Competent Agents and Customising Protocols

Ulle Endriss[1], Wenjin Lu[2], Nicolas Maudet[3], and Kostas Stathis[2]

[1] Department of Computing, Imperial College London
180 Queen's Gate, London SW7 2AZ (UK)
Email: ue@doc.ic.ac.uk

[2] Department of Computing, School of Informatics, City University
Northampton Square, London EC1V OHB (UK)
Email: {lu,stathis}@soi.city.ac.uk

[3] LAMSADE, Université Paris-Dauphine
75775 Paris Cedex 16 (France)
Email: maudet@lamsade.dauphine.fr

Abstract. In open agent societies, communication protocols and strategies cannot be assumed to always match perfectly, because they are typically specified by different designers. These potential discrepancies raise a number of interesting issues, most notably the problem of checking that the behaviour of an agent is (or will be) conformant to the rules described by a protocol. In this paper, we argue that the ability to merely conform to a protocol is not sufficient for an agent to be a *competent* user of that protocol. We approach the intuitive idea of protocol competence by introducing a notion that considers, broadly speaking, an agent's ability to reach a particular state of the interaction and we provide preliminary results that allow us to automatically check competence in the context of a specific class of logic-based agents. Finally, we illustrate how these results can facilitate the customisation of protocols used by agents that are not fully competent.

1 Introduction

Communication is one of the key feature of societies of artificial agents [11], and standards are required to regulate the distributed decision-making involved in interactions such as, for instance, negotiation or persuasion. A *protocol* specifies the "rules of encounter" governing a dialogue between two or more communicating agents [7]. It specifies which agent is allowed to say what in any given situation. It will usually allow for several alternative utterances in every situation and the agent in question has to choose one according to its *strategy.* The protocol is *public*, while each agent's strategy is *private.* In open societies, protocols and strategies cannot be assumed to match perfectly. Protocols are typically specified by the designer of the application, whereas strategies are implemented by the designer of the agent. The potential discrepancies caused by these different points of view raise a number of interesting issues, most notably the problem of checking that the behaviour of an agent is (or will be) conformant to the rules described by a protocol [2]. In this paper, we argue that the ability to merely conform to a protocol, however, is not sufficient for an agent to be a *competent* user of that protocol.

Intuitively, we understand competence with respect to a protocol as the capacity of the agent "to deal adequately" with a protocol. Surely this may involve different notions. To start with, it is clear that the agent must have the ability to understand the meaning of the messages —to put it another way, the agent must share the *ontology* used in the interaction. This means that the agent must interpret the meaning of communicative acts (the performatives and their content) properly. Another requirement is that the agent should be able to give meaningful answers within the time window specified by the protocol. Note that the problem is not whether the agent can actually give a response (a point that we shall discuss later on), but whether the answer (if given) is fully satisfactory to the agent. On top of that, the agent should be available until the protocol terminates and not leave the interaction before. In this paper, we will take for granted that agents (i) share the same ontology, (ii) have sufficient reasoning capabilities to deal with the different types of message exchanged, (iii) are available until the end of interaction. We are now in a position to give a first hint of the notion we will investigate in this paper: competence amounts to evaluating whether the agent can explore (*i.e.* reach the different states) of a given protocol.

The remainder of this paper is structured as follows: the next section describes the logic-based representation for protocols that we shall use throughout this paper. Section 3 discusses what we regard as a first requirement for an agent to be considered competent, namely that this agent should be able to give at least one legal response to any message expected as part of the protocol (exhaustive conformance). However, as we shall see, this is not a fully satisfactory notion of competence. In Section 4 we go one step further and introduce the competence of an agent as, generally speaking, its ability to reach a particular state of the interaction. We provide preliminary results that allow us to automatically check competence in the context of our logic-based agents. Section 5 introduces a notion of practical importance: the customisation of protocols in order to adapt them to agents that are not fully competent. We illustrate this idea by means of a protocol inspired by electronic transactions. Section 6 concludes.

2 Protocols for logic-based agents

In this paper, we shall only consider protocols that can be represented by means of a deterministic finite automaton (DFA). This is a widely used class of agent interaction protocols (see [6] and others), but we should also point out that certain types of interaction require more expressive formalisms (for instance, where concurrent communication is required). We recall here that a DFA consists of (i) a set of states (including an initial state, and a set of final states), (ii) a set of events, and (iii) a transition function δ which maps pairs of states and events to states. Given a DFA with transition function δ, a dialogue move P is a legal continuation wrt. a state S iff there exists a state S' such that $S' = \delta(S, P)$. We shall refer to *legal inputs* (respectively *outputs*) for an agent X as those legal

