Proving Trust Locally

Anders Moen Hagalisletto
Norwegian Computing Center and
Department of Informatics, University of Oslo, Norway
andersmo@ifi.uio.no

Abstract

Simulators and analyzers for protocols do not distinguish between the local 'subjective' view of the agents and the global 'objective' view of the analysis perspective. In practice this means that security analysis in general and protocol analysis in particular do neither model the agents local beliefs nor their local deduction power in a satisfactory way. This paper suggests a solution to the problem by proposing a new approach to epistemic logic, explained in terms of term rewriting, in order to make agents capable of performing logical deductions themselves. Local deductions of security properties, like trust, are crucial to assure that the agents can expand their knowledge about the environment they inhabit.

1 Introduction

Languages of epistemic logic [5] have been used to express security properties ([2], [7]) and to analyze protocols ([1], [3], [4], [10], [11]). Yet, it is not clear from any of the standard approaches how beliefs and knowledge should be interpreted in the context of real agents – as the semantics of the systems are typically rather abstract. The typical approaches takes a global view on the modeling and analysis of security critical system. Reachability analysis is usually performed by generating executions in terms of sequences of protocol events, while theorem proving approaches deploy a high level and "ideal" perspective on the epistemic operators. Hence if an agent is equipped with a classical deduction engine, the agent would try to prove infinitely many theorems. In practice this would mean that the agent would use most of its computation resources proving theorems. This is a problem, of course, since the applications and protocols call for concrete answers to real problems. Real agents do not have unlimited reasoning power, and their memory is always bounded. In other words: agents are finitary. Moreover, most of the theorems that are proven in classical normal epistemic logic is entirely irrelevant to agents acting in a security critical environment. Abstract denotational semantics that interprets axiomatizations of *idealized* epistemic capabilities is inappropriate to understand agency. We turn the problem upside down: The belief operator is given a concrete operational interpretation, therefore the epistemic axioms are revised according to the practical needs of the agent. Thus the logic and its semantics approach actual agent behaviour. If agents perform logical deductions *locally*, then two aspects become important: first that the deductions terminate as efficient as possible and second that agents are focused towards obtaining useful beliefs. Since the life-cycle of agents are executed within a tool, termination of the local deductions is a particular important problem to solve.

Trust is one important example of a security property that involves local deductions. *Trust* is currently a large research field. We are only concerned with the straightforward communication-oriented interpretation, introduced by the BAN logic [3].

> An agent a trusts an agent b, if a believes whatever b says is correct.

The part "whatever b says" includes the reference to a second order quantifier. Hence if agent b sends a message to a saying F, then a should be able to infer that a believes F, whatever the sentences F contains.

The paper addresses the following questions: Is it possible to build agents that are capable of making second order inferences? Are there efficient ways to do so? Fortunately the answer is yes, although problems about termination pop up at every corner. A tiny second order language is introduced in Section 2, in order to formalize the notion of trust and other security properties. Then in Section 3, an operational semantics is given in rewriting logic. In Section 4 we explore desirable epistemic inference schemes in the context of term rewriting, and revised temporal versions of the standard axioms are presented.

The minimal logical deduction system presented is the logical kernel of the agents in the simulator PROSA, as depicted in Figure 1. PROSA [8] is a framework for the

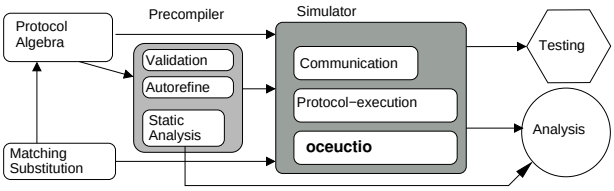

Figure 1. Architecture of PROSA.

specification and static and dynamic analysis of protocols. PROSA is implemented in Maude, which is based on rewriting logic. In this paper we mainly consider the modules Local deduction, and Communication, as depicted in Figure 1.

The concrete and operational interpretation of the belief operator also put new light on epistemic logic. Epistemic logic has been devoted to the investigation of two main interpretations: *implicit* or *ideal* beliefs and the notion of *explicit* or *concrete* beliefs. Neither of these approaches have explored real practical agents performing local deductions.

Finally the investigation served as a successful example of *method integration*, term rewriting is used to calibrate logical axioms. Unlike common approaches, the belief operator is given a concrete interpretation: The deduction engine is focused on the agents capability of performing effective deductions.

2 Formalization of security properties

In this section we define a language for specifying security properties based on temporal epistemic logic, and give some examples of formalizations of security properties like trust, confidentiality and honesty.

2.1 The formal specification language

A second order language [13] for specifying security properties, denoted $\mathscr{L}_S$, can be defined as follows: The *agent terms* include *agent names* and *agent variables*. The agent terms are typically written a, b, c, while the agent *variables* are denoted x, y, $x_1, x_2, \ldots$. Second order variables X, Y are place-holders for arbitrary sentences.

Definition 1 *Let a and b be agent terms, x an agent variable, and X a sentence variable. $\mathscr{L}_S$ is the least language s.t.:*

(i) $a = b$, $\mathsf{Agent}(a)$, $X \in \mathscr{L}_S$.
(ii) *If $\varphi, \psi, \Phi \in \mathscr{L}_S$, then so are the following:*

$\neg\varphi$, $\varphi \to \psi$,	*propositional logic*
$\mathsf{Transmit}(a, b, \varphi)$,	*"a sends φ to b"*
$\mathsf{Bel}_a(\varphi)$,	*"a believes φ"*
$\varphi \,\mathcal{U}\, \psi$,	*"φ holds until ψ holds"*
$\forall x\, \varphi$,	*1. order quantification*
$\forall X\ \Phi$.	*2. order quantification*

Even though the language appears to be very expressive, due to the second order variables and quantifiers, there are some strict limitations: Unlike λ-calculus, general functional application in $\mathscr{L}_S$ is not permitted, like $X(y)$ or $X(Y)$. Since a sentence Φ might contain second order variables, an $\forall X$ instance Φ^{inst} might have degree larger than the original sentence Φ. We say that:

