Dynamic Semantics for Agent Communication Languages

Michael Rovatsos
School of Informatics
The University of Edinburgh
Edinburgh EH8 9LE
United Kingdom
mrovatso@inf.ed.ac.uk

ABSTRACT

This paper proposes dynamic semantics for agent communication languages (ACLs) as a method for tackling some of the fundamental problems associated with agent communication in open multiagent systems. Based on the idea of providing alternative semantic "variants" for speech acts and transition rules between them that are contingent on previous agent behaviour, our framework provides an improved notion of grounding semantics in ongoing interaction, a simple mechanism for distinguishing between *compliant* and *expected* behaviour, and a way to specify sanction and reward mechanisms as part of the ACL itself. We extend a common framework for commitment-based ACL semantics to obtain these properties, discuss desiderata for the design of concrete dynamic semantics together with examples, and analyse their properties.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*

Keywords

Agent communication languages, social reasoning

1. INTRODUCTION

The field of agent communication language (ACL) research has long been plagued by problems of verifiability and grounding [10, 13, 17]. Early mentalistic semantics that specify the semantics of speech acts in terms of pre- and post-conditions contingent on mental states of the participants (e.g. [3, 4, 12, 15]) lack verifiability regarding compliance of agents with the intended semantics (as the mental states of agents cannot be observed in *open* multiagent systems (MASs)). Unable to safeguard themselves against abuse by malicious, deceptive or malfunctioning agents, mentalistic semantics are inherently unreliable and inappropriate for use in open MAS in which agents with potentially conflicting objectives might deliberately exploit their adversaries' conceptions of message semantics to provoke a certain behaviour.

Commitment-based semantics [6, 8, 14], on the other hand, define the meaning of messages exchanged among agents in terms of publicly observable commitments, i.e. pledges to bring about a state of affairs or to perform certain actions. Such semantics solve the verifiability problem as they allow for tracing the status of existing commitments at any point in time given observed messages and actions so that any observer can, for example, establish whether an agent has performed a promised action. However, this can only be done *a posteriori*, and this creates a grounding problem as no expectations regarding what will happen in the future can be formed at the time of uttering or receiving a message purely on the grounds of the ACL semantics.

Further, this implies that the semantics specification does not provide an interface to agents' deliberation and planning mechanisms and hence it is unclear how rational agents would be able to decide whether to subscribe to a suggested ACL semantics when it is deployed.

Finally, none of the existing approaches allows the ACL to specify how to *respond* to a violation of its semantics by individual agents. This has two implications: Firstly, it is left it up to the individual agent to reason about potential violations, i.e. to bear the burden of planning its own reaction to others' non-compliant behaviour (e.g. in order to sanction them) and to anticipate others' reactions to own misconduct without any guidance from the ACL specification. Secondly, existing approaches fail to exploit the possibilities of sanctioning and rewarding certain behaviours in a *communication-inherent* way by modifying the future meaning of messages uttered or received by compliant/deviant agents.

In this paper, we propose *dynamic semantics* (DSs) for ACLs as a solution to these problems. Our notion of DS is based on the very simple idea of defining different alternatives for the meaning of individual speech acts (so-called *semantic variants*) in an ACL semantics specification, and transition rules between *semantic states* (i.e. collections of variants for different speech acts) that describe the current meaning of the ACL. These elements taken together result in a FSM-like view of ACL specifications where each individual state provides a complete ACL semantics and state transitions are triggered by observed agent behaviour in order to (1) reflect future expectations based on previous interaction experience and (2) sanction or reward certain kinds of behaviour.

In defining a DS framework for commitment-based ACLs, this paper makes three contributions:

1. An extension of commitment-based ACL semantics to provide an improved notion of *grounding* commitments in agent interaction and to allow ACL specifications to be directly used for planning-based rational decision making.

2. A simple way of distinguishing between *compliant* and *expected* behaviour with respect to an ACL specification that enables reasoning about the potential behaviour of agents purely from an ACL semantics perspective.

3. A mechanism for specifying how meaning *evolves* with agent behaviour and how this can be used to describe *communication-inherent* sanctioning and rewarding mechanisms essential to the design of open MASs.

Furthermore, we discuss desiderata for DS design that can be derived from our framework, present examples and analyse their properties.

The remainder of this paper is structured as follows: Section 2 introduces a formal framework for dynamic ACL semantics. In section 3 we present an analysis and discussion of this framework and discuss desiderata for the design of ACLs with dynamic semantics. Section 4 reviews related approaches, and section 5 concludes.

2. FORMAL FRAMEWORK

Our general framework for describing the kind of MASs we are interested in is fairly simple. Let $Ag = \{1, \ldots, n\}$ a finite set of agents, $\{Ac_i\}_{i \in Ag}$ a collection of action sets (where Ac_i are the actions of agent i), $\mathcal{A} = \times_{i=1}^{n} Ac_i$ the joint action space, and Env a set of environment states. A *run* is a sequence $r = e_1 \overset{\vec{a}_1}{\rightarrow} \ldots \overset{\vec{a}_{t-1}}{\rightarrow} e_t$ where $\vec{a}_i \in \mathcal{A}$ ($\vec{a}_i[j]$ denotes the action of agent j in this tuple), and $e_i \in Env$. We define $|r| = t$, $last(r) = e_t$, $r[1:j]$ is short for the j-long initial sub-sequence of r, and we write $r' \sqsubseteq r$ for any run r' iff $\exists j \in \mathbb{N}. r' = r[1:j]$.