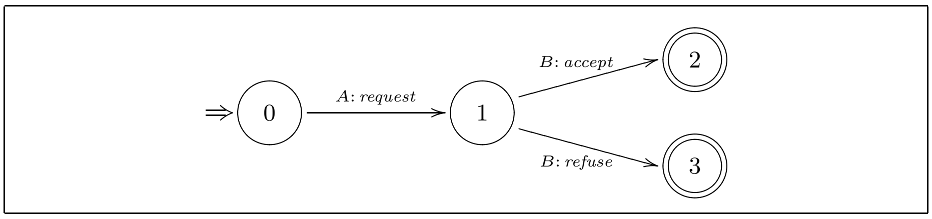

Fig. 1. A simple negotiation protocol

continuations where X is the receiver (respectively the utterer) of the dialogue move. Figure 1 is an (admittedly very simple) interaction protocol for negotiation which specifies that, following a *request* made by agent A, agent B should in turn either *accept* or *refuse* this request.

This protocol, as shown below, can be translated into two sets of if-then rules corresponding to the two subprotocols used by agents A and B, respectively.

$$\mathcal{P}_A : \begin{cases} \mathit{START}(T) \Rightarrow \mathit{request}(T{+}1) \\ \mathit{accept}(T) \quad \Rightarrow \mathit{STOP}(T{+}1) \\ \mathit{refuse}(T) \quad \Rightarrow \mathit{STOP}(T{+}1) \end{cases} \mathcal{P}_B : \left\{ \mathit{request}(T) \Rightarrow \mathit{accept}(T{+}1) \;\vee \mathit{refuse}(T{+}1) \right.$$

For each of the above implications, variables are understood to be implicitly universally quantified. In case we also have variables that only appear on the righthand side of an implication, these variables would be existentially quantified. This logic-based representation of protocols has been introduced in [2].

Shallow protocols. We call protocols that permit such a straightforward translation into if-then rules, with only a single performative on the lefthand side, *shallow*. Shallow protocols correspond to automata where it is possible to determine the next state of the dialogue on the sole basis of the previous event. Of course, this is not always the case since it may be necessary to refer to the current state of the dialogue to determine the new state. In principle however, any automata-based protocol can be transformed into a protocol that is shallow in this sense (by simply renaming any duplicate transitions).

To ensure that protocols defined in such a way are well-formed, we stipulate a list of constraints that rules should meet, in particular that any dialogue move occurring on the righthand side of the first subprotocol (except *STOP*) also occurs on the lefthand side of the second one, and vice versa (*matching*). We refer to [2] for further details. The meaning of each rule which appears in a protocol is then intuitively clear: it specifies for any expected dialogue move the set of correct responses the agent may utter in reply. The set of performatives appearing on the lefthand side of the constraints defining an agent's subprotocol are called the *expected inputs* for that agent.

Agents' strategies. Following [8], the communication *strategy* $\mathcal{S}$ of an agent (which forms part of its so-called *knowledge base* $\mathcal{K}$) is represented as a set of integrity constraints of the following form:

$$P(T) \wedge C \Rightarrow P'(T{+}1)$$

On receiving dialogue move P at time T, an agent implementing this rule would utter P' at time $T{+}1$, provided condition C is entailed by its (*private*) knowledge base. Variables are understood to be implicitly quantified in the same way as for our protocol-rules.

3 Competence as exhaustive conformance

In this section, we study the notion of (exhaustive) conformance, which could be regarded as a first requirement for an agent to be considered competent. Intuitively, an agent is conformant to a given communication protocol $\mathcal{P}$ whenever its utterances are legal according to that protocol. Following [2], we distinguish two *levels* of conformance:

– An agent is *weakly conformant* to a protocol $\mathcal{P}$ iff it never utters an illegal dialogue move (with respect to $\mathcal{P}$).
– An agent is *exhaustively conformant* to a protocol $\mathcal{P}$ iff it does utter a legal dialogue move whenever required to do so by $\mathcal{P}$.

That is, an agent is exhaustively conformant to the protocol $\mathcal{P}$ iff (i) it is weakly conformant to $\mathcal{P}$ and (ii) it will utter at least *some* dialogue move whenever it is its turn (the legality of which will be ensured by the first condition). This ability to give a response to any expected input may be considered as a first requirement for an agent's competence to use a given protocol. Note that [2] also introduces a third level (robust conformance), which requires agents not only to be exhaustively conformant, but also to react appropriately to illegal moves uttered by *other* agents. In this paper, however, we are only concerned with weak and exhaustive conformance.

