i.e., $x \in \mathcal{L}$.

`Promise` is a *commissive* act according to Searle [26]. Its aim is to allow an agent to commit itself to some future course of action. Thus, one expects that this agent will respect its promise in the future. The expression `Promise`:x means that the agent is committed to make x true in the future, with $x \in \mathcal{L}$. For example, if an agent ag_1 utters `Promise`:ϕ, where ϕ stands for "agent ag_2 will be paid", then ag_1 commits itself to ensure that ag_2 will be paid in the future.

In addition to the above speech acts, we will consider another act called `Retract` which does not belong to the different categories of speech acts defined by Searle. It can be seen as a meta-level act allowing agents to *withdraw* commitments already made. Allowing such a move makes it possible for the agents to have a kind of non-monotonic behavior (i.e., to change their points of view, to revise their beliefs, etc.) without being sanctioned. Syntactically, `Retract` : m is a move with m being itself a move, i.e., $m \in \mathcal{M}$.

3.2 Commitments

A *commitment* is a directed obligation from one agent, called the *debtor*, to another, called *creditor*, about the truth of a given fact or to perform certain actions in the future. A commitment is seen as an 'obligation' since the debtor of the commitment is constrained to respect this commitment. Contrarily to the mental states of an agent that are private, i.e., they are not visible to the other agents involved in the dialogue, the commitments of that agent are public.
In the scientific literature, one can find proposals where the semantics of an ACL is defined in terms of commitments. Examples of these are the semantics proposed by Colombetti [10], and the one given by Singh [29, 31]. Colombetti and Singh argued that agents are social entities, involved in social interactions, so they are committed to what they say. The basic idea is that an agent is committed to a given statement as soon as it reveals this statement during a dialogue. It is worth noticing that an agent which presents a statement does not necessarily agree upon that statement. Consequently, commitments are different from the agents private mental states like beliefs. This notion allows us to represent agent dialogues as observed by the participants and by an external observant, and not on the basis of the internal agents states.
In recent inter-agent communication approaches, the notions of dialogue games and (social) commitments are central. One rather influential dialogue game is

DC, proposed by Hamblin [17] in the course of analysing the fallacy of question-begging. DC provides a set of rules for arguing about the truth of a proposition. Each player has the goal of convincing the other, and can assert or retract facts, challenge the other player's assertions, ask whether something is true or not, and demand that inconsistencies be resolved. Associated with each player is a *commitment store*, which holds the commitments of the players during the dialogue. Commitments here are the information given by the players during the dialogue. There are then rules which define how the commitment stores are updated. Take for instance the assertion, it puts a propositional statement in the speaker's commitment store. What this basically means is that, when challenged, the speaker will have to justify his claim. But this does not presuppose that the challenge will come at the next turn in the dialogue.

For our purpose, we adopt this representation. Note that in this paper we are not interested in modeling the reasoning of agents, we only consider what is said by each agent. The idea is to provide a semantics for each speech act without worrying about the mental states of agents.

Each agent is supposed to be equipped with a commitment store, accessible to all agents, that will contain the utterances made during the dialogue and that commit the agent. Thus, a commitment store keeps tracks of two kinds of speech acts:

- Speech acts made by the agent itself such as assertions, promises and declarations. Recall that commitments induced from these speech acts commit the speaker, i.e., the sender of the speech act.
- Speech acts received from other agents, such as requests, challenges and questions. For instance if an agent ag_i makes a request r to another agent ag_j, the request (r) is stored in the commitment store of ag_j. Hence, ag_j is said *committed* to answer to it. Recall that commitments induced from these speech acts commit their receivers.

Note that in [7, 22] more structured commitment stores have been proposed. However, for the purpose of our semantics, only the above simple distinction is needed.

Definition 4 (Commitment store) *A* commitment store CS_i *associated with an agent* ag_i *is a pair* $CS_i = \langle A_i, O_i \rangle$ *with:*

- $A_i \subseteq \{m \in \mathcal{M} | \texttt{Act}(m) \in \{\texttt{Assert}, \texttt{Argue}, \texttt{Declare}, \texttt{Promise}\}\}$.
- $O_i \subseteq \{m \in \mathcal{M} \mid \texttt{Act}(m) \in \{\texttt{Question}, \texttt{Request}, \texttt{Challenge}\}\}$.

A dialogue evolves from one step to another as soon as a move is uttered. In what follows, CS_i^s denotes the commitment store of agent i at step s. A commitment store is supposed to be *empty* at the beginning of a dialogue (i.e., at step 0). Hence, for any agent ag_i, $CS_i^0 = \varnothing$. Then, each move uttered during a dialogue is stored in a commitment store except the move retract. Indeed, this last does

not commit neither its sender nor its receiver to anything. Its role is to retract some previously stated moves.
Given a set X of moves, X^i denotes the moves of X that are uttered from step 0 to step i. Let us now introduce two functions $\texttt{PROP}$ and $\texttt{PROP}_P$ that return sets of formulas as follows:

Definition 5 *Let $X \subseteq \mathcal{M}$.*

- $\texttt{PROP}(X)$ *is defined recursively by:*
  $\texttt{PROP}(X^0) = \varnothing$
  $$\texttt{PROP}(X^s) = \begin{cases} \texttt{PROP}(X^{s-1}) \cup \{x\} & \text{if } m = \texttt{Assert}{:}x \\ \texttt{PROP}(X^{s-1}) \cup S & \text{if } m = \texttt{Argue}{:}(S,c) \quad \text{where } m \text{ is the move} \\ \texttt{PROP}(X^{s-1}) \diamond x & \text{if } m = \texttt{Declare}{:}x \\ \texttt{PROP}(X^{s-1}) & \text{else} \end{cases}$$
  uttered at step s in X^s and $\diamond$ is an update operator described below.
- $\texttt{PROP}_P(X) = \{x \in \mathcal{L}$ *such that* $\exists \texttt{Promise}{:}x \in X\}$.

The above definition computes the set of formulas that represent the state of the world (according to what has been uttered during the dialogue). Note that Questions, Challenges and Requests are not considered in the definition since they don't describe the state of the world. Formulas that appear in assertions and arguments are directly taken into account. However, things are different with the formulas related to a move `Declare`. Indeed, by definition, after $\texttt{Declare}{:}x$ the world evolves in such a way that x becomes true. Consequently, one has to update the whole set of propositions previously uttered. For that purpose, an update operator [18, 34], denoted by $\diamond$, is needed. Several update operators have been introduced in the literature. The choice of the precise one to be used in our semantics is beyond the scope of this paper.