Writing $\mathcal{R}(Env, \mathcal{A})$ for the set of all possible runs, we can view each agent i as a function $g_i : \mathcal{R}(Env, \mathcal{A}) \rightarrow Ac_i$ describing the agent's action choices depending on the history of previous environment states and joint actions. The set of all agent functions for i given $\mathcal{A}$ and Env is denoted by $G_i(Env, \mathcal{A})$. The (finite, discrete, stationary, fully accessible, deterministic) environment is defined by a state transformer function $f : Env \times \mathcal{A} \rightarrow Env$, so that the system's operation for an initial state e_1 is defined by $e_{i+1} = f(e_i, \vec{g}(e_1 \overset{\vec{a}_1}{\rightarrow} \ldots \overset{\vec{a}_{i-1}}{\rightarrow} e_i))$ for all $i \geq 1$ ($\vec{g}$ is the joint vector of functions g_i). This definition implies that execution of actions is synchronised among agents, so that the system evolves though an execution of "rounds" where all agents perform their actions simultaneously.

We denote the set of all runs given a particular configuration of agent functions $\vec{g}$ by $\mathcal{R}(Env, \mathcal{A}, \vec{g})$. We write $g_i \sim r$ where g_i an agent function and r a run iff $\forall 1 \leq j \leq |r|. g_i(r[1:j]) = \vec{a}_j[i]$ (i.e. g_i is compatible with r in every time step as far as i's actions are concerned).

We use a (standard) propositional logical language $\mathcal{L}$ with entailment relation $e \models \varphi$ for $e \in Env$ and $\varphi \in \mathcal{L}$ defined in the usual way.[1] We introduce special propositions $Done(i, a)$ for each action $a \in \cup_{i=1}^{n} Ac_i$ in $\mathcal{L}$ to denote "it is true that action a has just been performed", extending $\models$ to runs r in the following way:

$$r \models \varphi \qquad \text{if } last(r) \models \varphi$$

$$r \models Done(i, a) \qquad \text{if } r = e_1 \overset{\vec{a}_1}{\rightarrow} \ldots \overset{\vec{a}_{t-1}}{\rightarrow} e_t \wedge a = \vec{a}_{t-1}[i]$$

i.e. $Done(i, a)$ is exactly true for those actions that made up part of the joint action vector $\vec{a}_{i-1}$ in the predecessor state, and all other formulae that were entailed by the last state of r are still valid. Our model implies that each agent executes exactly one action in each time step.

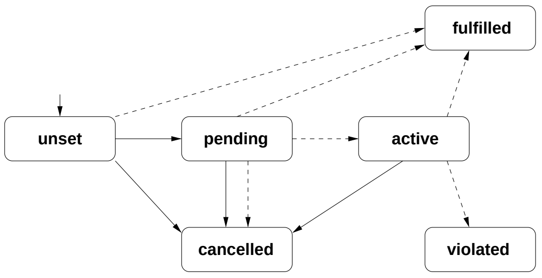

Figure 1: Commitment states and state transitions in the Fornara and Colombetti model: edges drawn using solid lines indicate transitions brought about by agent communication, dashed lines indicate physical agent action or environmental events that cause state transitions

2.1 Commitments

Our notion of *commitments* is based on a slight variation of the framework proposed by Fornara and Colombetti [6]: Commitments come into existence as "unset", e.g. when a request for achieving χ if a certain condition φ becomes true is issued from i to j. The commitment becomes "pending" if the debtor j is required to fulfill it, e.g. after having accepted it. A pending commitment will become "active" if its condition φ becomes true, and if χ is brought about in that case it becomes "fulfilled", otherwise "violated". Commitments can become "cancelled" in different situations, e.g. if an unset commitment is rejected. Also, environmental events can lead χ to become true in which case the commitment becomes fulfilled without the debtor's contribution. Figure 1 provides a graphic representation of commitment state transitions in this framework.

Apart from a slightly different notation used to maintain a more detailed history of commitments, we will extend them to also contain a *deactivation condition* ψ apart from φ (which we call *activation condition*) which causes any commitment to be cancelled if it becomes true.

[1]More precisely $\mathcal{L}$ contains atomic propositions $P = \{p, q, \ldots\}$, the usual connectives $\vee$ and $\neg$ (with abbreviations $\Rightarrow$ and $\wedge$). As for semantics, a function interpretation function $I : P \times Env \rightarrow \{\top, \bot\}$ assigns a truth value to each proposition in each environmental state, and the entailment relation $e \models \varphi$ for $e \in Env$ and $\varphi \in \mathcal{L}$ is defined inductively: $e \models \varphi$ if $\varphi \in P$ and $I(\varphi, e) = \top$; $e \models \neg\varphi$ if $e \not\models \varphi$; $e \models \varphi \vee \psi$ if $e \models \varphi$ or $e \models \psi$.

$$
\begin{array}{ll}
D: & CS \leftarrow CS \cup \{\langle \iota, \mathbf{c} : \chi \oplus \varphi \ominus \psi\rangle_t \mid \langle \iota, \mathbf{s} : \chi \oplus \varphi \ominus \psi\rangle \in CS, r \models \psi, \mathbf{s} \in \{u,p,a\}, \langle \iota, \mathbf{c} : \chi \oplus \varphi \ominus \psi\rangle \notin CS\} \\
A: & CS \leftarrow CS \cup \{\langle \iota, \mathbf{a} : \chi \oplus \varphi \ominus \psi\rangle_t \mid \langle \iota, \mathbf{p} : \chi \oplus \varphi \ominus \psi\rangle \in CS, r \models \varphi, \langle \iota, \mathbf{a} : \chi \oplus \varphi \ominus \psi\rangle \notin CS\} \\
S: & CS \leftarrow CS \cup \{\langle \iota, \mathbf{f} : \chi \oplus \varphi \ominus \psi\rangle_t \mid \langle \iota, \mathbf{a} : \chi \oplus \varphi \ominus \psi\rangle \in CS, r \models \chi, \langle \iota, \mathbf{f} : \chi \oplus \varphi \ominus \psi\rangle \notin CS\} \\
F: & CS \leftarrow CS \cup \{\langle \iota, \mathbf{f} : \chi \oplus \varphi \ominus \psi\rangle_t^{i \to j} \mid \langle \iota, \mathbf{a} : \chi \oplus \varphi \ominus \psi\rangle_{t-1}^{i \to j} \in CS, r \models Done(i,a), causes(a,\chi)\} \\
V: & CS \leftarrow CS \cup \{\langle \iota, \mathbf{v} : \chi \oplus \varphi \ominus \psi\rangle_t^{i \to j} \mid \langle \iota, \mathbf{a} : \chi \oplus \varphi \ominus \psi\rangle_{t-1}^{i \to j} \in CS, r \models Done(i,a), \neg causes(a,\chi)\}
\end{array}
$$