In general, checking *a priori* (that is, at design time) whether an agent will always behave in conformance to a given set of protocols is difficult, if not impossible. Still, as shown in [2], it *is* possible to guarantee at least weak conformance to a shallow protocol for the type of agent described in the previous section. To this end, we define the *response space* $\mathcal{S}^*$ of an agent with the communication strategy $\mathcal{S}$ based on the language $\mathcal{L}$ as follows:

$$\{P(T) \Rightarrow \bigvee\{P'(T{+}1) \mid [P(T) \wedge C \Rightarrow P'(T{+}1)] \in \mathcal{S}\} \mid P \in \mathcal{L}\} \text{ with } \bigvee\{\} = \bot$$

This is the set of protocol constraints we obtain by first dropping all conditions referring to the private knowledge of the agent in question and then conjoining implications with identical antecedents by collecting the corresponding consequents into a single disjunction. This abstraction from an agent's communicative behaviour is related to the idea of an agent automaton proposed by Singh [9]. It allows us to state a simple sufficiency criterion for weak conformance:

An agent is weakly conformant to a protocol $\mathcal{P}$ whenever that protocol is a logical consequence of the agent's response space.

This result shows that, in the case of weak conformance, it is possible to check conformance a priori by inspecting only a relatively small part of an agent's specification (namely, what we could call its "communication module"). In particular, we are *not* required to make any judgements based on the content of its (probably dynamically changing) knowledge base in general.

In the case of exhaustive conformance, the situation is rather different. To understand why, let us first take a closer look at what exhaustive conformance involves, beyond the requirements shared with the notion of weak conformance. As pointed out earlier, we can separate the property of exhaustive conformance into two parts: (i) weak conformance and (ii) the property of uttering any move at all. The latter property, which we shall simply refer to as *exhaustiveness* (of an agent) may be considered independently from a particular protocol. Sadri *et al.* [8], for instance, define the notion of exhaustiveness with respect to a given communication language (as being able to utter a response for any incoming move belonging to that language). Even for our agents, whose communicative behaviour is determined by constraints of the form $P(T) \wedge C \Rightarrow P'(T{+}1)$, it is not generally possible to guarantee exhaustiveness (be it with respect to a given protocol, language, or in general). We cannot generally ensure that one of these rules will indeed "fire" for an incoming move $P(T)$, because none of the additional conditions C may be entailed by the current state of the agent's knowledge base.

One way of ensuring exhaustive conformance would be to rely on logical truths that are independent from the (possibly dynamic) knowledge base of the agent. For a strategy $\mathcal{S}$, let $\mathrm{COND}_{\mathcal{S}}(P)$ denote the disjunction of all the private conditions that appear in $\mathcal{S}$ in a constraint together with the trigger $P(T)$:

$$\mathrm{COND}_{\mathcal{S}}(P) = \bigvee\{C \mid [P(T) \wedge C \Rightarrow P'(T{+}1)] \in \mathcal{S}\} \text{ with } \bigvee\{\,\} = \bot$$

Now, if $\mathrm{COND}_{\mathcal{S}}(P)$ is a logical theorem for every performative P appearing on the lefthand side of the relevant subprotocol of a protocol $\mathcal{P}$, then any agent implementing the strategy $\mathcal{S}$ is guaranteed to utter *some* move for any input expected in $\mathcal{P}$. Hence, we obtain a useful sufficient criterion for exhaustive conformance (again, with respect to our shallow protocols):

An agent with strategy $\mathcal{S}$ is exhaustively conformant to a protocol $\mathcal{P}$ whenever it is weakly conformant to $\mathcal{P}$ and $\mathrm{COND}_{\mathcal{S}}(P)$ is a theorem for every expected input P (for that agent, with respect to $\mathcal{P}$).

Of course, generally speaking, checking this condition is an undecidable problem because verifying theoremhood in first-order logic is. In practice, however, we would not expect this to be an issue given the simplicity of typical cases. As an example, consider a protocol consisting of only the following rule stipulating that any request by another agent X should be either accepted or refused:

$$\mathit{request}(X, T) \Rightarrow \mathit{accept}(T{+}1) \vee \mathit{refuse}(T{+}1)$$

An agent may implement the following simple strategy:

$$\begin{array}{l} \mathit{request}(X,T) \wedge \mathit{friend}(X) \quad \Rightarrow \mathit{accept}(T{+}1) \\ \mathit{request}(X,T) \wedge \neg \mathit{friend}(X) \Rightarrow \mathit{refuse}(T{+}1) \end{array}$$

The disjunction $\neg \mathit{friend}(X) \vee \mathit{friend}(X)$, with X being implicitly universally quantified, is a theorem. Hence, our agent would be exhaustively conformant (note that the agent is certainly going to be weakly conformant, because the protocol is a consequence of its response space —in fact, the two are even identical here). A similar idea is also present in [8], although not in the context of issues pertaining to protocol conformance.