Definition 2 *The minimal syntactic complexity of a sentence φ, denoted $\deg(\varphi)$, is defined by obvious recursion:*

$$
\begin{array}{ll}
(i) & \deg(a = b) = \deg(\mathsf{Agent}(a)) = \deg(X) = 1 \\
(ii) & \deg(\neg\varphi) = \deg(\mathsf{Bel}_a(\varphi)) = \\
 & \qquad \deg(\mathsf{Transmit}(a, b, \varphi)) = \deg(\varphi) + 1 \\
(iii) & \deg(\varphi \to \psi) = \deg(\varphi \,\mathcal{U}\, \psi) = \max(\varphi, \psi) + 1 \\
(iv) & \deg(\forall x\, \varphi) = \deg(\forall X\ \Phi) = \deg(\varphi) + 1
\end{array}
$$

Since there are both first and second order variables, two substitution operators are needed. First order substitution, $\mathrm{sub}(t, x, e)$ reads "replace the variable x with the term t in the expression e". The second order substitution function, denoted $\mathbf{sub}(F, X, \Phi)$, replaces every occurrence of the sentence variable X in the sentence Φ by the sentence F. Both $\mathrm{sub}(t, x, \Phi)$ and $\mathbf{sub}(F, X, \Phi)$ are defined by obvious recursion on $\deg(\Phi)$. Although it is possible to write sentences in $\mathscr{L}_S$ with free first or second order variables, it is prohibited in specifications. Substitution over quantifiers is then defined by $\mathrm{sub}(t, x, \forall y\, \Phi) = \forall y\, (\mathrm{sub}(t, x, \Phi))$ if $x \neq y$ and $\mathrm{sub}(t, x, \forall x\, \Phi) = \forall x\, \Phi$. The definition for second order substitution is similar. The function $\mathrm{SUB}(S, \Phi)$ recursively substitutes a set S consisting of pairs of agent variables and terms $\langle t, x\rangle$, and sentence variables and sentences $\langle F, X\rangle$ into the sentence Φ: It is defined by $\mathrm{SUB}(\varnothing, \Phi) = \Phi$, $\mathrm{SUB}(\langle t, x\rangle \cup S, \Phi) = \mathrm{SUB}(S, \mathrm{sub}(t, x, \Phi))$, $\mathrm{SUB}(\langle F, X\rangle \cup S, \Phi) = \mathrm{SUB}(S, \mathbf{sub}(F, X, \Phi))$.

The propositional connectives $\wedge$ and $\vee$ are definable from $\neg$ and $\to$, and $a \neq b$ iff $\neg(a = b)$. Let G denote a finite set of actual agent names $G = \{a_1, \ldots, a_n\}$, and x be an agent variable. The *next* and *future* operators are definable from until by the equations $\bigcirc\varphi \overset{\text{def}}{\longleftrightarrow} \bot \,\mathcal{U}\, \varphi$ and $\boxed{\mathsf{F}}\ \varphi \overset{\text{def}}{\longleftrightarrow} \top \,\mathcal{U}\, \varphi$ (in the proper future), $\boxdot\, \varphi \overset{\text{def}}{\longleftrightarrow} \varphi \wedge \boxed{\mathsf{F}}\varphi$ (now and always in the future).

2.2 Security properties

Although the core language $\mathscr{L}_S$ presented in definition 1 is small, it has the power to express a large set of high level properties, both properties about single agents, and relations between agents [9].

2.2.1 Trust

The sentence "agent a trusts agent b", which means that the agent a believes everything that b says, is denoted $\mathrm{Trust}(a, b)$. We say that a is the *subject* (trustor) in the relation and b is the *object* (trustee) in the relation. A standard formalization of the relation is:

$$\mathrm{Trust}(a, b) \stackrel{\text{def}}{\longleftrightarrow} \forall X\, (\mathsf{Transmit}(b, a, X) \rightarrow \mathsf{Bel}_a(X)).$$

The agent b might not know or believe that a stands in this relation to her. If b believes this is the case this reads $\mathsf{Bel}_b(\mathrm{Trust}(a, b))$. If a does not trust b, we say that a lacks trust in b, formally $\mathsf{Bel}_a(\neg \mathrm{Trust}(a, b))$.

2.2.2 Confidentiality

An important concept in computer security is the notion of *confidentiality*. A fact φ is *contingent secret* for a group of agents G, written $\mathrm{CSecret}(G, \varphi)$, iff every agent not in the group does not possess φ, formally:

$$\mathrm{CSecret}(G, \varphi) \stackrel{\text{def}}{\longleftrightarrow} \forall x(\mathsf{Agent}(x) \wedge x \notin G \rightarrow \neg\, \mathsf{Bel}_x\,(\varphi)).$$

An even stronger notion is obtained by requiring that a given fact φ remains secret in the future, formalized by $\boxdot \mathrm{CSecret}(G, \varphi)$.

2.2.3 Honesty

An agent is *honest* if the agent believes everything it says:

$$\mathrm{Honest}(a) \stackrel{\text{def}}{\longleftrightarrow} \forall X\, \forall y\, (\mathsf{Transmit}(a, y, X) \rightarrow \mathsf{Bel}_a(X)).$$

Honesty is an example of a *character*. A set of characters constitute a *personality*, in other words: the attitude of the agent.