Table 1: Environmental commitment processing rules for current run r with $|r| = t$

Definition 1. A *commitment* is a structure

$$\langle \iota, \mathbf{s} : \chi \oplus \varphi \ominus \psi\rangle_t^{i \to j}$$

where

- ι is a unique *commitment identifier*,
- $\mathbf{s}$ denotes the commitment *state* (any of **u**nset, **p**ending, **a**ctive, **v**iolated, **f**ulfilled, or **c**ancelled, abbreviated by the respective initial),
- i is the *debtor*, j is the *creditor*,
- $\chi \in \mathcal{L}$ is the *debitum* (i.e. the proposition that i commits to making true in front of j),
- $\varphi, \psi \in \mathcal{L}$ are the *activation/deactivation conditions*,
- and t is the instant (in a run) at which this commitment entered its current state $\mathbf{s}$.

As an example,

$$\langle x, \mathbf{v} : \mathit{received}(5, \$500) \oplus \mathit{received}(3, \mathit{toys}) \ominus \mathit{returned}(3, \mathit{toys})\rangle_{12}^{3 \to 5}$$

denotes that agent 3 violated commitment x towards agent 5 to pay him \$500 in timestep 12. He was supposed to make the payment after receiving the toys unless he sent back the toys. We introduce deactivation conditions so as to be able to completely revoke existing commitments: Sending back the money does not constitute a fulfillment of the original contract, but instead an annulment thereof. This provides us with the capability to define "validity conditions" using φ and ψ, which is useful for things like deadlines for unset commitments ("if I don't get a response within 3 time-steps my request will expire").

For brevity, we sometimes omit indices or content elements when clear from the context (in particular, we often write Γ for the *content* $\chi \oplus \varphi \ominus \psi$). We write $\mathcal{C}$ for the set of all possible commitments and denote sets of commitments (so-called *commitment stores*) by $CS \in \wp_{fin}(\mathcal{C})$.

To handle the effects of environmental events and agent actions on a commitment store CS, table 1 introduces five *commitment transition rules* which are executed in each time step by the system or any observer who intends to clarify the status of existing commitments in the order shown: the *deactivation* rule D is the first to fire and cancels any unset, pending or active commitments if ψ becomes true. For the remaining pending commitments[2], the *activation* rule A describes how these become active if φ becomes true. Note that when φ is true in subsequent states we check whether this active commitment is contained in CS to avoid duplicates (this is because we keep a full record of the commitment history for reasons which will become clear below).[3] Rule S caters for "serendipity" i.e. fulfillment of commitments not brought about by the respective agent, but simply by environmental changes that made the debitum true. Finally, the *fulfilment/violation* rules F/V record whether the action performed by the debtor in the previous step ($r \models Done(i,a)$) has caused the debitum χ of any commitment which became active in the previous timestep to become true. We need only consider those commitments that became active in the previous step $t-1$ since we can verify their fulfilment status in t. This verification hinges on a domain-dependent predicate $causes(a,\chi)$ which we have not mentioned so far. It should be true if action a is supposed to bring about χ, and delineates the existing social notion of what constitutes a "reasonable attempt" to achieve χ in the given context (its definition may range from requiring that χ has actually been achieved to allowing any action a that does not necessarily result in $\neg\chi$).

2.2 Grounding

In Fornara and Colombetti's and similar approaches, the status of commitments is verifiable, but they are not *grounded* in expectations about interaction. Such semantics (similar in style to what he have just defined in terms of CS update rules) tell us what commitments exist and which state they are in, but not how this will affect future agent behaviour.

To provide such grounding, we introduce notions of *compliant* and *expected behaviour*. An agent is behaving in compliance with its commitments if it always immediately fulfills all active commitments. More precisely, the behaviour of agent i is said to be *compliant with CS* at time t iff

$$\forall k \le t \left(\langle \iota, \mathbf{a} : \Gamma\rangle_k^{i \to j} \in CS \Rightarrow \langle \iota, \mathbf{f} : \Gamma\rangle_k^{i \to j} \in CS\right)$$

Though simple, this definition of compliance is not very useful because it places constraints on CSs but not on actual agent functions. To achieve this, we can instead use the contents of the CS to restrict the range of admissible agent functions to those that are in accordance with it using the following definition:

Definition 2. For any run $r \in \mathcal{R}(Env, \mathcal{A})$, let $CS(r)$ the set of commitments that has resulted from execution of r assuming that certain actions (including messages) create commitments or change their status. The set of *compliant agent functions* with respect to a commitment store CS is

[2] To avoid problems with contradictory commitment specifications (e.g. when both φ and ψ become true), we give deactivation strict precedence over activation.