Fulfilling the above criterion is not an unreasonable requirement for a well-designed communication strategy $\mathcal{S}$ that is intended to be used for interactions governed by a given protocol $\mathcal{P}$. This is why, in the remainder of this paper, we are sometimes going to assume that our agents are known to be exhaustively conformant (despite the fact that, in a more general setting, such an assumption would not be justified).

We continue our discussion of exhaustiveness by observing that, in cases where we can identify a *static* part of an agent's knowledge base (beyond the set of constraints making up its communication strategy), we can give an even more general sufficiency criterion that guarantees exhaustive conformance:

> *An agent with strategy $\mathcal{S}$ is exhaustively conformant to a protocol $\mathcal{P}$ whenever it is weakly conformant to $\mathcal{P}$ and* $\mathrm{COND}_{\mathcal{S}}(P)$ *is a logical consequence of the agent's knowledge base for every expected input P.*

To illustrate the idea, we slightly change our example from before an replace the agent's second communication rule with the following constraint:

$$\mathit{request}(X,T) \wedge \mathit{enemy}(X) \Rightarrow \mathit{refuse}(T{+}1)$$

That is, our agent will refuse any request by X if it considers X to be an enemy. Now our first criterion does not apply anymore; we cannot ensure exhaustive conformance. However, if the agent's knowledge base includes a formula such as $\neg \mathit{enemy}(X) \Rightarrow \mathit{friend}(X)$, expressing that anyone who is not an enemy should be considered a friend, then we can show that $\mathit{friend}(X) \vee \mathit{enemy}(X)$ is a logical consequence of that knowledge base and, thereby, that our agent will be exhaustively conformant to the protocol. Note that this agent may generate two responses for a single input, namely in cases where both $\mathit{friend}(X)$ and $\mathit{enemy}(X)$ are true. To avoid such situations (*i.e.* to ensure deterministic behaviour), we could add the formula $\neg(\mathit{friend}(X) \wedge \mathit{enemy}(X))$ to the knowledge base, resulting in the popular "you are either with us or against us" policy.

We can also use our criteria to pinpoint critical situations where a given (non-exhaustive) agent would not be able to respond appropriately. To illustrate this, suppose we remove the implication $\neg \mathit{enemy}(X) \Rightarrow \mathit{friend}(X)$ from our agent's knowledge base. When trying to establish exhaustive conformance, we would have to prove that $\mathit{friend}(X) \vee \mathit{enemy}(X)$ is —still— a consequence of the knowledge base, *i.e.* $\neg(\mathit{friend}(X) \wedge \mathit{enemy}(X)) \models \mathit{friend}(X) \vee \mathit{enemy}(X)$. Such a proof would be bound to fail and we could use it to construct an explicit counterexample. For instance, if we were to use *analytic tableaux*, we would attempt a refutation for a tableau initially labelled with the two formulas $(\forall X)\neg(\mathit{friend}(X) \wedge \mathit{enemy}(X))$ and $\neg(\forall X)(\mathit{friend}(X) \vee \mathit{enemy}(X))$, *i.e.* the (only) premise and the negated conclusion. We would soon end up with a saturated open branch containing the literals $\neg\mathit{friend}(a)$ and $\neg\mathit{enemy}(a)$, where a is the Skolem constant introduced when analysing the negated conclusion.[4] For details we refer to the literature on analytic tableaux [1, 3].

This branch gives rise to the first-order model $\mathcal{M} = (\mathcal{D}, \mathcal{I})$ with $\mathcal{D} = \{a\}$ and both $a \notin \mathcal{I}(\mathit{friend})$ and $a \notin \mathcal{I}(\mathit{enemy})$. This shows that our agent will fail to be exhaustively conformant as soon as it receives a request from an agent a that is neither (known to be) a friend nor an enemy. Of course, if we have additional information about the application domain, which allows us, for instance, to infer that such a situation would be impossible, then we may still be able to show exhaustive conformance. However, such methods would go beyond the purely logic-based techniques we are interested in for the purpose of the present paper.

Exhaustive conformance can only be considered a first requirement for competent agents. To illustrate this point, let us consider again the protocol of Figure 1, and assume that agent B takes part in the interaction using the following response space:

$$\mathcal{S}^* = \{\mathit{request}(T) \Rightarrow \mathit{refuse}(T{+}1)\}$$

Even if this agent was indeed exhaustively conformant (as discussed before), it would intuitively not be competent as it could never reach state 2 (and consequently the interaction could never terminate with an accepted request). In the following section, we try to overcome this limitation by introducing a new definition of competence.

4 Competence as reachability

In this section, we define competence as the ability of a (group of) agent(s) to reach the different stages of an interaction. In particular, from a practical point of view, we shall be especially interested in the termination of interactions.