3.3 The notion of penalty

It is natural to associate with each commitment a penalty that sanctions agents when the commitment is violated.
The need of *cumulating* sanctions when several violations have occurred is a reason for using a penalty based framework, in which *additivity* is crucial. Our basic idea is to adapt the penalty logic framework, proposed in [12] for handling inconsistency in knowledge bases, to the case of handling commitments and their violation in our ACL semantics.
Let us first recall the principles of penalty logic. A *penalty knowledge base* is a multi-set of pairs $\langle \varphi_i, \alpha_i \rangle$ where φ_i is a propositional formula, and α_i is a cost to be paid in case the formula φ_i is violated. This cost is a strictly positive number or may be infinite, thus it is an element of $\mathbb{N}^* \cup \{+\infty\}$. When $\alpha_i = +\infty$, this means that it is forbidden to violate the formula φ_i. Given a penalty knowledge base KB, it is possible to compute the penalty to be paid for getting the base KB consistent. This amounts to remove the less important formulas from KB knowing that removing a formula φ_i induces a penalty α_i. More precisely, the

penalty of a consistent subset T of KB is the sum of costs of the formulas that are not in T:

$$C(T) = \sum_{\varphi_i \in KB \setminus T} \alpha_i$$

Note that when the penalty knowledge base KB is consistent, its associated penalty will be equal to zero.

In what follows, the above concepts and definitions will be adapted in order to capture the semantics of speech acts. For that purpose, we first need to define the penalty knowledge base in the ACL context. As said before, each interacting agent ag_i is equipped with a commitment store CS_i keeping track of moves that commit this agent ag_i. Each move m_j in CS_i may sanction the agent if this agent violates the commitment induced by the speech act $\texttt{Act}(m_j)$. Thus, for each commitment store, one can define its corresponding penalty base defined as follows:

$$PCS_i = \{\langle m_j, \alpha_j \rangle | m_i \in CS_i \text{ and } \alpha_j \in I\!N^\star \cup \{+\infty\}\}$$

The value α_j denotes the cost of the commitment induced by $\texttt{Act}(m_j)$. The question now is where does this value come from? Does is depend on the whole move (i.e., the speech act and its content) or only on the speech act? In what follows, the cost is supposed to depend only on the speech act. Indeed, each speech act in $\mathcal{S}$ is supposed to have a *cost* which is a strictly positive integer or the infinity:

$$\texttt{Cost} : \mathcal{S} \longmapsto I\!N^\star \cup \{+\infty\}.$$

Thus, $\alpha_j = \texttt{Cost}(Act(m_i))$. Different speech acts may have different values. This captures the idea that some speech acts are more important than others. For instance, violating a promise may be more costly than not answering a question. Of course, this can be extended to the case where penalty depends also on the content of moves since a speech act is generally accompanied by some content. However, for the sake of simplicity, we consider here the simple case, and leave the notion of content for further research.
The penalty associated with each commitment store is computed as follows:

Definition 6 (Penalty) *Let $CS_i = \langle A_i, O_i \rangle$ be a commitment store, and $X \subseteq A_i \cup O_i$. The* penalty *associated with X w.r.t. CS_i is*

$$c(X) = \sum_{m \in X} \texttt{Penalty}(m)$$

$$\texttt{Penalty}(m) = \begin{cases} \texttt{Cost}(\texttt{Act}(m)) & \textit{if the commitment } m \textit{ is violated in } A_i \\ 0 & \textit{otherwise} \end{cases}$$

Since a commitment store is empty at the beginning of a dialogue, its initial penalty is equal to 0. Moreover, at any step, the penalty of a given commitment store can be computed in a very simple way as shown in the next section.

3.4 Violation criteria

As shown before, a penalty is to be paid if a commitment is violated. This section presents in details when commitments induced from each speech act of $\mathcal{S}$ are violated. Subsequently, we suppose that the agent ag_i utters the move to the agent ag_j.

1. `Assert`:x During a dialogue, an agent can assert that a given propositional formula is true. This agent is not allowed to contradict itself during all the dialogue otherwise it will have to pay a penalty (except if it retracts that proposition). Indeed, a move `assert`:x is *violated* if the A_i part of the commitment store of the agent ag_i makes it possible to find an argument with a conclusion $\neg x$. Formally:

Definition 7 *A move* `Assert`:x *is* violated *iff*

$$\exists (S, \neg x) \in \texttt{Arg}(\texttt{PROP}(A_i)).$$

In order to avoid any form of wishful thinking, in the above definition, the promises made by the agent are not taken into account when checking the violation of an assert move, even if they are stored in the A_i part of the commitment store. Indeed, promises may not be satisfied yet by the agent.

2. `Argue`:x During a dialogue, an agent can provide an argument x in favor of some conclusion. Then, this agent is not allowed to contradict itself in the sense that it cannot produce an undercutter against x.

Definition 8 *A move* `Argue`:x *is* violated *iff*

$$\exists (S', y) \in \texttt{Arg}(\texttt{PROP}(A_i)) \textit{ such that } (S', y) \textit{ undercuts}^2\ x.$$

As for assert moves and for the same reason, promises are not taken into account when looking for counter arguments.

3. `Declare`:x During a dialogue, an agent can modify the state of a certain proposition x by declaring it true. The move `Declare`:x commits the honesty of the agent which carries it out in the sense that the agent should be empowered to modify the value of x. This capacity is defined by the role of the agent. For instance, it is not allowed for a simple citizen to marry people. Moreover, an agent can really modify this value only if there is no argument against performing that action. Formally:

Definition 9 *A move* `Declare`:x *is* violated *iff*

$$x \notin \texttt{Role}(ag_i) \textit{ or } \exists (S, \neg y) \in \texttt{Arg}(\texttt{Prop}(A_i)) \textit{ with } y \in \texttt{Precond}(x)$$

[2] See Definition 2 in Section 2.

where $\texttt{Precond} : \mathcal{L} \rightarrow 2^{\mathcal{L}}$ *is a function that gives for any formula* φ *the pre-conditions for setting* φ *to true and that verifies:* $\texttt{Precond}(\bot) = \{\bot\}$ *and* $\texttt{Precond}(\top) = \varnothing$.

The definition of Precond is beyond the scope of this paper. In the rest of the paper, it is supposed to be given. For example, in order to open an auction, one should check whether the buyers are present. If a formula can never be set to true then the function Precond returns $\{\bot\}$. When, there is no pre-condition for setting the formula to true, the function returns $\varnothing$.

4. Question:x During a dialogue, an agent may receive questions from other agents to which it is committed to answer either positively or negatively. The absence of any argument in favor of x or $\neg x$ in the part A_j of the commitment store of the agent that receives the move means that the agent has not given any answer.

Definition 10 *A move* Question:x *is* violated *iff*

- $\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(A_j))$ *and*
- $\nexists\ (S, \neg x) \in \texttt{Arg}(\texttt{PROP}(A_j))$.

Again, promises are not considered when building arguments. Note that we check the existence of an argument in favor of x or $\neg x$ instead of just the existence of a proposition equivalent to x or to $\neg x$ in A_j. The reason is that the question can be answered implicitly via other assertions of the agent. In this setting, it is not possible to answer "I don't know" to a question. But, this could be easily handled by introducing a new speech act Desinform.

5. Request:x An agent is expected to give a positive or a negative answer to any request it receives from other agents.

Definition 11 *A move* Request:x *is* violated *iff*

- $\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(A_j) \cup \texttt{PROP}_P(A_j))$ *and*
- $\nexists\ (S', \neg x) \in \texttt{Arg}(\texttt{PROP}(A_j) \cup \texttt{PROP}_P(A_j))$.