[3] While commitment identifiers adversely affect the readability of our notation, they are necessary here to uniquely determine which pending commitment is activated.

defined as

$$\begin{aligned}
\mathit{compliant}(CS) := \big\{ g_i \in G_i(\mathit{Env}, \mathcal{A}) \big| \\
\forall r \sim g_i . \langle \iota, \mathbf{p} : \chi \oplus \varphi \ominus \psi \rangle^{i \to j} \in CS(r) = CS . \\
\forall r' \sqsupseteq r . \langle \iota, \mathbf{a} : \chi \oplus \varphi \ominus \psi \rangle^{i \to j}_{|r'|} \in CS(r') \Rightarrow \\
\big(\exists a \in Ac_i . \mathit{causes}(a, \chi) \wedge g_i(r') = a \big) \ \big\}
\end{aligned}$$

What this definition captures is the following characterisation of a compliant agent function g_i: "for all runs r that the agent function g_i contributes to: if r has created a pending commitment regarding χ, then if this commitment becomes active at the end of some extension r' of r in the future, g_i will cause the agent to perform an action a that causes χ".[4]

Next, to cater for the anticipation of non-compliant behaviour we need to introduce a notion of "expected" behaviour that overrides compliant behaviour. For this, we introduce a second type of commitments which we will call *expectations* to avoid confusion and distinguish from ordinary (now called *normative*) commitments by using round brackets $(\iota, \mathbf{s} : \Gamma)^{i \to j}_t$. They are treated exactly like other commitments in terms of the rules introduced above but express what the agent is *expected* to do (in the non-normative sense of an objective prediction of behaviour) rather than what it is *supposed* to do in a normative sense.

To define the notions we need below, we introduce the following constructs:

$$\begin{aligned}
\lceil CS \rceil := \quad & \{ \langle \iota, \mathbf{s} : \Gamma \rangle \in CS | \mathbf{s} \in \{ \mathbf{u}, \mathbf{p}, \mathbf{a}, \mathbf{f}, \mathbf{v} \} \} \\
\lfloor CS \rfloor := \quad & \{ (\iota, \mathbf{s} : \Gamma) \in CS | (\iota, \mathbf{s} : \Gamma) \in CS, \\
& \quad \langle \iota, \mathbf{s}' : \Gamma \rangle \in CS, \mathbf{s}, \mathbf{s}' \in \{ \mathbf{u}, \mathbf{p}, \mathbf{a}, \mathbf{f}, \mathbf{v} \} \}
\end{aligned}$$

$\lceil CS \rceil$ simply restricts the commitment store to all normative commitments. Hence, $\mathit{compliant}(\lceil CS \rceil)$ specifies what agents are supposed to do. $\lfloor CS \rfloor$, on the other hand, overrides all normative commitment elements in CS for which an expectation also exists, i.e. expectations are given precedence over the normative commitments. With this, we can define expected behaviour as

$$\mathit{expected}(CS) := \mathit{compliant}(\lfloor CS \rfloor)$$

i.e. behaviour that adheres to expectations where such expectations exist and is compliant otherwise. The separate, parallel, treatment of compliant and expected behaviour has two advantages: Firstly, we can respond to "unexpected" compliant behaviour, i.e. when we expect that someone will not obey their commitments we can still respond to it if they do (and, for example, regain trust in them). Secondly, we can cater for a variety of rules for translating commitment stores to actual future events which a reasoning agent can use in its planning process. For the purposes of this paper, we will assume that agents base their predictions about others on expected behaviour if it is different from compliant behaviour, and that they predict compliant behaviour, else.

[4]Note the quantification in this definition: the property has to hold for *every* run that gave rise to ι and is compatible with g_i. In particular, this must be independent of any part of the history (e.g. other agents' actions and previous environment states) given $CS(r)$. We also quantify over all extensions r' of r, i.e. fulfillment of the commitment has to happen if the appropriate conditions arise *regardless* of other factors.

2.3 Static ACL Semantics

Table 2 shows an example for a small fragment of an ACL semantics defined using our framework, with two alternative definitions (AC and $AC2$) for the semantics of the accept message type. Each of the so-called *dialogue operators* (similar to AI planning action schemata) is defined using the graphical notation

$$\begin{array}{c} \hline p \\ \boxed{a} \\ q \\ \hline \end{array}$$

where p, a, and q are schemata for preconditions, messages (of a certain type), and post-conditions, respectively. Preconditions determine whether an action schema is applicable in a certain situation or not and contain formulae from $\mathcal{L}$ and/or constraints on the current contents of CS. Post-Conditions contain changes to the knowledge base and modifications to CS, i.e. they are interpreted like add/delete-lists in traditional AI planning. For any such operator $o = \langle p, a, q \rangle$ we define $\mathit{pre}(o) = p$, $\mathit{action}(o) = a$ and $\mathit{post}(o) = q$. All elements of a dialogue operator can contain logical variables in their pre- and post-conditions and sender/receiver/content variables in the action slot.

In our example fragment, the operator RQ for requests creates an unset commitment with a fresh identifier ι and current timestamp (we assume that $r \models \mathit{time}(t) \Leftrightarrow |r| = t$, and there is a global system time that can be inspected by all agents), and AC/RJ add a pending/cancelled equivalent of ι to CS. A fragment consisting of $\{RQ, RJ, AC\}$ is equivalent to the standard semantics of the respective performative types defined in [6].[5] Note that our operators only contain objectively verifiable pre- and post-conditions, and if agents want to conform to it they need to comply with these operators. In the following, we will assume that agents always adhere to the ACL specification *syntactically*[6].

Using $AC2$ instead of AC enables us to exploit the power of our distinction between compliant and expected behaviour, expressing that we don't trust i to adhere to the "normal" semantics of `accept`: its postcondition specifies that $\mathit{expected}(CS)$ is not restricted to behaviours that will fulfill the commitment but suggest that it has actually been cancelled. At the same time, we maintain the normative commitment that ι *is* pending so that i's behaviour would be seen to lie within $\mathit{compliant}(CS)$ if i deviates from our (pessimistic) expectation and does the "right" thing instead.

2.4 Dynamic Semantics

2.4.1 Defining Dynamic Semantics

To define DS for ACLs we now introduce a state transition system in which each state specifies an "ordinary" (static) commitment-based semantics and a "range" of agent pairs for which these semantics are assumed to apply.

[5]Note that we allow for requesting identical things before receiving a response and responding several times to the same request. Simple additional conditions can be introduced to avoid these effects which we omit here for lack of space. The same is true of additional constraints to manage control flow issues in actual dialogues (e.g. turn-taking).