To begin with, it is worth noting that by the condition of matching on the well-formedness of protocols, any legal (and "complete") dialogue will either end with a *STOP* move or be infinite. If a dialogue ends illegally (*i.e.* with a final move different from *STOP*) it must be the case that one of the agents is not exhaustively conformant to the protocol in operation. If both agents are known to be at least weakly conformant to the protocol (but not necessarily exhaustively conformant), then we can distinguish three situation: (i) the dialogue ends legally with a *STOP* move, (ii) the dialogue terminates illegally with a move different

[4] Again, due to the undecidability of first-order logic, this technique may not always be applicable, but we believe that in many practical cases it will.

from *STOP*, and (iii) the dialogue goes on forever.[5] Ideally, we would like to avoid both (ii) and (iii). In practice, we want our agents to have at least the *potential* to achieve (i). While two exhaustively conformant agents will never generate a dialogue that terminates *illegally*, they may or may not have the *competence* to generate a dialogue that does terminate at all.

For a pair of agents, having the competence to produce a legally ending dialogue amounts to these agents being able to generate a dialogue following on from a *START* move that includes (and thereby ends with) a *STOP* move. More generally, for any two dialogue moves P and P', we may ask whether a given pair of agents possesses the competence to generate a dialogue including P' once P has been uttered. Here, P' may be the *STOP* move or any other move leading to an "interesting" state in the dialogue.

> *For some protocol $\mathcal{P}$, two agents have the joint competence to reach P' from P iff they have the ability to generate a sequence of dialogue moves that are legal with respect to $\mathcal{P}$ and that include P' once P has been uttered.*

In particular, two agents have the joint competence to generate a legally terminating dialogue iff the have the joint competence to reach *STOP* from *START*.

We will now investigate how this notion of competence can be automatically checked in the context of our logic-based agents. To this end, we will make use of the notion of response space already introduced in Section 3; or more precisely of a propositional representation of these response spaces obtained by removing the reference to time as well as to any other variables (this, of course, assumes that the content of the dialogue moves is not directly relevant to our purpose).

We call the set of formulas we obtain by removing all references to variables from an agent's response space the *flattened* response space of that agent. A flattened dialogue constraint such as $\mathit{request} \Rightarrow \mathit{accept} \vee \mathit{refuse}$ may be interpreted as requiring any legal dialogue that includes a *request* move to also include either an *accept* or a *refuse* move (of course, in addition to that, the *intended* meaning is that one of these moves *immediately* follows the *request* move, although this is not made explicit in this simplified representation). We note here that this kind of flattening operation would not be meaningful for protocols (and hence response spaces) that are not shallow, *i.e.* for protocols where we may have more than one trigger on the lefthand side of a protocol constraint (referring to different times in the dialogue history).

We can use the notion of a flattened response space to formulate a sufficient criterion for joint competence:

> *Two agents with flattened response spaces $\mathcal{S}^*_A$ and $\mathcal{S}^*_B$ that are exhaustively conformant to protocol $\mathcal{P}$ have the competence to reach P' from P iff $\mathcal{S}^*_A \cup \mathcal{S}^*_B \cup \{P\} \models P'$*

[5] Note that we do not classify infinite dialogues as illegal. In fact, a (badly designed) protocol may even preclude agents from ending a dialogue after having have made a certain unfavourable choice (*i.e.* after having uttered some dialogue move leading to a conversational state from where there are no paths leading to a final state).

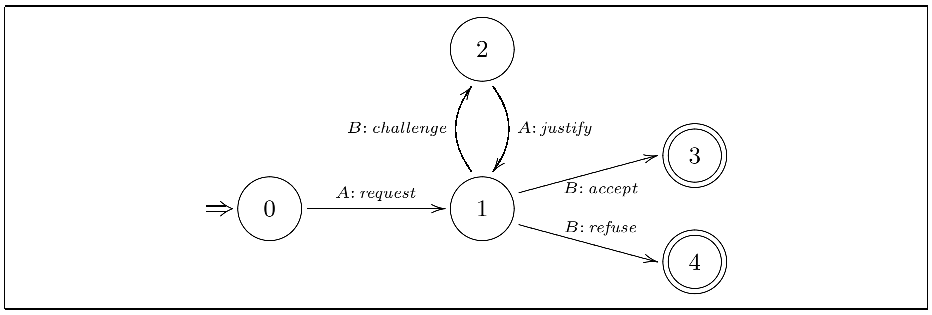

Fig. 2. A negotiation protocol with a justify/challenge loop

To see that the above is only a suitable competence criterion for agents that are exhaustively conformant, it is important to recall that $P \Rightarrow \bot$ will form part of an agent's (flattened) response space for every expected input P for which there is no rule at all in the agent's communication strategy. This means, for instance, that *STOP* would be derivable from the union of the two response spaces together with $\{\mathit{START}\}$ even though, clearly, such a pair of agents would not be guaranteed to reach a final state.