3 Operational semantics for $\mathscr{L}_P$

Agents communicate with other agents over a network. A message in the network consists of a message content m, the sender's name and the receivers name. If a and b are agent names, and m is the message content we write $\mathsf{msg}\ m\ \mathsf{from}\ a\ \mathsf{to}\ b$. The entities *agents* and *messages* are the only inhabitants in the network model. We introduce the *parallel operator* $\parallel$, and use the notation $o_1 \parallel o_2 \parallel \ldots \parallel o_n$, to express that the entities $o_1, o_2, \ldots, o_n$ coexist concurrently. A rewrite rule $t \longmapsto t'$ can be interpreted as a local transition rule allowing an instance of the term t to evolve into the corresponding instance of the pattern t'. Each rewrite rule describes how a part of a configuration can evolve in one transition step. A *configuration* is a snapshot of a dynamic system evolving. The parallel operator is an Abelian monoid over the set of configurations with an identity element (the empty configuration). The agent is a structure containing four slots:

$$\prec \underbrace{\mathsf{id}}_{\text{agent name}} \mid \underbrace{\mathsf{bel},}_{\text{set of sentences}} \underbrace{\mathsf{pers},}_{\text{personality}} \underbrace{\mathsf{in},\ \mathsf{out}}_{\text{buffers}} \succ$$

where id denotes the *identity* or *name* of the agent, while bel denotes its current *set of beliefs*. In the paper we also consider the interpretation of beliefs as a *bag* of formulas (multiple occurences are permitted), and show how bags and contraction of beliefs do not have any undesirable effects. The beliefs are interpreted extensionally, that is informally we say that $\mathsf{Bel}_{\mathsf{id}}(\varphi)$ if and only if $\varphi \in \mathsf{bel}$. The variable denoted in represents the *in-buffer* - messages waiting for processing in protocol, while out denotes the *out-buffer* - messages intended for transmission.

The *personality* of an agent is given by a set of *characters*. *Honesty*, defined in section 2.2, is an example of a character. Examples of characters that might be possessed by a given agent a include *dishonest* (a always tell the opposite of what a believes), *conscious* (a remembers each interaction), *forgetful* (a forgets each interaction), *verbose* (a tells everybody everything that a believes), *silent* (a is not sending any message) [9].

The personality of the agents determines which actions they prefer to do and which actions they can do. For convenience we assume that every agent is conscious and that attributes not used in a rule are omitted. The decision to send messages is a conscious act of mind, messages are added to the out-buffer of an agent, meaning that it is in a state of being ready for transmission. The rule SEND is given as follows:

$$\begin{array}{l} \prec a \mid \mathsf{bel}, \mathsf{pers}, \mathsf{in}, \mathsf{out} \cup \{\mathsf{Transmit}(a, b, F)\} \succ \\ \longmapsto \\ \prec a \mid\ \mathsf{bel} \cup \{\mathsf{Transmit}(a, b, F)\}, \mathsf{pers}, \mathsf{out} \succ \\ \parallel\ \mathsf{msg}\ F\ \mathsf{from}\ a\ \mathsf{to}\ b \\ \qquad\qquad \text{if } \mathsf{Honest}\,(a) \in \mathsf{pers} \text{ and } F \in \mathsf{bel} \end{array}$$

In the rule for receiving messages, the agent b believes that he has received the message and b also memorizes that a is an agent, as given by the REC_1 rule:

$$\begin{array}{l} \prec b \mid \mathsf{bel}, \mathsf{pers}, \mathsf{in}, \mathsf{out} \succ \parallel\ \mathsf{msg}\ F\ \mathsf{from}\ a\ \mathsf{to}\ b \\ \longmapsto \\ \prec b \mid\ \mathsf{bel} \cup \{\mathsf{Transmit}(a, b, F)\}, \mathsf{pers}, \\ \qquad\qquad \mathsf{in} \cup \{\mathsf{Transmit}(a, b, F)\}, \mathsf{out} \succ \end{array}$$

The in-buffer in represents messages ready for protocol execution. Similar rules can be constructed for the other personalities, described above.

4 Local deduction

The agents can perform the deduction themselves. Agents can be strengthened to be limited theorem provers, denoted *local reasoners*. They can perform deductions during each transition step, and extract as much relevant information out of the belief set as possible. But how is this possible? Rewriting logic assumes that the specifications written are both terminating and confluent. It is not surprising that the main problem about local reasoners is to assure that they stop proving during each transition step. The only way to manage this is requiring that the agents are *logically focused*, that is, they minimize computationally bad strategies for deduction and maximize good strategies for obtaining information. The deduction engine for beliefs are based on the following axioms in epistemic logic:

$$\begin{array}{ll}
\mathsf{Bel}_a(\phi) \wedge \mathsf{Bel}_a(\phi) \rightarrow \mathsf{Bel}_a(\phi) & \text{contraction} \\
\mathsf{Bel}_a(\phi \wedge \psi) \rightarrow \mathsf{Bel}_a(\phi) \wedge \mathsf{Bel}_a(\psi) & \wedge\text{-distribution} \\
\mathsf{Bel}_a(\phi) \wedge \mathsf{Bel}_a(\phi \rightarrow \psi) \rightarrow \mathsf{Bel}_a(\psi) & \text{modus ponens} \\
\mathsf{Bel}_a(\mathsf{Bel}_a(\phi)) \rightarrow \mathsf{Bel}_a(\phi) & \text{reflective awareness}
\end{array}$$

The first axiom is a tautology in propositional logic $\wedge$-contraction: in the context of epistemic logic it express that beliefs might be contracted. The second axiom states that beliefs distributes over conjunction, and is provable in any normal epistemic logic. The third axiom, also denoted (K_{Bel}), is the well-known K-axiom for epistemic logics, which expresses that beliefs are closed under modus ponens. The fourth axiom, denoted 4^C_{Bel}, says that if an agent has reflective awareness of its beliefs, the agent also posses these beliefs. Reflective awareness is rarely if ever mentioned in contexts of epistemic logics. Local theorem proving by agents that are *eager to know*, gives a practical example of how useful this axiom can be. The deductions that involves trust relations to infer new beliefs, have the following pattern:

$$\dfrac{\dfrac{\mathsf{Bel}_a(\mathsf{Bel}_a(\varphi)) \rightarrow \mathsf{Bel}_a(\varphi) \quad [\text{ Reflective awareness }]}{\mathsf{Bel}_a(\varphi)} \quad \dfrac{\mathsf{Bel}_a(\mathsf{Transmit}(b,\, a,\, \varphi)) \quad [\text{ Fresh message }] \qquad \dfrac{\mathsf{Bel}_a(\forall X\ (\mathsf{Transmit}(b,\, a,\, X) \rightarrow \mathsf{Bel}_a(X))) \quad [\text{ Trust Axiom: } \mathrm{Trust}(a,\, b)\ ]}{\mathsf{Bel}_a(\mathsf{Transmit}(b,\, a,\, \varphi) \rightarrow \mathsf{Bel}_a(\varphi))}}{\mathsf{Bel}_a(\mathsf{Bel}_a(\varphi))}}{\mathsf{Bel}_a(\varphi)}$$