[6]This means that, for an appropriate variable substitution ϑ, $r \models \mathit{pre}(o)\vartheta$ holds when o is applied at r and that $CS(r)$ is transformed according to $\mathit{post}(o)\vartheta$ after its application.

$$RQ: \quad \frac{time(t), new(\iota)}{\boxed{\texttt{request}(i,j,\iota:\Gamma)}} \quad \frac{}{CS \leftarrow CS \cup \{\langle \iota, \mathbf{u} : \Gamma\rangle_t^{i \to j}\}}$$

$$RJ: \quad \frac{\langle \iota, \mathbf{u} : \Gamma\rangle_{t'}^{j \to i} \in CS, time(t)}{\boxed{\texttt{reject}(i,j,\iota:\Gamma)}} \quad \frac{}{CS \leftarrow CS \cup \{\langle \iota, \mathbf{c} : \Gamma\rangle_t^{i \to j}\}}$$

$$AC: \quad \frac{\langle \iota, \mathbf{u} : \Gamma\rangle_{t'}^{j \to i} \in CS, time(t)}{\boxed{\texttt{accept}(i,j,\iota:\Gamma)}} \quad \frac{}{CS \leftarrow CS \cup \{\langle \iota, \mathbf{p} : \Gamma\rangle_t^{i \to j}\}}$$

$$AC2: \quad \frac{\langle \iota, \mathbf{u} : \Gamma\rangle_{t'}^{j \to i} \in CS, time(t)}{\boxed{\texttt{accept}(i,j,\iota:\Gamma)}} \quad \frac{}{CS \leftarrow CS \cup \{\langle \iota, \mathbf{p} : \Gamma\rangle_t^{i \to j}\} \cup \{\langle \iota, \mathbf{c} : \Gamma\rangle_t^{i \to j}\}}$$

Table 2: Example commitment-based semantics for a small ACL fragment

$$\begin{tikzcd} s_0 \arrow[r, bend left, "{\langle \iota, \mathbf{v} : \Gamma\rangle^{i \to j} \in CS :\ \{(i,*)\} \cup \{(j,i)\}}"] & s_1 \arrow[l, bend left, "{\forall \langle \iota, \mathbf{v} : \Gamma\rangle_t^{i \to j} \in CS\ \exists \langle \iota, \mathbf{f} : \Gamma'\rangle_{t'}^{i \to j'} \in CS . t' > t :\ \{(i,*)\}}"] \end{tikzcd}$$

Figure 2: FSM-like state transition diagram describing the Δ-relation in a DS specification

Definition 3. A *dynamic semantics* (DS) is a structure $\langle O, S, s_0, \Delta\rangle$ where

- $O = \{o_1, o_2, \ldots, o_n\}$ a set of dialogue operators,
- $S \subseteq \wp(O)$ is a set of *semantic states* specified as subsets of dialogue operators which are valid in this state,
- $s_0 \in S$ is the initial semantic state,
- and the *transition relation* $\Delta \subseteq S \times \wp(\mathcal{C}) \times \wp(Ag \times Ag) \times S$ defines the transitions over S triggered by conditions expressed as elements of $\wp(\mathcal{C})$ ($\mathcal{C}$ is the set of all possible commitments).

The meaning of a transition $(s, c, \{(i_1, j_1), \ldots, (i_n, j_n)\}, s') \in \Delta$ is as follows: Assume a mapping $act : Ag \times Ag \to S$ which specifies that the semantics of operators in s holds for messages sent from i to j . Then, if $CS \in c$ (i.e. the current CS matches the constraint c given as a collection of possible CSs) this will trigger a transition to state s' for all pairs of agents in $\{(i_1, j_1), \ldots, (i_n, j_n)\}$ for which the constraint was satisfied and will update act accordingly. In other words, the act mapping tracks which "version" of the semantics is valid for which pairs of communication partners over time.

2.4.2 Example

To illustrate these concepts, consider the following example: Let $O = \{RQ, RJ, AC, AC2\}$, $S = \{s_0, s_1\}$ where $s_0 = \{RQ, RJ, AC\}$ and $s_1 = \{RQ, RJ, AC2\}$, i.e. there are two possible states of the semantics which only differ in their definition of `accept` (we call alternative versions of a single dialogue operator like AC and $AC2$ *semantic variants*). We assume that initially $act(i, j) = s_0$ for all agents $i, j \in Ag$.

We describe δ by the transition diagram shown in figure 2. In this diagram, edges carry labels "$c : A$" where c is a constraint on the contents of CS followed by a description of the set of agent pairs A for which the transition should be made to the target state. Writing $A(s) = act^{-1}(s)$ for the so-called *range* of agent pairs for which s is active, we use agent variables like i and j and the wildcard symbol $*$ that can be bound to any agent in $A(s)$, and we assume that this binding carries over to descriptions of A. For example, the edge with label "$\langle \iota, \mathbf{v} : \Gamma\rangle^{i \to j} \in CS :\ \{(i,*)\} \cup \{(j,i)\}$" can be interpreted as follows: "select all pairs $(i, j) \in A(s_0)$ for which $\langle \iota, \mathbf{v} : \Gamma\rangle^{i \to j} \in CS$ applies (i.e. i has violated some commitment toward j) and make s_1 valid for the set of agents $\{(i, k) | k \in A(s_0)\} \cup \{(j, i)\}$". This means that *for all* agents i who have lied, s_1 will become active for (i, j') where $j' \in A(s_0)$ and s_1 will also become active for (j, i).

The way the DS of the diagram above works is as follows: initially the semantics says (for every agent i) that they will fulfill any commitment truthfully (the use of AC ensures that expected behaviour is equivalent to compliant behaviour). If an agent i violates a commitment once then s_1 will become active for i towards all other agents, so that they won't expect i to fulfill any future commitments. Moreover, this will also apply to (j, i) so that the culprit i should not expect the deceived agent j to keep its promises towards i either in the future. However, this will not affect expectations regarding their interactions with i by agents other than i (i.e. they still have no right to violate their own commitments). This reflects the idea that (only) agents that have been fooled are allowed to "trespass" (only) against those agents who "trespassed" against them. However, if i ever fulfills any commitment again (after the latest violation, this is ensured by the complex constraint used as a label for the transition from s_1 to s_0), the semantics in s_0 will become valid for i again. In this case, though, s_1 will still be valid for the pair (j, i), i.e. agent j will regain trust in i but cannot be expected to be trustworthy toward i ever again.