To illustrate this notion of competence, we introduce an improvement of our negotiation protocol, depicted in Figure 2, which includes a justify/challenge loop inspired by argumentation-based protocols [5]. For instance, given the following flattened response spaces

$$\mathcal{S}_A^* : \begin{cases} \mathit{START} & \Rightarrow \mathit{request} \\ \mathit{challenge} & \Rightarrow \mathit{justify} \\ \mathit{accept} & \Rightarrow \mathit{STOP} \\ \mathit{refuse} & \Rightarrow \mathit{STOP} \end{cases} \quad \mathcal{S}_B^* : \begin{cases} \mathit{request} \Rightarrow \mathit{accept} \vee \mathit{refuse} \\ \mathit{justify} \;\Rightarrow \mathit{accept} \vee \mathit{refuse} \end{cases}$$

it is easy to see that, for $P = \mathit{START}$ and $P' = \mathit{STOP}$, we indeed have

$$\mathcal{S}_A^* \cup \mathcal{S}_B^* \cup \{P\} \models P'$$

Intuitively, this means that termination will be reached by the agents equipped with these response spaces. In general, of course, this will not be the case. Consider for instance the following response spaces.

$$\mathcal{S}_A^* : \begin{cases} \mathit{START} & \Rightarrow \mathit{request} \\ \mathit{challenge} & \Rightarrow \mathit{justify} \\ \mathit{accept} & \Rightarrow \mathit{STOP} \\ \mathit{refuse} & \Rightarrow \mathit{STOP} \end{cases} \quad \mathcal{S}_B^* : \begin{cases} \mathit{request} \Rightarrow \mathit{accept} \vee \mathit{refuse} \vee \mathit{challenge} \\ \mathit{justify} \;\Rightarrow \mathit{accept} \vee \mathit{refuse} \vee \mathit{challenge} \end{cases}$$

In this case, clearly, the agents can generate a legal infinite dialogue by alternating between uttering challenges and justifications.

To analyse this situation, let us consider the set of all *minimal models* of $\mathcal{S}_A^* \cup \mathcal{S}_B^* \cup \{\mathit{START}\}$. We identify a model with the set of propositional letters that are being interpreted as true in that model and call a model minimal iff there is no other model for which this characteristic set is a strict subset of the characteristic set of the former. In the case of $\mathcal{S}_A^* \cup \mathcal{S}_B^* \cup \{START\}$, we obtain the following three minimal models:

$$\begin{aligned}\{\ &\{START, \mathit{request}, \mathit{accept}, STOP\},\\ &\{START, \mathit{request}, \mathit{refuse}, STOP\},\\ &\{START, \mathit{request}, \mathit{challenge}, \mathit{justify}\}\ \}\end{aligned}$$

We observe that we can distinguish two kinds of models: (i) minimal models including *STOP*, (ii) minimal models not including *STOP*. These correspond to the different sort of dialogues that can be generated from these response spaces: the two first models (including *STOP*) correspond to terminating dialogues, whereas the last model represents the infinite loop. In this case, we can only state that termination is *possible*, since there exists at least one minimal model including both *START* and *STOP*. However, as long as there is a minimal model not including *STOP*, termination cannot be guaranteed.

But it is also noteworthy in this example that, even when the agents have entered the loop, they still have the possibility to reach termination (B can choose to utter *refuse* or *accept* after a *justify*). This is not necessarily the case in general, and we can then distinguish *good* and *bad* loops. A good loop is a loop that need not be infinite, *i.e.* at least one path to a final state is still open. A bad loop is a loop where agents have no other choice than to repeat the same sequence of utterances (that would be the case in our example if B could neither utter *accept* nor *refuse* in reply to *justify*). We now give a criterion that allows us to assess whether a given loop is good:

> *Consider the model of a loop. If for every P in that model, $\mathcal{S}_A^* \cup \mathcal{S}_B^* \cup \{P\}$ still has got at least one minimal model including STOP, then this is a good loop, otherwise this is a bad loop.*

One may be inclined to believe that bad loops only result from badly designed protocols. However, because the designer of a protocol does not know in advance the competence of the agents that will take part in the interaction, it may be the case that a protocol turns out to be problematic because it is used by agents that are not fully competent. In this case, it can be useful for the designer of the application to *customise* his protocol, in order to avoid these undesirable situations.