Note that to check whether a request is violated or not, we look for an argument in favor of x in both $\texttt{PROP}(A_j)$ and $\texttt{PROP}_P(A_j)$. The reason is that a request can get an answer in two ways: either through a promise ensuring that in the future the requested proposition will be set to true or to false, or because it is already stated (either by declarations or assertions) to true or false.

6. Challenge:x Agents are committed to provide arguments for any challenged proposition, otherwise their commitment is violated.

Definition 12 *A move* Challenge:x *is* violated *iff*

$$\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(A_j)) \textit{ with } S \not\equiv \{x\}.$$

Let us take the example of an agent which asserts x, after which the other agent makes a challenge on x. It is clear that the argument $(\{x\}, x)$ can be built from $\texttt{Arg}(\texttt{PROP}(A_j))$, however this is not an answer to the challenge. In order to avoid such problem, the above definition requires that the argument presented after a challenge should be different from x.

7. $\texttt{Promise}:x$ During a dialogue, an agent may make promises to other agents. This agent should pay a penalty in case it does not respect this promise. This can be checked on the part A_i of its commitment store. Indeed, if an argument in favor of proposition x can be built then the promise is honored otherwise it is considered violated.

Definition 13 *A move* $\texttt{Promise}:x$ *is* violated *iff*

$$\nexists (S, x) \in \texttt{Arg}(\texttt{PROP}(A_i)).$$

8. $\texttt{Retract}:m$ Agents may decide to retract some previously uttered moves. The aim of this move is to allow agents to revise their beliefs without being sanctioned. The speech act $\texttt{Retract}$ is different from the other elements of $\mathcal{S}$ since it does not induce any commitment, thus $\texttt{Penalty}(\texttt{Retract}:m) = 0$. Moreover, after such a move the commitment store is updated as follows:

Definition 14 *Let* CS_i^s *be the commitment store of an agent* ag_i *at step* s*. A move* $\texttt{Retract}(m)$ *at step* $s+1$ *has the following effect:*

$$CS_i^{s+1} = CS_i^s \setminus \{m\}$$

with $m \in CS_i^s$*.*

According to the above definitions, it is clear that a commitment is considered as violated as soon as the corresponding speech act is uttered in a dialogue. This indicates that a commitment is created. When, that commitment is fulfilled or withdrawn the penalty of the commitment store decreases. Note also that time is handled implicitly in our semantics. Indeed, a commitment store is defined in such a way that at each "step" of the dialogue; it is possible to compute the set of moves that have been uttered. Thus, it is possible to check at each step which are the commitments that are created, which ones are fulfilled, etc.

3.5 Summary

A summary of the semantics of the various speech acts is given in Table 1.

4 Properties

The aim of this section is to show that the proposed semantics satisfies some key and desirable properties. The first property ensures that the semantics sanctions only bad behaviors of agents, and that any bad behavior is sanctioned.

Move	Stored in	Violated if	Penalty
Assert:x	A_i	$\exists\ (S, \neg x) \in \texttt{Arg}(\texttt{PROP}(A_i))$	Cost(Assert)
Argue:(S, c)	A_i	$\exists (S', y) \in \texttt{Arg}(\texttt{PROP}(A_i))$ such that (S', y) undercuts (S, c)	Cost(Argue)
Declare:x	A_i	- $x \notin \texttt{Role}(ag_i)$ - or $\exists (S, \neg y) \in \texttt{Arg}(\texttt{PROP}(A_i))$ with $y \in \texttt{Precond}(x)$.	Cost(Declare)
Question:x	O_j	$\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(A_j))$ and $\nexists\ (S, \neg x) \in \texttt{Arg}(\texttt{PROP}(A_j))$	Cost(Question)
Request:x	O_j	$\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(A_j) \cup \texttt{PROP}_P(A_j))$ and $\nexists\ (S', \neg x) \in \texttt{Arg}(\texttt{PROP}(A_j) \cup \texttt{PROP}_P(A_j))$	Cost(Request)
Challenge:x	O_j	$\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(A_j))$ with $S \not\equiv \{x\}$	Cost(Challenge)
Promise:x	A_i	$\nexists (S, x) \in \texttt{Arg}(\texttt{PROP}(A_i))$	Cost(Promise)
Retract:x	not stored	None	0

Table 1. The semantics of the speech acts

Proposition 1
- *If* $c(CS_i) > 0$, *then* $\exists\ m \in CS_i$ *such that* m *is violated.*
- *If* $\exists\ m \in CS_i$ *such that* m *is violated, then* $c(CS_i) > 0$.

A quite obvious property, that follows directly from the definition of a commitment store and from the definitions of the different violation criteria, states that the cost of a commitment store is independent of the syntax of the contents of the moves.

Proposition 2 (Syntax independence)
$\forall m, m' \in \mathcal{M}$,
if $\texttt{Act}(m) = \texttt{Act}(m')$ *and* $\texttt{PROP}(\{m\}) \equiv \texttt{PROP}(\{m'\})$,
then $c(\{m\}) = c(\{m'\})$.

Another important result is the fact that if the total penalty of part A_i is null then all the stated information is consistent.

Proposition 3 (Consistency)
If $\sum_{m \in A_i} \texttt{Penalty}(m) = 0$ *then* $\texttt{PROP}(CS_i)$ *is consistent.*

Before introducing other interesting properties, we first need to define a notion of *independency*. For this purpose, we will use a notion of *novelty* introduced by Greiner and Genesereth in [16], and analyzed more deeply in the propositional case by Marquis and Lang in [21, 19]. Indeed, in [19] the novelty of a propositional formula φ with respect to another formula ψ in a given context is defined by the fact that φ allows to built a minimal abductive explanation of ψ or $\neg\psi$, while this is not possible in the same context without φ. Considering that a minimal abductive explanation is an argument, this leads to propose the following definition.
Intuitively, a formula φ is *new* for ψ in a context Σ, if φ allows to deduce new arguments for ψ or for $\neg\psi$. This means that φ is linked to ψ in the context

Σ. When this is not the case, this means that φ does not allow to deduce new arguments neither for ψ nor for $\neg\psi$. In that case, we say that φ is *independent* from ψ. Note that this relation is not symmetric.

Definition 15 *(Novelty) Let φ, ϕ be two propositional formulas, and Σ a set of formulas.*

- *φ is* new *for ϕ w.r.t. Σ iff:*
  - $\exists\ (S,\phi) \in \texttt{Arg}(\Sigma \cup \{\varphi\})$ *and* $(S,\phi) \notin \texttt{Arg}(\Sigma)$, *or*
  - $\exists\ (S,\neg\phi) \in \texttt{Arg}(\Sigma \cup \{\varphi\})$ *and* $(S,\neg\phi) \notin \texttt{Arg}(\Sigma)$
- *φ is said to be* independent *from ϕ w.r.t. Σ otherwise.*

This above definition allows us to define a notion of independency between two moves given a dialogue. Namely, two moves m_1 and m_2 are independent if m_2 has no influence on m_1. This means that the content of the first move should be independent from the properties of the second move. When one of the two moves is based on a declarative speech act (`Declare` in our case), the preconditions of this move should be independent from the properties of the other move.