Rather than suggesting that this is a particularly useful *communication-inherent* mechanism for sanctioning and rewarding specific kinds of behaviour, this example serves to illustrate the expressiveness of our framework and the kind of distinctions it enables us to make.

2.4.3 Formal Semantics

The semantics of a DS can be defined inductively as follows: Let $CS(r)$ denote the contents of the commitment store after run r as before. We use the notation

$$A(\delta, CS) = \{(i, j) | CS|_{i,j} \in c\} \cap A(s) \cap A$$

to denote the set of agents that are to be "moved" from s to s' due to transition rule $\delta = (s, c, A, s') \in \Delta$ given CS, where $CS|_{i,j}$ is the set of commitments that mention i and/or j (in their sender/receiver/content slots). In other words, $A(\delta, CS)$ contains those pairs of agents who are (i) mentioned in the commitments covered by the constraint c, (ii) contained in the range of s, and (iii) explicitly listed in A as belonging to those pairs of agents that should be affected by the transition δ.

Definition 4. The *state* of a dynamic semantics $\langle O, S, s_0, \Delta\rangle$ after run r with immediate predecessor r' is defined as a mapping act_r as follows:

1. $r = \varepsilon$: $act_\varepsilon(i,j) = s_0$ for all $i, j \in Ag$
2. $r \neq \varepsilon$:

$$act_r(i,j) = \begin{cases} s' & \text{if } \exists \delta = (s, c, A, s') \in \Delta. \\ & \quad (i,j) \in A(\delta, CS(r)) \\ act_{r'}(i,j) & \text{else} \end{cases}$$

This maintains the property $act_{r'}^{-1}(s) = act_r^{-1}(s) - A(\delta, CS(r'))$, which specifies that the agent pairs to be "moved" from s to s' are removed from the range of s and added to the range of s'.

What is not ensured by this definition is consistency of the state transition system, i.e. making sure that the semantic successor state is uniquely identified for any state of the commitment store and previous state so that every agent pair is only assigned one active state in each step, i.e. act_r is actually a function for any r.[7]

2.4.4 Integration

Once the DS itself has been specified, we need to integrate the different components of our framework to monitor the dynamics of our ACL semantics and its implications for expected agent behaviour.

Starting with an initially empty commitment store CS and initial semantic state s_0 such that $act_\varepsilon(i,j) = s_0$ for any two agents i and j, the agent (or external observer) observes (a partial subset of) everything that is communicated in the system in each step. By applying the commitment transition rules (D, A, S, F and V) we can update CS accordingly, ignoring any observed message sent from i to j that does not syntactically match the dialogue operator set defined in $act_r(i,j)$ for a current run r. After this update has been performed for all observed messages and actions in this cycle, which should not depend on the ordering of messages[8], we can compute for any message sent from i to j the new value of $act_{r'}(i,j)$ depending on the semantic transition rules of the DS if r' is the successor run of r. With this, we can then determine what the compliant and expected behaviour of agents will be under these new conditions.

Thus, an agent can use information about expected behaviour in its own planning processes by assuming that all agents involved will exhibit their expected (rather than just compliant) behaviours. This prediction will not always be more accurate than under normal (static) ACL semantics, but since it is common knowledge that agents assume expected behaviour to occur (and, by virtue of the DS-ACL specification, have the "right" to do that) most reasonable dynamic ACL specifications will make provisions to ensure that it is "safer" to assume expected rather than fully compliant behaviour if they want to promote their use by agents.

[7]One way of ensuring this is to require that $\forall s \in S. (\cap\{c|(s, c, A, s') \in \Delta(s)\} = \emptyset)$ so that no two constraints pertaining to outgoing edges of s can be fulfilled by CS at a time. In some cases this may be too coarse-grained – it would be sufficient for constraints to be mutually exclusive for the same pair of agents at any point in time – but this would have to be verified for an individual DS on a case-by-case basis.

[8]This is the case for our operators, because their pre- and post-conditions never concern or affect any commitments other than those that involve *both* i and j – avoiding any connection to third parties helps us keep the CS-update independent of the order in which observations are processed.

2.4.5 Complexity Issues

The main disadvantage of our approach is the space complexity of the dynamic ACL specification: If d is the number of dialogue operators in a language and b is the maximum number of semantic variants of a single dialogue operator within this language, the DS specification would have to specify $O(d^b)$ states. In many cases, however, most of the speech acts will not have different variants (like RQ and RJ in our example) and this may significantly reduce the number of DS states that need to be specified.

As for the run-time behaviour of our semantics processing mechanism, we can assume that n messages/actions are sent/performed in each processing step in a system with n agents. Every commitment processing rule (D, S, etc.) has to perform a pass over the contents of CS. In the worst case every originally created commitment (of which there may be nt after t steps) may have immediately become pending, active and violated (which doesn't require any further physical actions, so that every agent can create a new commitment in each step).Thus, if any agent creates a new commitment in each step without ever fulfilling it, this will result in the total size of CS being in $O(nt)$.[9]

Regarding semantic state transitions, as many as n different pairs of agents could be affected in a single iteration by n messages. Assuming that the verification of CS-constraints for these transitions would take $O(nt)$, this yields a total update time of $O(n^2t)$ for tracking DS evolution. This bound can be reduced to $O(n^2)$ if a "quasi-stationarity" assumption is made by limiting the "window" of earlier commitments that are being considered when verifying transition constraints to a constant size (and thus obtaining a finite set of possible commitment stores).[10]

3. ANALYSIS AND DISCUSSION

The main strength of our framework is that it allows us to exploit the three main elements of *reciprocity*:

- *Reputation-based adaptation:* The DS adapts the expectations toward agent i according to i's previous behaviour by modifying the semantic state to better reflect this behaviour (based on the assumption that it will repeat itself in the future).
- *Mutuality of expectations:* The DS adapts the expectations toward j's behaviour according to i's previous behaviour toward j to better reflect j's response to i's observed behaviour (in particular, allowing j to behave toward i as i behaved toward j earlier).
- *Recovery mechanisms:* The DS allows i to revert to an earlier semantic state after having undone a change in expectations by a further, later change of behaviour (e.g. by means of "redemption").