5 Customising protocols

There are two ways of adapting protocols to the needs of individual agents that are not fully competent to use them in the first place. The first approach is to require an enhancement of the strategies of the agents, the second to customise the protocol itself (these two techniques are called *expansion* and *filtering* in the context of game-based interaction studied in [10]). We will now show by means

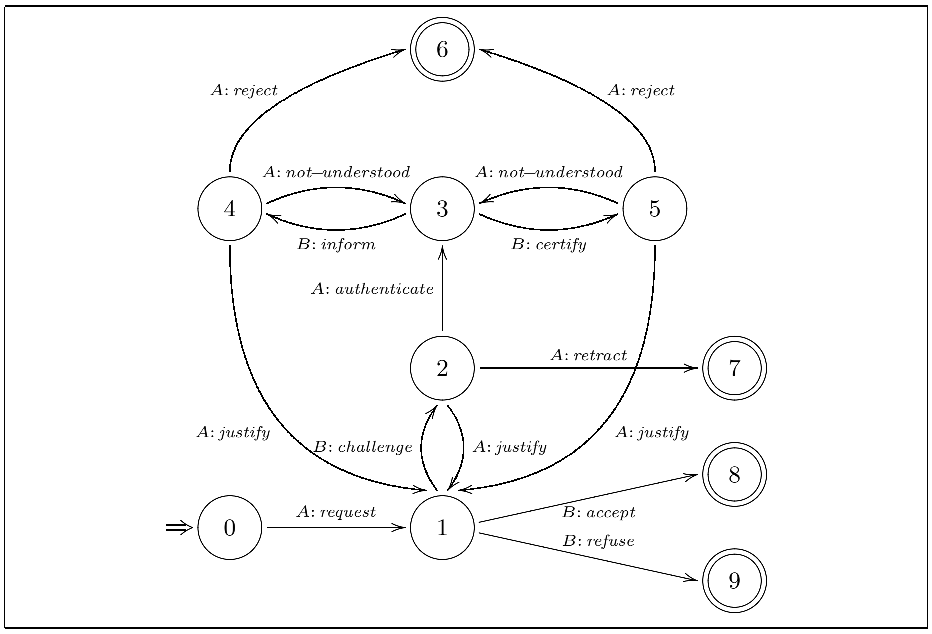

Fig. 3. A protocol for electronic transactions

of an example how the preliminary results put forward in the previous section can be used to customise protocols.

For this purpose, we will introduce a further improvement of our negotiation protocol, inspired by electronic transactions (see Figure 3). The interaction starts with a consumer (A) *request*ing a good. The seller (B) can then *accept* or *refuse* (leading to terminal states 8 or 9), or alternatively *challenge* the consumer (for instance, the seller may challenge that the consumer has the permission to purchase this good). The consumer is in a situation where he can *retract* his request, provide some *justif* ication, or himself interrogate the seller as to whether he is indeed in a position to challenge him in the first place. This is done by requiring an *auhenticat*ion of the seller. This latter move leads to a sort of subprotocol starting from state 3. The protocol then assumes that the buyer could use two methods of authentication: (i) a simple method involving a text ticket simply provided by an *inform* message; or (ii) a complex method using a encryption key included in a *certif* icate. In both cases, the consumer should explicitly indicate if he cannot evaluate the response provided by the seller (*not-understood*, in which case the interaction goes back to the previous state in order to possibly use a different method of authentication). In case of a negative evaluation of the authentication provided, the buyer will *reject* and terminate the interaction (state 6). In case of a positive evaluation, the buyer should then give justification to the challenge uttered earlier. Note that this protocol is shallow.

Let us now assume that the consumer has no way to evaluate an encrypted certificate provided by the seller (*i.e.* the rule for the trigger *certify* in his response space does not include *reject*), as well as no argumentative capabilities to *justify* his claims. As far as the seller is concerned, we will assume on the other hand that only the encrypted authentication can be provided (*i.e.* the seller's response space does not include *inform*). In other words, after a *challenge*, A can only either ask B to *authenticate*, or *retract*. It is easy to observe that, if A decides to ask for an authentication, the interaction will end up in a bad loop between states 3 and 5 (that is, the consumer explaining he does not understand the provided certificate, but the seller lacking alternative methods to authenticate himself).

Our proposal in such a situation is to *customise* the protocol to prevent the participants from entering this loop by cutting this part of the protocol. This can be done systematically by (i) identifying the bad loops resulting from the agents' response spaces, as discussed in the previous section, and then (ii) removing all occurrences in the protocol of those moves of the bad loops that do not allow to reach termination. In our example, the minimal model corresponding to the bad loop would be

$$\{START, \mathit{request}, \mathit{challenge}, \mathit{authenticate}, \mathit{certify}, \mathit{not\text{-}understood}\}$$

which would lead to the subsequent deletion (for the period of this interaction) of *authenticate*, *certify*, *not-understood* from the protocol —indeed, none of these moves P is such that $\mathcal{S}_A^* \cup \mathcal{S}_B^* \cup \{P\}$ has got at least one minimal model including $STOP$. The resulting protocol could then be used safely by the agents (even if any challenge by the seller would immediately lead to the retraction of the request by the buyer).