Note that the previous deduction, formulated as natural deduction alike inferences, hides an important distinction between the external view, represented by each of the outermost belief operators Bel, and the internal deduction activity inside agents, represented by the arguments of the outermost operators. Hence as considered from inside, the agents might observe:

$$\dfrac{\dfrac{\mathsf{Bel}_a(\varphi) \rightarrow \varphi \quad (3) \qquad \dfrac{\mathsf{Transmit}(b,\, a,\, \varphi) \quad (1) \qquad \dfrac{\forall X\ (\mathsf{Transmit}(b,\, a,\, X) \rightarrow \mathsf{Bel}_a(X)) \quad (2)}{\mathsf{Transmit}(b,\, a,\, \varphi) \rightarrow \mathsf{Bel}_a(\varphi)}\ \mathrm{IA}}{\mathsf{Bel}_a(\varphi)}}{\varphi}$$

The agent a receives a message (1), checks whether it matches a trust relation (2), and if it does, performs modus ponens on the instantiated conditional, and removes the outermost belief (3). The label IA denotes the instantiation algorithm. The previous inferences does not capture local deductions faithfully. The deduction is performed top down. The instantiation of the universal quantifier is not performed blindly, an instantiation is triggered by the incoming message $\mathsf{Transmit}(b, a, \varphi)$: Each quantified sentence is matched with the received message in order to search for a potential instantiation. Reflection is part of the propositional fragment of the deduction engine, and is executed whenever possible.

4.1 Interpreting epistemic deductions

The agents minimize their reasoning by trying to abandon superfluous information and avoid recursive traps for deductions: Idempotence of bel entails that multiple copies of a single sentence are avoided. Conjuncts are broken down when the outermost connective is a belief operator, since beliefs distributes over conjunction ($\wedge$). Note that the framework for local deduction breaks a holy principle in logic that was stressed by Frege: that an implication is not a causal nor temporal relation [6]. In contrast, a local deduction in the PROSA framework is triggered by a communication event and takes one time unit to execute:

$$\begin{array}{ll}
\mathsf{Bel}_a(\varphi) \wedge \mathsf{Bel}_a(\varphi) \rightarrow \bigcirc\,(\mathsf{Bel}_a(\varphi)) & (\mathrm{contr}^{\mathrm{TIME}}) \\
\mathsf{Bel}_a(\varphi \wedge \psi) \rightarrow \bigcirc\,(\mathsf{Bel}_a(\varphi) \wedge \mathsf{Bel}_a(\psi)) & (\wedge\mathrm{distr}^{\mathrm{TIME}})
\end{array}$$

In the semantics idempotence of beliefs follows from general idempotence of every set. The distribution of Bel over conjunctions is interpreted as distribution on the highest level of the belief set of the agent:

$$\begin{array}{ll}
\prec a \mid \mathsf{bel} \cup \{\varphi, \varphi\} \succ \ \longmapsto \prec a \mid \mathsf{bel} \cup \{\varphi\} \succ & (\mathrm{C}_\wedge) \\
\prec a \mid \mathsf{bel} \cup \{\varphi \wedge \psi\} \succ \longmapsto \prec a \mid \mathsf{bel} \cup \{\varphi\} \cup \{\psi\} \succ & (\mathrm{D}_\wedge)
\end{array}$$

Removing superfluous information does not make the agent smarter. However, new information can be obtained by using modus ponens on implications closed under the belief operator in addition to new messages received by the agent. Formalized modus ponens $\mathrm{K}_{\mathrm{Bel}}$ has been presented as one of the guiding axioms. By modifying the axiom slightly using propositional logic and introduce the next operator, it gets the shape of the sentence $(\mathrm{K}^{\mathrm{TIME}}_{\mathrm{Bel}})$:

$$\mathsf{Bel}_a(\varphi) \wedge \mathsf{Bel}_a(\varphi \rightarrow \psi) \rightarrow \bigcirc(\mathsf{Bel}_a(\varphi) \wedge \mathsf{Bel}_a(\psi))$$

Now we have stated explicitly what the target beliefs of the agent should be. But this is still not enough. If the agent a trusts an agent b with respect to the sentence φ, this would read: $\mathsf{Bel}_a(\mathsf{Transmit}(b, a, \varphi) \rightarrow \mathsf{Bel}_a(\varphi))$. But axiom ($\mathrm{K}^{\mathrm{TIME}}_{\mathrm{Bel}}$) only gives $\mathsf{Bel}_a(\mathsf{Transmit}(b, a, \varphi)) \stackrel{\text{next}}{\Longrightarrow} \mathsf{Bel}_a(\ \mathsf{Bel}_a(\varphi))$, which is not strictly informative. It is more appropriate to claim that a trusts b should entail: $\mathsf{Bel}_a(\mathsf{Transmit}(b, a, \varphi)) \rightarrow \mathsf{Bel}_a(\varphi)$. Hence the timed version of reflective awareness of beliefs, given by

$$\mathsf{Bel}_a(\mathsf{Bel}_a(\varphi)) \rightarrow \bigcirc\ \mathsf{Bel}_a\ (\varphi) \quad (4^{C\,\mathrm{TIME}}_{\mathrm{bel}}),$$