In open systems in which we cannot enforce certain behaviours, these are effectively the only available means for indirect sanctions and rewards.

[9]This is actually only a lower bound on the complexity for commitment processing which could become even worse if dominated by the complexity of verifying entailment $\models$; however, this would also hold for a "static" ACL semantics.

[10]For example, this could be useful if we want to discard commitments whose status was last modified more than k time steps ago (this is problematic, as it might force us to discard certain unset/pending commitments before they become pending/active).

There are two further dimensions that affect DS-based sanctioning and reward mechanisms and are orthogonal to the above: One concerns the *character* of the semantic state changes (i.e. whether it is a reward or punishment), the other the *degree* of adaptation (reputation-based mechanisms, for example, need not realistically reflect the behaviour of the culprit, but may instead utilise immediate (exaggerated) "stigmatisation" of agents as a deterrent).

Albeit simple, our example DS described above makes use of all these aspects, and apart from consistency and completeness, it also satisfies some other useful properties:

1. Non-redundancy: No two dialogue operators in $\mathcal{O}$ should have identical pre- and post-conditions, and any two semantic variants of an operator must differ in terms of pre- and/or post-conditions:

$$\forall o, o' \in \mathcal{O} .(pre(o) = pre(o') \wedge post(o) = post(o') \Rightarrow o = o')$$

$$\forall o, o' \in O .(action(o) = action(o') \Rightarrow$$
$$pre(o) \neq pre(o') \vee post(o) \neq post(o))$$

2. Reachability of all semantic states: Any constraint causing a transition must be satisfiable in principle when using the dialogue operators and physical actions that are provided:

$$\forall (s, c, A, s') \in \Delta\ \exists r \in \mathcal{R}(Env, \mathcal{A}). CS(r) \cap c \neq \emptyset$$

3. Distinction between expected and compliant behaviour: The content of expectations must differ from that of normative commitments at least for some semantic variants (giving rise to non-compliant expectations for some runs):

$$\exists r \in \mathcal{R}(Env, \mathcal{A}) .expected(CS(r)) \neq compliant(CS(r))$$

4. Compliance/deviance realisability: It must be possible for agents in principle to comply with normative commitments or deviate from them in principle:

$$\exists r \in \mathcal{R}(Env, \mathcal{A}) .expected(CS(r)) \neq \emptyset \wedge$$
$$compliant(CS(r)) \neq \emptyset$$

While not absolutely essential, these constitute desiderata for the design of DS-ACLs as they add to the simplicity and clarity of a given semantics specification. Our framework raises interesting questions regarding further potential properties of DS such as:

1. *Respect for commitment autonomy*: The semantics must not allow an agent to create a pending commitment for another agent or to violate a commitment on behalf of another agent. While in some cases some agents should be able to enforce commitments upon others, this should generally be avoided to ensure agent autonomy.

2. *Avoiding commitment inconsistency:* The ACL must either disallow commitment to contradictory actions or beliefs, or at least provide operators for rectifying such contradictory claims. Under contradictory commitments, no possible behaviour can be compliant – it is up to the designer to decide to which extent this should be permitted.

3. *Unprejudiced judgement:* Expected behaviour prediction must not deviate from compliant behaviour prediction if deviant behaviour has not been observed so far (in particular this must hold for the initial semantic state). This might not always be desirable as "initial distrust" is necessary in some systems, but it increases the chances that agents will agree to participate in communication.

4. *Convergence:* The semantic state of each of the dialogue operators will remain stable after a finite number of transitions, regardless of any further agent behaviour[11]. If this property holds, this would imply that agents can stop tracking semantic state transitions after some amount of initial interaction. The advantage of this is reduced complexity, which of course comes at the price of giving up adaptiveness.

5. *Forgiveness:* After initial deviance, further compliant behaviour of an agent should lead to a semantic state that predicts compliant behaviour for that agent again. Here, we have to trade off cautiousness against the provision of incentives to resume cooperative behaviour. Trusting an agent makes others vulnerable to exploitation – "blacklisting" an agent forever, though, might lead that agent to keep up its unpredictable and potentially malicious behaviour.

6. *Equality:* Unless this is required by domain-specific constraints, the same dynamics of semantics should apply to all parties involved.

Our simple example semantics satisfies all these properties apart from convergence. Many of the above properties are debatable, as we have to trade off cautiousness against the provision of incentives for cooperative behaviour. While we cannot make any general statements here regarding "optimal" DS-ACL design, our framework provides the tools to test and evaluate the performance of different such communication-inherent sanctioning and rewarding mechanisms (i.e. social rules that do not presuppose ability to direct punishment or reward through physical actions) in real-world applications.

4. RELATED WORK

Expectation-based reasoning about interaction was first proposed in [2], considering the evolution of expectations described as probabilistic expectations of communication and action sequences. The same authors suggested a more general framework for expectation-based communication semantics [9], and argue for a "consequentialist" view of semantics that is based on defining the meaning of utterances in terms of their expected consequences and updating these expectations with new observations [11]. However, their approach does not use an explicit notion of commitments which in our framework mediates between communication and behaviour-based grounding, and provides a clear distinction between a normative notion of compliance and a more empirical notion of expectation.