Definition 16 *(Move independency) Let $X \subseteq \mathcal{M}$ be a set of n moves and let m be a move uttered at step $s \leq n$ ($m = X^s$), let m' be a new move uttered at step $n+1$.*
m' is independent *from m given X if*

- *when* $\texttt{Act}(m) \neq \texttt{Declare}$ *then* $\texttt{Content}(m)$ *is independent from* $\texttt{PROP}(m')$ *given* $\texttt{PROP}(\texttt{X}) \cup \texttt{PROP}_\texttt{P}(\texttt{X})$
- *when $m = Declare : x$ then* $\texttt{Precond}(x)$ *is independent from* $\texttt{Content}(m')$ *given* $\texttt{PROP}(\texttt{X}) \cup \texttt{PROP}_\texttt{P}(\texttt{X})$

Note that in the above definition, we consider the independency between the content of the first move and the formulas of the second move. This is because we must check if the last move has influenced the commitment induced by the first move. The commitment of the first move depends on its content with respect to the information exchanged during the dialogue. The set `PROP` contains the formulas that are asserted or argued, hence they may influence the state of knowledge. The content of the move maybe of no interest with respect to this state of knowledge when the speech act is a question or a challenge.

Lemma 1. *Let $X \subseteq \mathcal{M}$ be a set of n moves and let m be a move uttered at step $s \leq n$ ($m = X^s$), let m' be a new move uttered at step $n+1$.*
If m' is independent from m w.r.t. X and if $\texttt{Act}(m') \notin \{\texttt{Declare}, \texttt{Retract}\}$ *then for all* $x \in \texttt{Content}(m)$

$$\forall S \subseteq \mathcal{L}, \quad (S,x) \in \texttt{Arg}(\texttt{PROP}(X \cup \{m'\})) \\ (S,\neg x) \in \texttt{Arg}(\texttt{PROP}(X)) \Leftrightarrow (S,\neg x) \in \texttt{Arg}(\texttt{PROP}(X \cup \{m'\}))$$

This lemma claims that when two moves are independent, if there are no arguments against or in favor of the content of the first move then the arrival of the second move will not change it. This lemma allows to settle the following property about the evolution of the violation of a move when an independent move is uttered. Namely, the violation status does not change when an independent move is uttered.

Proposition 4 *Let CS_i be a commitment store obtained after n steps, m be a move already uttered in CS_i, and let $m' \in \mathcal{M}$ be a move uttered at step $n+1$.*

If m' is independent from m w.r.t. CS_i and if $\texttt{Act}(m') \notin \{\texttt{Declare}, \texttt{Retract}\}$ then m is violated in $CS_i \Leftrightarrow m$ is violated in $CS_i \cup \{m'\}$.

This proposition is only valid with a second move which is not a `Declare`, in order to extend this result we have to impose that the update operator should be independency compatible.

Definition 17 *An update operator $\diamond$ is* independency compatible *iff for every formulas φ, ψ, and every set of formulas Σ,*
if φ is independent to ψ w.r.t. Σ then $\Sigma \vdash \psi \Leftrightarrow \Sigma \diamond \varphi \vdash \psi$

Proposition 5 *Let CS_i be a commitment store obtained after n steps, m be a move already uttered in CS_i, and let $m' \in \mathcal{M}$ be a move uttered at step $n+1$.*

If the update operator associated to `Declare` *is independency compatible and if m' is independent from m w.r.t. CS_i and if $\texttt{Act}(m') \neq \texttt{Retract}$ then m is violated in $CS_i \Leftrightarrow m$ is violated in $CS_i \cup \{m'\}$.*

Proposition 4 implies that if two formulas are independent w.r.t. the formulas of a commitment store, then the penalty of two moves conveying these formulas is decomposable. Formally:

Corollary 51 (Independence) *Let CS_i be a commitment store obtained after n steps, m be a move already uttered in CS_i, and let $m' \in \mathcal{M}$ be a move uttered at step $n+1$.*

If m is independent from m' w.r.t. CS_i, and $\texttt{Act}(m) \notin \{\texttt{Declare}, \texttt{Retract}\}$ then $c(\{m, m'\}) = c(\{m\}) + c(\{m'\})$

Proposition 5 allows to extend this property for `Declare` moves when the associated update operator is independency compatible.

5 Illustrative examples

In this section we present two dialogues between agent ag_1 and agent ag_2 using the semantics presented in this paper. We suppose that the dialogue sequences presented below are allowed by a given protocol. Our analysis focuses only on agent ag_1. We will show that in the first dialogue, ag_1 behaves very well, so it has no penalty to pay at the end of the dialogue. Whereas in the second dialogue, ag_1 violates some of its commitments, and is expected to pay a penalty.

Example 1 *In this example, we will show how the commitment store of an agent ag_1 evolves during a dialogue with another agent ag_2. In what follows, the expression $ag_2 \rightarrow ag_1$ means that agent ag_2 addresses a move to agent ag_1. At the beginning of the dialogue, the commitment store is as follows:*

A_1	O_1
$\varnothing$	$\varnothing$

$c(CS_1) = 0$

$ag_2 \rightarrow ag_1$: *Give me a nail please. (ag2n stands for "ag_2 has the nail")*

A_1	O_1
$\varnothing$	Request:$ag2n$

$c(CS_1) =$ Cost(Request)

$ag_1 \rightarrow ag_2$: *No.*

A_1	O_1
Assert:$\neg ag2n$	Request:$ag2n$

$c(CS_1) = 0$

$ag_2 \rightarrow ag_1$: *Why not?*

A_1	O_1
Assert:$\neg ag2n$	Request:$ag2n$ Challenge:$\neg ag2n$

$c(CS_1) =$ Cost(Challenge)

$ag_1 \rightarrow ag_2$: *Because I want to hang a mirror (hm for "hang a mirror") and thus I need this nail (nn for "need a nail"). I cannot give you a nail if I need it.*

A_1	O_1
Assert:$\neg ag2n$ Argue:$(\{hm,\ hm \rightarrow nn,\ nn \rightarrow \neg ag2n\},\ \neg ag2n)$	Request:$ag2n$ Challenge:$\neg ag2n$

$c(CS_1) = 0$

In this dialogue the agent ag_1 has an exemplary behavior since after each move, the penalties associated with its commitment store are canceled. That means that it does not contradict itself (regarding its assertion and its argument), and that it answers to the request and to the challenge made by ag_2.