would be sufficient to solve the problem. In the mapping of $\mathrm{K}^{\mathrm{TIME}}_{\mathrm{Bel}}$ to its corresponding rule, we remove the implication $\varphi \rightarrow \psi$ to avoid obvious non-terminating execution. Thus *economical modus ponens* (emp) and *reflective awareness* (refa) are characterized by the two rules:

$$\begin{array}{ll} \prec a \mid \mathsf{bel} \cup\{\varphi, \varphi \rightarrow \psi\} \succ \longmapsto \prec a \mid \mathsf{bel} \cup\{\varphi, \psi\} \succ & (\mathrm{emp}) \\ \prec a \mid \mathsf{bel} \cup\{\mathsf{Bel}_a(\phi)\} \succ \longmapsto \prec a \mid \mathsf{bel} \cup\{\phi\} \succ & (\mathrm{refa}) \end{array}$$

Let T_1 denote the previous theory $T_1 = \{\mathrm{SEND}, \mathrm{REC}_1, \mathrm{C}_\wedge, \mathrm{D}_\wedge, \mathrm{emp}, \mathrm{refa}\}$. Then we have that

Theorem 1 *Any specification over $\mathscr{L}_S$ interpreted in T_1 is terminating.*

Proof: Suppose that S is an arbitrary specification, containing n agents. We prove the theorem by induction on the number of messages in the output buffer. Without loss of generality we might suppose that S does not contain messages in the network, since if S' contained m messages we assume that they are received by the agents in question by asserting that $S' \longmapsto^* S$. Messages where the receiver is not contained in S, do not play any role for the argument, since these messages are not processed. Let $\mathscr{A}$ denote the set of agents in S, and let $\mathrm{Out}(S)$ denote the function that measures the total number of messages in the out-buffers of the agents in S

$$\mathrm{Out}(S) = \sum_{\prec a \mid \mathsf{bel}, \mathsf{out} \succ \in \mathscr{A}} \mid \mathsf{out} \mid.$$

Induction basis: Then there are no messages in any out-buffer, hence $\mathrm{Out}(S) = 0$. Every agent a in S has initially a finite set of sentences: $\mathsf{bel} = \{\varphi \mid \mathsf{Bel}_a\ (\varphi)\}$, $\mid \mathsf{bel} \mid = n$. A measure for termination (weight function) can be defined by summation of the syntactic complexity of the belief set, as given by the function: $\mathrm{sdeg}(\mathsf{bel})$:

$$\mathrm{sdeg}(\mathsf{bel}) = \sum_{\varphi \in \mathsf{bel}} \deg(\varphi) \times 2^{\deg(\varphi)}.$$

Obviously $\mathrm{sdeg}(\mathsf{bel}) \geqslant 0$. By an induction on the length of the execution sequences from S we can prove that Since $\mathrm{Out}(S) = 0$, there are no messages to be processed, thus the belief set is never extended with new beliefs about transmissions. Each of the epistemic deduction rules R in T_1 reduces the weighting of bel:

$$\prec a \mid \mathsf{bel}, \mathsf{out} \succ \parallel \mathscr{C} \longmapsto^R \prec a \mid \mathsf{bel}', \mathsf{out} \succ \parallel \mathscr{C} \\ \Longrightarrow \mathrm{sdeg}(\mathsf{bel}) > \mathrm{sdeg}(\mathsf{bel}').$$

Consider an arbitrary sentence φ in bel. If φ is an atomic sentence an atomary sentence, Consider (emp): $\prec a \mid \mathsf{bel} \cup\{\varphi, \varphi \rightarrow \psi\} \succ \longmapsto \prec a \mid \mathsf{bel} \cup\{\varphi, \psi\} \succ$. We must show that $\mathrm{sdeg}(\mathsf{bel} \cup \{\varphi, \varphi \rightarrow \psi\}) > \mathrm{sdeg}(\mathsf{bel} \cup \{\varphi, \psi\})$, which is established by showing that $\mathrm{sdeg}(\{\varphi, \varphi \rightarrow \psi\}) > \mathrm{sdeg}(\{\varphi, \psi\})$. Then we have both that $\mathrm{sdeg}(\{\varphi, \varphi \rightarrow \psi\}) = \deg(\varphi) \times 2^{\deg(\varphi)} + (\max(\deg(\varphi), \deg(\psi)) + 1) \times 2^{\max(\deg(\varphi), \deg(\psi))+1}$, and at the same time we have $\mathrm{sdeg}(\{\varphi, \psi\}) = \deg(\varphi) \times 2^{\deg(\varphi)} + \deg(\psi) \times 2^{\deg(\psi)}$. The largest value of $\mathrm{sdeg}(\{\varphi, \psi\})$ occurs when $\deg(\varphi) = \deg(\psi)$, suppose therefore that the latter is the case, and $\deg(\varphi) = m$. Then $\mathrm{sdeg}(\{\varphi, \varphi \rightarrow \psi\}) = m \times 2^m + (m+1) \times 2^{m+1} = m \times 2^m + 2^{m+1} + m \times 2^{m+1}$ and $\mathrm{sdeg}(\{\varphi, \psi\}) = m \times 2^m + m \times 2^m = m \times 2^{m+1}$, which proves the result. $\max(\deg(\varphi), \deg(\psi)) = \deg(\varphi)$.

The verifications for the other rules are similar and are therefore omitted.

Consider the induction step: Suppose that the specification S that contains n output messages, is extended into a specification $S^\star$ such that one agent a has one extra output message to process, in other words $\mathrm{Out}(S^\star) = n + 1$. Suppose that this message is $m = \mathsf{Transmit}(a, b, F)$. By induction hypothesis S is terminating. Suppose without loss of generality that we emulate the terminating rewrites of S inside $S^\star$, such that $S^\star \longmapsto^* S''$ and the only possible rewrite in S'' is the transmission of m. If there exists an agent b that can receive m, and a can send m, then:

$$S^\star \longmapsto^* S'' \longmapsto^{\mathrm{SEND}} S_1 \longmapsto^{\mathrm{REC}} S_2,$$

otherwise S'' itself is a final state.