In practice, we envisage this technique of customising protocols for interactions in open agent societies to be used as follows. Suppose two agents are about to enter an interaction governed by a particular protocol $\mathcal{P}$. Before starting the dialogue, they may send their respective response spaces to the authority regulating this interaction. Using the agents' response spaces, the authority can determine whether the agents would be competent users of $\mathcal{P}$. In case they are not, the authority can customise $\mathcal{P}$, using the methodology described earlier, and propose to the agents to use that customised protocol instead of the original one.

6 Conclusion

In this paper, we have argued that the ability to merely *conform* to a protocol (in the sense of not uttering any illegal dialogue moves) is not sufficient for an agent to be a *competent* user of that protocol. As a first approach to protocol competence we have further examined the concept of *exhaustive conformance* introduced in [2] and defined criteria that allow us to check whether a given agent can be expected to behave exhaustively conformant to a given protocol. While exhaustive conformance does guarantee that an agent will be able to utter at least some legal performative at any point in a dialogue, it does, for instance, still not ensure that such a dialogue will eventually terminate. Therefore, as a further step towards characterising agents that are truly competent users of a protocol, we have introduced a notion that considers, broadly speaking, an agent's ability to reach a particular state of the interaction, and we have provided preliminary results that allow us to automatically check competence in the context of the specific class of logic-based agents considered here. We have then shown how these results can facilitate the *customisation* of protocols used by agents that are not fully competent.

Whereas the notion of *competence as reachability* discussed in this paper makes reference to the group of agents involved in the interaction, it may be useful for a single agent to (even roughly) evaluate whether it is competent to use a given protocol, regardless of whom it may happen to interact with. For this purpose, we may evaluate what would happen if our agent was to interact with a *perfect* agent (that is, an agent that could provide any expected inputs, as defined by the protocol).

The notion of response space can also be used to provide a rough approximation of the individual competence of the agent: intuitively, given a protocol $\mathcal{P}$, we would expect a competent agent to have a response space that (almost) "covers" $\mathcal{P}$, namely it should have the potential to utter as many dialogue moves as the protocol allows.

In our future work, we plan to investigate whether the techniques used in this paper could help to develop a methodology for assisting in the design of *fair* protocols, in the sense that such a protocol should "treat all participants equally, or, if not, make explicit any asymmetries in their treatment" [4]. For instance, agents may assign different values to different final states of a protocol. In case a pair of agents would not have the competence to reach any of the final states rated positively by one of the agents after a particular dialogue move has been uttered, this would amount to an unfair advantage for the other agent.

Acknowledgements. We would like to thank the ESAW referees for their helpful comments. This research has been funded by the European Union as part of the SOCS project (IST-2001-32530) and has been carried out while the third author was employed at City University, London.

References

1. M. D'Agostino, D. Gabbay, R. Hähnle, and J. Posegga, editors. *Handbook of Tableau Methods.* Kluwer Academic Publishers, 1999.
2. U. Endriss, N. Maudet, F. Sadri, and F. Toni. Protocol Conformance for Logic-based Agents. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-2003).* Morgan Kaufmann, 2003.
3. M. Fitting. *First-order Logic and Automated Theorem Proving.* Springer-Verlag, 2nd edition, 1996.
4. P. McBurney and S. Parsons. Desiderata for Inter-agent Protocols. In *Proceedings of the First International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-2002)*, Bologna, Italy, 2002.

5. S. Parsons, C. Sierra, and N. Jennings. Agents that Reason and Negotiate by Arguing. *Journal of Logic and Computation*, 8(3):261–292, 1998.
6. 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-1999).* Morgan Kaufmann, 1999.
7. J. S. Rosenschein and G. Zlotkin. *Rules of Encounter.* MIT Press, 1994.
8. F. Sadri, F. Toni, and P. Torroni. Dialogues for Negotiation: Agent Varieties and Dialogue Sequences. In *Proceedings of the 8th International Workshop on Agent Theories, Architectures and Languages (ATAL-2001).* Springer-Verlag, 2001.
9. M. P. Singh. A Customizable Coordination Service for Autonomous Agents. In *Proceedings of the 4th International Workshop on Agent Theories, Architectures and Languages (ATAL-1997)*, 1997.
10. K. Stathis. A Game-based Architecture for Developing Interactive Components in Computational Logic. *Journal of Functional and Logic Programming*, 2000(5), 2000.
11. M. Wooldridge. *An Introduction to Multiagent Systems.* MIT Press, 2002.