Example 2 *Let us study the following dialogue between two agents ag_1 and ag_2:*

$ag_2 \rightarrow ag_1$: *Do you think that Newspapers can publish (pub) the information X.*

A_1	O_1
$\varnothing$	Question:pub

$c(CS_1) =$ Cost(Question)

$ag_1 \rightarrow ag_2$: *No.*

A_1	O_1
Assert:$\neg pub$	Question:pub

$c(CS_1) = 0$

$ag_2 \rightarrow ag_1$: *why?*

A_1	O_1
Assert:$\neg pub$	Question:pub Challenge:$\neg pub$

$c(CS_1) =$ Cost(Challenge)

$ag_1 \rightarrow ag_2$: *Because X concerns the private life of A (pri) and A does not agree to publish it (agr).*

A_1	O_1
Assert:$\neg pub$ Argue:$(\{pri, \neg agr,$ $pri \wedge \neg agr \rightarrow \neg pub\}, \neg pub)$	Question:pub Challenge:$\neg pub$

$c(CS_1) = 0$

$ag_2 \rightarrow ag_1$: *But A is a minister (min) and information about ministers are public.*

A_2	O_2
Argue:$(\{min, min \rightarrow \neg pri\}, \neg pri)$	$\varnothing$

$c(CS_2) = 0$

$ag_1 \rightarrow ag_2$: *Yes, you are right.*

A_1	O_1
Assert:$\neg pub$	Question:pub
Argue:$(\{pri, \neg agr,$	Challenge:$\neg pub$
$pri \wedge \neg agr \rightarrow \neg pub\}, \neg pub)$	
Argue:$(\{min, min \rightarrow \neg pri\}, \neg pri)$	

$c(CS_1) = 2 \times$ Cost(Argue)

In the above example, the agent ag_1 answers the question and the challenge it received, thus there is no penalty to pay for those moves. However, this agent has presented an argument $a = <\{pri, \neg agr, pri \wedge \neg agr \rightarrow \neg pub\}, \neg pub>$, and accepted its undercutter $b = <\{min, min \rightarrow \neg pri\}, \neg pri>$. Consequently, the set PROP(CS_i) is inconsistent, and this makes it possible to even construct an undercutter $c = <\{pri, min \rightarrow \neg pri\}, \neg min>$ for the argument b. The agent has then to pay twice the cost of an Argue move. Note, however that from PROP(CS_i) it is not possible to construct an argument whose conclusion is pub. This means that the agent is still coherent w.r.t. its assertion ($\neg pub$). Thus, there is no cost to pay for the assert move.

6 Related work

As already said in the introduction, there are mainly three families of approaches to ACL semantics:

1. mentalistic semantics
2. protocol-based semantics
3. commitment-based semantics

The first standard agent communication languages are KQML [13] and FIPA-ACL [14]. Both languages have been given a mentalistic semantics. The semantics is based on a notion of speech act close to the concept of illocutionary act as developed in speech act theory [26]. Such semantics assumes, more or less explicitly, some underlying hypothesis, in particular, that agents are "sincere" and "cooperative". While this may be well fitted for some special cases of interactions, it is obvious that negotiation dialogues are not cooperative. Another more important limitation of this approach is the fact that it is not verifiable since it is based on agents mental states. Our semantics does not refer at all to the mental states of the agents. Moreover, it treats another speech act, namely Argue, which allows agents to exchange arguments.

The approach developed by Pitt and Mamdani in [24] and Alberti et al. in [2] is based on a notion of protocol. A protocol defines what sequences of moves

are conventionally expected in a dialogue. The meaning of a speech act equates to the set of possible following answers. This turns out to be too rigid in several circumstances. Current research aims at defining flexible protocols, which rely more on the *state of the dialogue*, and less on dialogue history. This state of dialogue is captured by the notion of commitment. Moreover, the meaning of a speech act is not unique since it may change from one protocol to another.

The social semantics developed in [10, 31] are the most closest to our work. In these works, semantics is based on social commitments brought about by performing a speech act. For example, by affirming a data, an agent commits on the truth of that data. After a promise, the agent is committed carrying it out. In his work, Singh has proposed a formal language based on CTL, and a formal model in which the notion of commitment is described by using an accessibility relation. In the same line of research, Colombetti and Verdicchio have proposed a logical model of commitments using CTL* [32]. A number of predicates in order to represent events and actions have been introduced. While these models are expressive enough, we think that they suffer from the following drawbacks:

1. The definition of commitments complicates the agent architecture in the sense that it needs an ad hoc apparatus. Commitments are introduced especially for modeling communication. Thus, agents should reason not only on their beliefs, desires, etc, but also on commitments. In our approach, we didn't introduce any new language for encoding or handling commitments. The only thing needed to define the meaning of each speech act is classical logic, simple commitment stores, and a procedure based on the existence of arguments for checking whether commitments are violated or not. Note that, arguments are also used by agents during a dialogue for justifying their claims. Thus, they are not particularly introduced for handling commitments. In sum, the reasoning models of agents (i.e., the models defined for reasoning about its beliefs and desires) are sufficient for capturing the semantics of an ACL, there is no need to introduce a new model.
2. The level at which communication is treated is very abstract, and there is a considerable gap to fill in order to bring the model down to the level of implementation, i.e., to be used in practical dialogues. Our semantics handles very well the dynamics of dialogue. It takes as input a dialogue expressed in terms of moves stored in commitment stores.
3. The concept of commitment itself is ambiguous and its semantics is not clear. In [10, 31], for example, by affirming a data, an agent commits on the truth of that data. The meaning of the commitment here is not clear. It may be that the agent can justify the data or can defend it against any attack, or that the agent is sincere. In this paper we have proposed for each speech act a clear, intuitive, simple and unambiguous semantics. This was possible thanks to the notion of goal of a speech act.

Another semantics which is considered as *hybrid* has been proposed in [7]. This semantics is a novel combination of an agent-internal reasoning level and an agent-external commitment level within a single two-level framework. Like our

approach, contents of moves are supposed to be stored in a structured commitment store. The meaning of each speech act is given by a pre-condition that should be satisfied before uttering a move involving that speech act, and a post-condition with specifies how the commitment store is updated after the move. The pre-condition refers to the mental states of agents. It is expressed in terms of the existence or the absence of arguments. Consequently, this approach is not verifiable. Another important weakness of that approach is that it is not clear when commitments are fulfilled or violated. The semantics presented in this paper answers clearly these two questions via the penalty value associated to each commitment store.

7 Conclusion and perspectives

This paper has introduced a new simple and verifiable ACL semantics. The interpretation of each speech act equates to the penalty to be paid in case the commitment induced by that speech act is violated. In this semantics, a violation criterion is given for each considered speech act. Note that in order to add a new speech act, one needs simply to define a new violation criterion associated with it. This semantics is based on propositional logic, and the violation criteria amount to compute arguments. Our semantics satisfies interesting properties that show its well-foundedness. It also offers other advantages regarding dialogue protocols. For instance, one does not need to specify the different moves allowed after each move in the protocol itself. Agents only need to minimize the penalty to pay at the end of the dialogue. This give birth to very flexible protocols. Protocols can also be simplified by extending our semantics by rules which were generally defined in the protocol itself. For instance, in the semantics, we can sanction agents which repeat the same move several times during a dialogue. Another rule which can be removed from the protocol is that of turn taking. One can imagine that agents can make several moves per turn but they have to pay a penalty for that. In doing so, the protocols are more flexible and consequently, the agent's strategies become very rich.

An extension of this work to first order logic is under study. Another interesting extension would be to handle explicitly time in order to be able to deal with deadlines for instance.

Moreover, the notion of penalty may play a key role in defining agent's *reputation* and *trust* degrees. It is clear that an agent that pays a lot of penalties during dialogues may lose its credibility, and will no longer be trusted. Examining more deeply penalties can help to figure out agents profiles: cooperative agent, consistent agent, thoughtful agent (i.e., agent which respects its promises)...