In S_2, the agent b has received the message. By induction hypothesis the only possible deductions may be performed using message m. Suppose further that b trusts a, otherwise S_2 is already a final state. Then by at most $2^{\deg(F)} - 1$ applications of the epistemic deduction rules, a final state $S_2 \longmapsto^* S_3$ is reached. ■

4.2 The standard epistemic axioms

The local deduction engine balances on a thin line of termination. Consider the standard axioms of epistemic logic: Consider the standard axioms of epistemic logic are the *epistemic modus ponens* (K_{bel}), *consistency* (D_{bel}), *positive introspection* (4_{bel}) and *negative introspection* (5_{bel}):

$$\mathsf{Bel}_a(\varphi) \wedge \mathsf{Bel}_a(\varphi \rightarrow \psi) \rightarrow \mathsf{Bel}_a(\psi) \quad (K_{\mathrm{bel}})$$
$$\mathsf{Bel}_a(\phi) \rightarrow \neg\, \mathsf{Bel}_a\,(\neg\phi) \quad (D_{\mathrm{bel}})$$
$$\mathsf{Bel}_a(\varphi) \rightarrow \mathsf{Bel}_a(\mathsf{Bel}_a(\varphi)) \quad (4_{\mathrm{bel}})$$
$$\neg\, \mathsf{Bel}_a(\phi) \rightarrow \mathsf{Bel}_a(\neg\, \mathsf{Bel}_a\,(\phi)) \quad (5_{\mathrm{bel}})$$

The axiom D_{bel} expresses that the agent a's belief set is consistent, in other words $D_{\mathrm{bel}} \rightarrow \neg(\mathsf{Bel}_a(\phi) \wedge \mathsf{Bel}_a(\neg\phi))$. The axiom of consistency is not included as a guiding axioms, since agents can have contradictory beliefs. A typical example is when an agent a trusts two agents b and c, but receives contradictory assertions from b and c. The negative introspection axiom (5_{bel}) is not coherent with the requirement that agents act only based on explicit beliefs. Axiom (5_{bel}), and any operation interpretation is abandoned since an agent should certainly *not* have explicit beliefs about everything it does not believe.

Observation 1 *The distribution axiom for implication and the axiom of positive introspection both cause non-terminating specifications.*

Reflective awareness works opposite of the classical axioms of epistemic logic [5], the axioms of solipsisms, positive and negative introspection. Together with economical modus ponens, reflective awareness *focus* the agent's beliefs.

4.3 Weak second order instantiation

The single agent properties and the trust relation were formulated as second order sentences. Thus to be able to deduce sentences from the properties, those sentences must be instantiated. Each time a new sentence enters a's beliefs, the sentences that contains second order quantifiers are instantiated and the potential antecedents are matched towards the recently added belief. Several of the security properties described in this paper are sentences on the normal form $\mathbf{Q}\,(F \rightarrow G)$, where $\mathbf{Q}$ is a quantifier prefix consisting of first or second order universal quantifiers. For convenience we say that the quantifier prefix might take only three forms, either $\mathbf{Q} = (\forall X)(\forall y_1)\ldots(\forall y_n)$, $\mathbf{Q} = (\forall X)$, or $\mathbf{Q} = (\forall y_1)\ldots(\forall y_n)$. We write $\mathbf{Q}^1$ if there are only first order or no quantification in the prefix. Let $\vec{y}$ denote a sequence (possibly empty) of first order variables, $\vec{y} = y_1, \ldots, y_n$. Then we can characterize particularly simple classes of sentences, the *nice second order*, the *second order simple*, and the *simple quantified* sentences:

Definition 3 *The* nice second order *sentences $\mathscr{L}_{2O}$, is the least set such that:*

(i) $\mathsf{Agent}(a) \in \mathscr{L}_{2O}$
(ii) *If* $\psi \in \mathscr{L}_S$ *then* $\mathsf{Bel}_a\,(\psi), \mathsf{Transmit}(a, b, \psi) \in \mathscr{L}_{2O}$
(iii) *If* $F(X, \vec{y})$, $G(X, \vec{y}) \in \mathscr{L}_{2O}$, *then* $\forall X \forall \vec{y}(F(X, \vec{y}) \rightarrow G(X, \vec{y})) \in \mathscr{L}_{2O}$
(iv) *If* $F(x, \vec{y})$, $G(x, \vec{y}) \in \mathscr{L}_{2O}$, *then* $\forall x \forall \vec{y}(F(x, \vec{y}) \rightarrow G(x, \vec{y})) \in \mathscr{L}_{2O}$

A sentence φ is *second order simple*, denoted $\varphi \in \mathscr{L}_\forall$, if φ is a closed sentence and $\varphi \in \mathscr{L}_{2O}$. If $\varphi \in \mathscr{L}_\forall$ and $\varphi = \mathbf{Q}\,(F \rightarrow G)$, where $\mathbf{Q}$ is a quantifier prefix, then φ is a *simple quantified* sentence. Obviously $\mathscr{L}_\forall \subseteq \mathscr{L}_{2O} \subseteq \mathscr{L}_S$. The rule for receiving messages is then changed, such that the agent extracts as much logical information from the message and the belief-set as possible, as in the rule REC_2:

$$\begin{array}{l} \prec b \,|\, \mathsf{bel}, \mathsf{pers}, \mathsf{in}, \mathsf{out} \succ \;\|\; \mathsf{msg}\;\; F \;\; \mathsf{from}\;\; a \;\; \mathsf{to}\;\; b \\ \longmapsto \\ \prec b \,|\, \mathsf{bel} \cup \{\mathsf{Transmit}(a, b, F)\} \cup \{\mathbb{e}(F, \mathsf{bel})\}, \mathsf{pers}, \\ \qquad \mathsf{in} \cup \{\mathsf{Transmit}(a, b, F)\}, \mathsf{out} \succ \end{array}$$