Grounding for (mentalistic) ACL semantics has been investigated in [7] where grounded information is viewed as "information that is publicly expressed and accepted as being true by all the agents participating in a conversation". Like [1] (which bases the notion of "publicly expressed" on roles rather than internal states of agents) these authors' main concern is to provide a verifiable basis for determining the semantics of expressed mental states and commitments. Though our framework is only concerned with commitment to the achievement of states of affairs rather than exchanged information, in a sense, DS provides an alternative view by specifying what will happen if the assumptions on which "what is publicly accepted" is based are violated.

[11]In a non-trivial sense, i.e. when some initial transitions are possible in principle

Our framework is also related to deontic methods for the specification of obligations, norms and sanctions. In this area, [16] is the only framework that we are aware of which considers *dynamic* obligations, norms and sanctions. However, as we have described above we solely utilise *semantic evolution* as a sanctioning and rewarding mechanism, i.e. unlike this work we do not assume that agents can be directly punished or rewarded.

Finally, the FSM-like structure of the DS transition systems in combination with agent communication is reminiscent of work on *electronic institutions* [5], but there the focus is on providing different means of communication in different "scenes" of the interaction process (e.g. different protocols for different phases of market-based interaction) whereas we focus on different semantic variants that are to be used in the same interaction context.

5. CONCLUSION

This paper introduces dynamic semantics for ACLs as a method for dealing with some fundamental problems of agent communication in open systems, the simple underlying idea being that different courses of agent behaviour can give rise to different interpretations of meaning of the messages exchanged among agents. Based on a common framework of commitment-based semantics, we presented a notion of *grounding* for commitments based on notions of *compliant* and *expected* behaviour. We then defined dynamic semantics as state transition systems over different semantic states that can be viewed as different "versions" of ACL semantics in the traditional sense, and can be easily associated with a planning-based view of reasoning about communication. Thereby, our focus was on simplicity and on providing mechanisms for tracking semantic evolution in a "down-to-earth", algorithmic fashion to ensure applicability to many different agent designs.

We discussed the properties of our framework showing how it can be used as a powerful *communication-inherent* mechanism for rewarding and sanctioning agent behaviour in open systems without compromising agent autonomy, discussed its integration with agents' planning processes, complexity issues, and presented a list of desiderata for the design of ACLs with such semantics.

Currently, we are working on fully-fledged specifications of dynamic semantics for more complex languages and on extending our approach to mentalistic semantics where we view statements about mental states as commitments regarding the rational implications of these mental states (a simple example for this is that an agent commits itself to dropping an ostensible intention that it is claiming to maintain if that intention turns out to be unachievable). In this context, we are particularly interested in appropriate mechanisms to detect and respond to *lying* by "interrogating" suspicious agents and forcing them to commit themselves to (sets of) mental states publicly while sanctioning them when these are inconsistent with their actions.

6. REFERENCES

[1] G. Boella, R. Damiano, J. Hulstijn, and L. van der Torre. ACL Semantics between Social Commitments and Mental Attitudes. In *Proceedings of the International Workshop on Agent Communication* , 2006.

[2] W. Brauer, M. Nickles, M. Rovatsos, G. Weiß, and K. F. Lorentzen. Expectation-Oriented Analysis and Design. In *Proceedings of the 2nd Workshop on Agent-Oriented Software Engineering* , LNCS 2222, 2001. Springer-Verlag, Berlin.

[3] P. R. Cohen and H. J. Levesque. Communicative actions for artificial agents. In *Proceedings of the First International Conference on Multi-Agent Systems*, pages 65–72, 1995.

[4] P. R. Cohen and C. R. Perrault. Elements of a Plan-Based Theory of Speech Acts. *Cognitive Science*, 3:177–212, 1979.

[5] M. Esteva, J. Rodriguez, J. Arcos, C. Sierra, and P. Garcia. Formalising Agent Mediated Electronic Institutions. In *Catalan Congres on AI*, pages 29–38, 2000.

[6] N. Fornara and M. Colombetti. Operational specification of a commitment-based agent communication language. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 536–542, Bologna, Italy, 2002. ACM Press.

[7] B. Gaudou, A. Herzig, D. Longin, and M. Nickles. A New Semantics for the FIPA Agent Communication Language based on Social Attitudes. In *Proceedings of the 17th European Conference on Artificial Intelligence*, Riva del Garda, Italy, 2006. IOS Press.

[8] F. Guerin and J. Pitt. Denotational Semantics for Agent Communication Languages. In *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 497–504. ACM Press, 2001.

[9] M. Nickles, M. Rovatsos, and G. Weiss. Empirical-Rational Semantics of Agent Communication. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, New York, NY, 2004.

[10] J. Pitt and A. Mamdani. Some Remarks on the Semantics of FIPA's Agent Communication Language. *Autonomous Agents and Multi-Agent Systems*, 2:333–356, 1999.

[11] M. Rovatsos, M. Nickles, and G. Weiß. Interaction is Meaning: A New Model for Communication in Open Systems. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, Melbourne, Australia, 2003.

[12] M. D. Sadek. Dialogue acts are rational plans. In *Proceedings of the ESCA/ETRW Workshop on the Structure of Multimodal Dialogue*, pages 1–29, 1991.

[13] M. Singh. Agent communication languages: Rethinking the principles. *IEEE Computer*, 31(12):55–61, 1998.

[14] M. Singh. A social semantics for agent communication languages. In *Proceedings of the IJCAI Workshop on Agent Communication Languages*, 2000.

[15] M. P. Singh. A semantics for speech acts. *Annals of Mathematics and Artificial Intelligence*, 8(1–2):47–71, 1993.

[16] G. Weiß, M. Nickles, M. Rovatsos, and F. Fischer. Specifying the Intertwining of Cooperation and Autonomy in Agent-based Systems. *Journal of Networks and Computer Applications*, 29, 2007.

[17] M. J. Wooldridge. Verifiable semantics for agent communication languages. In *Proceedings of the Third International Conference on Multi-Agent Systems*, pages 349–356, Paris, France, 1998.