Another possible refinement consists of introducing granularity in the definition of the function `Cost`. The basic idea is to take into account the content of moves when defining their costs. This captures the idea that, for instance, some asserted propositions are more important than others. For example, affirming that the weather is beautiful can be less important than affirming that the president is dead.

References

1. M. Alberti, A. Ciampolini, M. Gavanelli, E. Lamma, P. Mello, and P. Torroni. Logic based semantics for an agent communication language. In *Barbara Dunin-Keplicz and Rineke Verbrugge, editors, FAMAS'03 - Formal Approaches to Multi-Agent Systems*, pages 21–36, 2003.
2. M. Alberti, A. Ciampolini, M. Gavanelli, E. Lamma, P. Mello, and P. Torroni. A social ACL semantics by deontic constraints. In *3rd International Central and Eastern European Conference on Multi-Agent Systems CEEMAS'03, volume 2691 of LNAI*, pages 204–213, 2003.
3. L. Amgoud and C. Cayrol. Inferring from inconsistency in preference-based argumentation frameworks. *International Journal of Automated Reasoning*, Volume 29, N2:125–169, 2002.
4. L. Amgoud and F. Dupin de Saint Cyr. A semantics for agent communication languages based on commitments and penalties (a preliminary report). In *Sixth International Workshop on Computational Logic in Multi-Agent Systems, CLIMA'05*, 2005.
5. L. Amgoud and F. Dupin de Saint Cyr. Towards ACL semantics based on commitments and penalties. In *17th European Conference on Artificial Intelligence, ECAI'06*, pages 235–239, 2006.
6. L. Amgoud, N. Maudet, and S. Parsons. Modelling dialogues using argumentation. In E. Durfee, editor, *Proceedings of the 4th International Conference on Multi-Agent Systems, ICMAS'00*, pages 31–38, Boston, USA, July 2000. IEEE Press.
7. L. Amgoud, N. Maudet, and S. Parsons. An argumentation-based semantics for agent communication languages. In F. Van Harmelen, editor, *Proceedings of the European Conference on Artificial Intelligence, ECAI'02*, pages 38–42, Lyon, France, July 2002. IOS Press.
8. L. Amgoud, S. Parsons, and N. Maudet. Arguments, dialogue, and negotiation. In W. Horn, editor, *Proceedings of the European Conference on Artificial Intelligence, ECAI'00*, pages 338–342, Berlin, Germany, August 2000. IOS Press.
9. J. L. Austin. How to do things with words. In *Clarendon Press, Oxford Uk*, 1962.
10. M. Colombetti. A commitment-based approach to agent speech acts and conversations. In *Proceedings of the Workshop on Agent Languages and Conversation Policies. I4th International Conference on Autonomous Agents*, pages 21–29, 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 Journal*, 77:321–357, 1995.
12. F. Dupin de Saint-Cyr, J. Lang, and T. Schiex. Penalty logic and its link with Dempster-Shafer theory. In *Proceedings of the* 10^{th} *Conference on Uncertainty in Artificial Intelligence, UAI'94*, pages 204–211. Morgan Kaufmann, July 1994.
13. T. Finin, Y. Labrou, and J. Mayfield. KQML as an agent communication language. In *J. Bradshaw, ed. Softaware agents, MIT Press, Cambridge.*, 1995.
14. FIPA. ACL message structure specification. In *FIPA 02. Foundation for Intelligent Physical Agents*, 2002.
15. T. F. Gordon. The pleadings game. *Journal of Artificial Intelligence and Law*, 2:239–292, 1993.
16. R. Greiner and M. Genesereth. What's new? a semantic definition of novelty. In *Proceedings of the Eight International Joint Conference on Artificial Intelligence, IJCAI'83*, pages 450–454, 1983.
17. C. L. Hamblin. *Fallacies.* Methuen, London, UK, 1970.

18. H. Katsuno and A.O. Mendelzon. On the difference between updating a knowledge base and revising it. In J. Allen and al., editors, *Proc. of the 2nd International Conference on Principles of Knowledge Representation and Reasoning, KR'91*, pages 387–394, Cambridge, MA, 1991.
19. J. Lang, P. Liberatore, and P. Marquis. Conditional independence in propositional logic. *Artificial Intelligence Journal*, Volume 141(1-2):79–121, 2002.
20. J. MacKenzie. Question-begging in non-cumulative systems. *Journal of philosophical logic*, 8:117–133, 1979.
21. P. Marquis. Novelty revisited. In *Proceedings of the Sixth International Symposium on Methodologies for Intelligent Systems, ISMIS'91*, pages 550–559, 1991.
22. P. McBurney and S. Parsons. Posit spaces: a performative theory of e-commerce. In M. Wooldridge J. S. Rosenschein, T. Sandholm and M. Yokoo, editors, *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems, AAMAS'03*, pages 624–631, New York City, NY, USA, 2003. ACM Press.
23. 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.
24. J. Pitt and A. Mamdani. A protocol based semantics for an agent communication language. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence, IJCAI'99*, pages 486–491, 1999.
25. H. Prakken. On dialogue systems with speech acts, arguments, and counterarguments. In *7th European workshop on Logic for Artificial Intelligence, JELIA'00*, Berlin, 2000.
26. J. R. Searle. Speech acts. In *Cambridge University Press*, 1969.
27. J. R. Searle and D. Vanderveken. Foundations of illocutionary logic. In *Cambridge University Press*, 1985.
28. G. R. Simari and R. P. Loui. A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence Journal*, 53:125–157, 1992.
29. M. P. Singh. Agent communication languages: Rethinking the principles. In *IEEE Computer*, pages 40–47, 1998.
30. M. P. Singh. An ontology for commitments in multiagent systems: toward a unification of normative concepts. *Artificial Intelligence and Law*, 7:97–113, 1999.
31. M. P. Singh. A social semantics for agent communication languages. In *IJCAI'99 Workshop on Agent Communication Languages*, pages 75–88, 1999.
32. M. Verdicchio and M. Colombetti. A logical model of social commitment for agent communication. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS'03*, pages 528–535, 2003.
33. D. N. Walton and E. C. W. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. State University of New York Press, Albany, NY, 1995.
34. M. Winslett. *Updating Logical Databases*. Cambridge University Press, Cambridge, UK, 1990.
35. M. J. Wooldridge. Semantic issues in the verification of agent communication languages. *Journal of Autonomous Agents and Multi-Agent Systems*, 3(1):9–31, 2000.
36. 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.

Appendix

Proposition 1

- If $c(CS_i) > 0$, then $\exists\ m \in CS_i$ s.t. m is violated.
- If $\exists\ m \in CS_i$ s.t. m is violated, then $c(CS_i) > 0$.

Proof. It comes directly from the definition of the cost of a commitment store: $c(CS_i) = \sum_{m \in CS_i \text{ s.t. } m \text{ is violated in } CS_i} \texttt{Cost}(\texttt{Act}(m))$ and from the fact that $\texttt{Cost} : \mathcal{S} \longmapsto I\!N^* \cup \{+\infty\}$, is strictly positive.