Thus define $T_2 = (T_1 - \{\mathrm{REC}_1\}) \cup \{\mathrm{REC}_2\}$. The rewriting theory T_2 is designed in order to be able to perform correct deductions of trust according to the inference schemes introduced in the beginning of this section. If F is a sentence and bel is a set of beliefs, then the function $\mathbb{e}(F, \mathsf{bel})$ returns the instantiations of universally quantified sentences in bel, such that F matches the antecedents in bel. The boolean function $\mathbb{M}(F', F)$ decides if F' may match F: that is, F' match F iff they are equal except that some of the terms or some of the sentences in F' correspond to agent variables and sentence variables in F. The function $\mathbb{S}(F', F)$ performs the matching of the terms in F' with variables in F, resulting in a set of matching pairs. The next example illustrates a successful matching:

$$\begin{array}{l} \mathbb{S}(\mathsf{Transmit}(\mathrm{Alice}, \mathrm{Bob}, \mathsf{Bel}_{\mathrm{Carol}}(\mathsf{Agent}(\mathrm{Bob}))), \\ \quad \mathsf{Transmit}(x, y, \mathsf{Bel}_{\mathrm{Carol}}(X))) \\ \qquad = \{\langle \mathrm{Alice}, x\rangle, \langle \mathrm{Bob}, y\rangle, \langle \mathsf{Agent}(\mathrm{Bob}), X\rangle\} \end{array}$$

Definition 4 *The* logical expansion *of a set of sentences S with respect to a given sentence F, denoted $\mathbb{e}(F, S)$, is defined by:*

(i) $\mathbb{e}(F, \varnothing) = \{F\}$
(ii) $\mathbb{e}(F, \{\mathbf{Q}\,(\varphi \rightarrow \psi)\} \cup S) =$
$\{\mathrm{SUB}(\mathbb{S}(F, \varphi), \varphi \rightarrow \psi)\} \cup \mathbb{e}(F, S)$
if $\mathbf{Q}\,(\varphi \rightarrow \psi)$ *is a simple quantified sentence, and* $\mathbb{M}(F, \varphi)$, *else* $\mathbb{e}(F, \{\varphi\} \cup S) = \mathbb{e}(F, S)$.

The REC_2: rule contains a potential recursion trap $\{\mathbb{e}(F, \mathsf{bel})\}$, hence it is not obvious that the deduction terminates. Fortunately we can prove that

Theorem 2 *Second order universal instantiation is terminating for specifications interpreted in T_2.*

Confidentiality is *not* a simple quantified sentence. This is not a big problem since most security policy interactions can be performed by wrapping such sentences inside beliefs or transmissions. Consider a scenario where Alice trusts Bob, and Bob wants to enforce Alice to believe that this trust relation is a secret: $\mathsf{Bel}_{\mathrm{Alice}}(\mathrm{Trust}(\mathrm{Alice}, \mathrm{Bob}))$ (A_1),

and

$$\mathsf{Transmit}(\mathrm{Bob}, \mathrm{Alice}, \mathrm{CSecret}(\{\mathrm{Alice}, \mathrm{Bob}\}, \mathrm{Trust}(\mathrm{Alice}, \mathrm{Bob}))). \quad (A_2)$$

Both sentences (A_1) and (A_2) are simply quantified sentences. When Alice receives (A_2), hence $\mathsf{Bel}_{\mathrm{Alice}}(A_2)$, she can perform the required second order instantiation according to Definition 4, hence we can conclude that $\mathsf{Bel}_{\mathrm{Alice}}(\mathrm{CSecret}(\{\mathrm{Alice}, \mathrm{Bob}\}, \mathrm{Trust}(\mathrm{Alice}, \mathrm{Bob})))$.

4.4 Recursive second order instantiation

When the agent a performs an instantiation, new consequences might occur in a's mind. Let us consider the naive agent Ed. Ed makes himself a disciple of every agent y that claims to Ed that y is honest, that is, Ed establishes a trust relation towards every such agent y (clause B_1). Ed also thinks that every agent that he trust, trusts him as well (clause B_2).

$$\mathsf{Bel}_{\mathrm{Ed}}(\forall y(\mathsf{Transmit}(y, \mathrm{Ed}, \mathrm{Honest}(y)) \to \mathsf{Bel}_{\mathrm{Ed}}(\mathrm{Trust}(\mathrm{Ed}, y)))) \quad (B_1)$$
$$\mathsf{Bel}_{\mathrm{Ed}}(\forall y(\mathsf{Bel}_{\mathrm{Ed}}(\mathrm{Trust}(\mathrm{Ed}, y)) \to \mathsf{Bel}_{\mathrm{Ed}}(\mathrm{Trust}(y, \mathrm{Ed})))) \quad (B_2)$$

The sentences B_1 and B_2 indicate two kinds of weaknesses of the rules presented so-far, due to *eager reflection* and *recursive expansion*. Consider first eager reflection: The antecedent in B_2, $\mathsf{Bel}_{\mathrm{Ed}}(\mathrm{Trust}(\mathrm{Ed}, y))$ might not match any sentence in Ed's beliefs, because the required match $\mathsf{Bel}_{\mathrm{Ed}}(\mathsf{Bel}_{\mathrm{Ed}}(\mathrm{Trust}(\mathrm{Ed}, y)))$ might have been reduced by reflection. We therefore introduce an axiom removing the outermost belief, denoted *synchronous reflection* (SR):

$$\mathsf{Bel}_a(\mathsf{Bel}_a(\varphi) \to \psi) \to \mathsf{Bel}_a(\varphi \to \psi) \quad (\mathrm{SR})$$