Proposition 2 $\forall m, m' \in \mathcal{M}$, if $\texttt{Act}(m) = \texttt{Act}(m')$ and $\texttt{PROP}(\{m\}) \equiv \texttt{PROP}(\{m'\})$, then $c(\{m\}) = c(\{m'\})$.

Proof. Since violation conditions are based on the existence of arguments and since the roles of agents are also syntax independent, we get:
For any commitment store CS_i,

- $\forall a \in \{\texttt{Assert}, \texttt{Declare}, \texttt{Question}, \texttt{Request}, \texttt{Challenge}, \texttt{Promise}\}$ and $\forall x, y \in \mathcal{L}$ such that $x \equiv y$,

  $a : x$ is violated in CS_i if and only if $a : y$ is violated in CS_i.

- for all argument (S, x) and (S', y) such that $S \equiv S'$,

  $\texttt{Argue} : (S, x)$ is violated in CS_i iff $\texttt{Argue} : (S', y)$ is violated in CS_i.

Proposition 3 If $\sum_{m \in A_i} \texttt{Penalty}(m) = 0$ then $\texttt{PROP}(CS_i)$ is consistent.

Proof. We reason by induction on the set A_i. If A_i is empty then this property is verified. We suppose this property verified for a given set A_i. And show that it is verified for $A_i \cup \{m\}, \forall m \in \mathcal{M}$ such that $\texttt{Act}(m) \in \{\texttt{Assert}, \texttt{Argue}, \texttt{Declare}\}$. For this purpose, we suppose that $c(A_i \cup \{m\}) = 0$ (it means, by definition, that $c(A_i) = 0$ and by induction hypothesis that $\texttt{PROP}(CS_i)$ is consistent) and we suppose that $\texttt{PROP}(CS_i \cup \{m\})$ is inconsistent. Let us consider every possible m:

$m = \texttt{Assert}{:}x$ if $\texttt{PROP}(CS_i \cup \{m\})$ is inconsistent then $\texttt{PROP}(CS_i) \cup \{x\}$ is inconsistent so $\texttt{PROP}(CS_i) \vdash \neg x$. Now, by hypothesis, $\texttt{PROP}(CS_i)$ is consistent, so $\exists (S, \neg x) \in Arg(\texttt{PROP}(CS_i))$. A fortiori, $(S, \neg x) \in Arg(\texttt{PROP}(CS_i \cup \{m\})$. Hence m is violated in $A_i \cup \{m\}$.

$m = \texttt{Argue}{:}(S, c)$ if $\texttt{PROP}(CS_i \cup \{m\})$ is inconsistent then $\texttt{PROP}(CS_i) \cup S$ is inconsistent so $\exists (S', \neg h) \in Arg(\texttt{PROP}(CS_i))$ where $h \in S$. A fortiori, $(S', \neg h) \in Arg(\texttt{PROP}(CS_i \cup \{m\})$. Hence S is undercut by S' so m is violated in $A_i \cup \{m\}$.

$m = \texttt{Declare}{:}x$ Since $\texttt{PROP}(CS_i)$ is consistent by hypothesis, and also is x, and since $\diamond$ is an update operator verifying Winslett or Katsuno and Mendelzon postulates, then by using postulate $MB4$ of Winslett (or its equivalent $U3$ in Katsuno and Mendelzon) we get that $\texttt{PROP}(CS_i) \diamond \{x\}$ should also be consistent. Hence the assumption that $\texttt{PROP}(CS_i \cup \{m\})$ is inconsistent is absurd.

Hence, we have proven that if m is an assertion or an argumentation then $\texttt{PROP}(CS_i \cup \{m\})$ is inconsistent implies that m is necessarily violated. Using proposition 1, we get that $c(A_i \cup \{m\}) \neq 0$ which is contradictory with our assumption. For the case where $m = \texttt{Declare}{:}x$, we show that $\texttt{PROP}(CS_i \cup \{m\})$ can not be inconsistent. Hence, in all cases, if $c(A_i \cup \{m\}) = 0$ then $\texttt{PROP}(CS_i \cup \{m\})$ is consistent. So, by induction the result is true for any set A_i. □

Lemma 1 Let $X \subseteq \mathcal{M}$ be a set of n moves and let m be a move uttered at step $s \leq n$ ($m = X^s$), let m' be a new move uttered at step $n+1$.
If m' is independent from m w.r.t. X and if $\texttt{Act}(m') \notin \{\texttt{Declare}, \texttt{Retract}\}$ then for all $x \in \texttt{Content}(m)$

$$\forall S \subseteq \mathcal{L}, \quad (S,x) \in \texttt{Arg}(\texttt{PROP}(X)) \Leftrightarrow (S,x) \in \texttt{Arg}(\texttt{PROP}(X \cup \{m'\}))$$
$$(S,\neg x) \in \texttt{Arg}(\texttt{PROP}(X)) \Leftrightarrow (S,\neg x) \in \texttt{Arg}(\texttt{PROP}(X \cup \{m'\}))$$

Proof. Since m' is neither a `Retract` nor a `Declare` move, then by definition of PROP, we have $\texttt{PROP}(X \cup \{m'\}) \subseteq \texttt{PROP}(X) \cup \texttt{PROP}(\{m'\})$. Let $x \in \texttt{Content}(m)$ then $\forall S \subseteq \mathcal{L}$, if $(S,x) \in \texttt{Arg}(\texttt{PROP}(X))$ then (S,x) is an argument built from a greater set, i.e., $(S,x) \in \texttt{Arg}(\texttt{PROP}(X \cup \{m'\}))$. Now if we suppose that $(S,x) \notin \texttt{Arg}(\texttt{PROP}(X))$ but that $(S,x) \in \texttt{Arg}(\texttt{PROP}(X \cup \{m'\}))$ it would mean that $\texttt{PROP}(\{m'\})$ is new for x w.r.t. $\texttt{PROP}(X)$. This contradicts the initial independence assumption. Hence $(S,x) \in \texttt{Arg}(\texttt{PROP}(X)) \Leftrightarrow (S,x) \in \texttt{Arg}(\texttt{PROP}(X \cup \{m'\}))$. The same reasonning holds for an argument $(S,\neg x)$.

Proposition 4 Let CS_i be a commitment store obtained after n steps, m be a move already uttered in CS_i, and let $m' \in \mathcal{M}$ be a move uttered at step $n+1$.

If m' is independent from m w.r.t. CS_i and if $\texttt{Act}(m') \notin \{\texttt{Declare}, \texttt{Retract}\}$ then m is violated in $CS_i \Leftrightarrow m$ is violated in $CS_i \cup \{m'\}$.

Proof. Note that for all m and for all m' whose speech act is not `Retract` we have $(\texttt{PROP}_P(CS_i)) \subseteq (\texttt{PROP}_P(CS_i \cup \{m'\}))$ (since m' is not a declarative move).

- ($\Rightarrow$) Let us suppose that m is violated in CS_i, then there are 7 possibilities for m (not 8 since the retract move can not be violated):
  $m = \texttt{Assert}{:}x$ it means that $\exists\, (S,\neg x) \in \texttt{Arg}(\texttt{PROP}(CS_i))$, hence, using Lemma 1, $(S,\neg x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}))$, so m is also violated in $CS_i \cup \{m'\}$.
  $m = \texttt{Argue}{:}(S,c)$ it means that $\exists (S',y) \in \texttt{Arg}(\texttt{PROP}(CS_i) \cup \{m\})$ such that (S',y) undercuts (S,c), this argument (S',y) belongs to a greater set $\texttt{Arg}(\texttt{PROP}(CS_i) \cup \{m'\})$, hence m is also violated in $CS_i \cup \{m'\}$.
  $m = \texttt{Declare}{:}x$ it means that either:
  - (1)$x \notin \texttt{Role}(ag_i)$
  - or (2) $\exists (S,\neg y) \in \texttt{Arg}(\texttt{Prop}(CS_i))$ with $y \in \texttt{Precond}(x)$.
  For the case (1) it implies that m is also violated in $CS_i \cup \{m'\}$ since the roles can not evolve during the dialog. For the case (2), $(S,\neg y)$ belongs also to the greater set $\texttt{Arg}(\texttt{Prop}(CS_i \cup \{m'\}))$. Hence, m is violated in $CS_i \cup \{m'\}$ in the two cases.
  $m = \texttt{Challenge}{:}x$ it means that $\nexists\ (S,x) \in \texttt{Arg}(\texttt{PROP}(CS_i))$ with $S \not\equiv x$. Using Lemma 1, we get $\nexists\ (S1,x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}))$. Hence, m is also violated in $CS_i \cup \{m'\}$.

$m = \texttt{Question}{:}x$ it means that (4) $\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i))$ and (5) $\nexists\ (S, \neg x) \in \texttt{Arg}(\texttt{PROP}(CS_i))$. Using Lemma 1 we get from (4) and (5) that $\nexists\ (S2, x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}))$ and $\nexists\ (S3, \neg x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}))$. So m is also violated in $CS_i \cup \{m'\}$.

$m = \texttt{Request}{:}x$ it means that $\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i) \cup \texttt{PROP}_P(CS_i)$ and $\nexists\ (S', \neg x) \in \texttt{Arg}(\texttt{PROP}(CS_i) \cup \texttt{PROP}_P(CS_i))$. Using Lemma 1, we get similarly as in the previous case that m is also violated in $CS_i \cup \{m'\}$

$m = \texttt{Promise}{:}x$ it means that $\nexists(S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i))$, using Lemma 1, we get that $\nexists(S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}))$ and thus that m is also violated in $CS_i \cup \{m'\}$.

– ($\Leftarrow$) Let us assume that m is violated in $CS_i \cup \{m'\}$, there are also 7 possibilities for m:

$m = \texttt{Assert}{:}x$ it means that $\exists\ (S, \neg x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}))$, if $(S, \neg x) \notin \texttt{Arg}(\texttt{PROP}(CS_i))$ then it would mean that $\texttt{PROP}(\{m'\})$ is new for x w.r.t. $\texttt{PROP}(CS_i)$ which contradicts the initial independence assumption. Hence, $(S, \neg x) \in \texttt{Arg}(\texttt{PROP}(CS_i))$, so m is also violated in CS_i.

$m = \texttt{Argue}{:}(S, c)$ it means that $\exists(S', y) \in \texttt{Arg}(\texttt{PROP}(CS_i) \cup \{m'\})$ such that (S', y) undercuts (S, c), the initial independence assumption implies that (S', y) should also belongs to $\texttt{Arg}(\texttt{PROP}(CS_i))$, so m is also violated in CS_i.

$m = \texttt{Declare}{:}x$ it means that either:
- $(1) x \notin \texttt{Role}(ag_i)$
- or (2) $\exists(S, \neg y) \in \texttt{Arg}(\texttt{Prop}(CS_i) \cup \{m'\})$ with $y \in \texttt{Precond}(x)$.
For the case (1) it implies that m is also violated in $CS_i \cup \{m\}$ since the roles can not evolve during the dialog. For the case (2), we use again the initial independence assumption between m and m' which implies that $(S, \neg y)$ belongs also to $\texttt{Arg}(\texttt{Prop}(CS_i))$. Hence, m is violated in CS_i in the two cases.

$m = \texttt{Challenge}{:}x$ it means that $\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}))$ with $S \not\equiv x$. This implies that it does not exist such an argument in a subset of this set, i.e., in $\texttt{Arg}(\texttt{PROP}(CS_i))$. So, m is also violated in CS_i.

$m = \texttt{Question}{:}x$ it means that $\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}))$ and $\nexists\ (S, \neg x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}))$. Hence it does not exist an argument for or against x in the smaller set $\texttt{Arg}(\texttt{PROP}(CS_i))$. So m is also violated in CS_i.

$m = \texttt{Request}{:}x$ it means that $\nexists\ (S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}) \cup \texttt{PROP}_P(CS_i \cup \{m'\}))$ and $\nexists\ (S', \neg x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}) \cup \texttt{PROP}_P(CS_i \cup \{m'\}))$. A similar proof as in the previous case gives that m is also violated in CS_i.

$m = \texttt{Promise}{:}x$ it means that $\nexists(S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i \cup \{m'\}))$ hence $\nexists(S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i))$. m is also violated in CS_i.

Proposition 5 Let CS_i be a commitment store obtained after n steps, m be a move already uttered in CS_i, and let $m' \in \mathcal{M}$ be a move uttered at step $n+1$.

If the update operator associated to `Declare` is independency compatible and if m' is independent from m w.r.t. CS_i and if $\texttt{Act}(m') \neq \texttt{Retract}$ then m is violated in $CS_i \Leftrightarrow m$ is violated in $CS_i \cup \{m'\}$.

Proof. The previous proof stated that for all m and m', if $\texttt{Act}(m') \notin \{\texttt{Declare}, \texttt{Retract}\}$ then $\texttt{PROP}(CS_i)) \subseteq (\texttt{PROP}(CS_i \cup \{m'\}))$.
Let us consider $m' = \texttt{Declare} : x'$, we have $\texttt{PROP}(CS_i \cup \{m'\}) = \texttt{PROP}(CS_i) \diamond x'$. If m is not a declare move, then the independence assumption implies that x' is independent of $\texttt{Content}(m)$ w.r.t. $\texttt{PROP}(CS_i)$, then, since $\diamond$ is independency compatible, we have: $(S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i) \Leftrightarrow (S, x) \in \texttt{Arg}(\texttt{PROP}(CS_i) \diamond x')$ and the same for $(S, \neg x)$. Hence the violation of m in CS_i is equivalent to the violation of m in $CS_i \cup \{m'\}$. Now, if $m = \texttt{Declare} : x$, then the independence assumption implies that x' is independent of $\texttt{Precond}(x)$ w.r.t. $\texttt{PROP}(CS_i)$, then, since $\diamond$ is independency compatible, x' is also independent of $\texttt{Precond}(x)$ w.r.t. $\texttt{PROP}(CS_i) \diamond x'$. Hence the result.