The axiom is given an operational interpretation by

$$\mathsf{Bel}_a(\mathsf{Bel}_a(\varphi) \to \psi) \to \bigcirc\, \mathsf{Bel}_a\,(\varphi \to \psi) \quad (\mathrm{SR}^{\mathrm{TIME}})$$

The corresponding rule to be added to the rewrite theory, is denoted (ar):

$$\prec a \mid \mathsf{bel} \cup \{\mathsf{Bel}_a(\varphi) \to \psi\} \succ \longmapsto \prec a \mid \mathsf{bel} \cup \{\varphi \to \psi\} \succ$$

While (SR) and $(\mathrm{SR}^{\mathrm{TIME}})$ could be a bit difficult to justify, the rewrite (ar) has a natural interpretation: The antecedent belief in the term to be rewritten is just unnecessary. Does $T_1 \cup \{\mathrm{ar}\}$ and $T_2 \cup \{\mathrm{ar}\}$ terminate? Fortunately the answer is yes:

Theorem 3 *Both $T_1 \cup \{\mathrm{ar}\}$ and $T_2 \cup \{\mathrm{ar}\}$ give terminating specifications.*

Yet the problem with the naive disciple is not solved by the epistemic inference in propositional logic, we must get behind the quantifier barrier:

$$\mathsf{Bel}_a(\mathbf{Q}\,(\mathsf{Bel}_a(\varphi) \to \psi)) \to \mathsf{Bel}_a(\mathbf{Q}\,(\varphi \to \psi)) \quad (\mathrm{SR}_\forall)$$

Hence the corresponding rewrite rule is given by the following:

$$\prec a \mid \mathsf{bel} \cup \{\mathbf{Q}\,(\mathsf{Bel}_a(\varphi) \to \psi)\} \succ \longmapsto \prec a \mid \mathsf{bel} \cup \{\mathbf{Q}\,(\varphi \to \psi)\} \succ \quad (\mathrm{ar}_\forall)$$

Corollary 1 *Both $T_1 \cup \{\mathrm{ar}, \mathrm{ar}_\forall\}$ and $T_2 \cup \{\mathrm{ar}, \mathrm{ar}_\forall\}$ are terminating.*

Consider recursive expansion: Ed is only capable of performing his correct deductions if the logical expansion is performed recursively over the new sentences produced. New sentences occur in the mind of the agent through communication (REC_2) or by deduction. Suppose that Ed receives a message from Alice of the required kind: $\mathsf{Transmit}(\mathrm{Alice}, \mathrm{Ed}, \mathrm{Honest}(\mathrm{Alice}))$, or written explicitly:

$$\mathsf{Transmit}(\mathrm{Alice}, \mathrm{Ed}, \forall X \forall z(\mathsf{Transmit}(\mathrm{Alice}, z, X) \to \mathsf{Bel}_{\mathrm{Alice}}(X))).$$

If Ed applies the old deduction rules presented in the beginning of section 4.1, then Ed is not able to deduce $\mathrm{Trust}(\mathrm{Alice}, \mathrm{Ed})$. One instantiation is certainly not enough, the instantiation must be performed recursively on the deduced formulas. Hence the epistemic deduction rules in the rewrite theory T_2 are replaced by the rules

$$\prec a \mid \mathsf{bel} \cup \{\varphi \wedge \psi\} \succ \longmapsto \prec a \mid \mathsf{bel} \cup \{\varphi, \psi\} \cup \mathbb{e}(\varphi, \mathsf{bel}) \cup \mathbb{e}(\psi, \mathsf{bel}) \succ \text{ if } \varphi \notin \mathsf{bel} \text{ or } \psi \notin \mathsf{bel} \quad (\mathrm{D}^{\wedge}_{\mathrm{SO}})$$

$$\prec a \mid \mathsf{bel} \cup \{\varphi, \varphi \to \psi\} \succ \longmapsto \prec a \mid \mathsf{bel} \cup \{\varphi, \psi\} \cup \mathbb{e}(\psi, \mathsf{bel}) \succ \text{ if } \psi \notin \mathsf{bel} \quad (\mathrm{emp}_{\mathrm{SO}})$$

$$\prec a \mid \mathsf{bel} \cup \{\mathsf{Bel}_a(\varphi)\} \succ \longmapsto \prec a \mid \mathsf{bel} \cup \{\varphi\} \cup \mathbb{e}(\varphi, \mathsf{bel}) \succ \text{ if } \varphi \notin \mathsf{bel} \quad (\mathrm{refa}_{\mathrm{SO}})$$

The idea behind the rules $(\mathrm{D}^{\wedge}_{\mathrm{SO}}, \mathrm{emp}_{\mathrm{SO}}, \mathrm{refa}_{\mathrm{SO}})$ is that each expand the freshly created sentences to check whether they might trigger any interesting instantiation of a second order simple sentence. It is important that the previous deductions performed by the agents are preserved within the second order extension. Hence contraction $(\mathrm{C}_\wedge)$ is included and the remaining axioms are interpreted by

$$\prec a \mid \mathsf{bel} \cup \{\varphi \wedge \psi\} \succ \longmapsto \prec a \mid \mathsf{bel} \succ \text{ if } \varphi \in \mathsf{bel} \text{ and } \psi \in \mathsf{bel} \quad (\mathrm{D}^{\wedge}_{\mathrm{PL}})$$

$$\prec a \mid \mathsf{bel} \cup \{\varphi, \varphi \to \psi\} \succ \longmapsto \prec a \mid \mathsf{bel} \cup \{\varphi\} \succ \text{ if } \psi \in \mathsf{bel} \quad (\mathrm{emp}_{\mathrm{PL}})$$

$$\prec a \mid \mathsf{bel} \cup \{\mathsf{Bel}_a(\varphi)\} \succ \longmapsto \prec a \mid \mathsf{bel} \succ \text{ if } \varphi \in \mathsf{bel} \quad (\mathrm{refa}_{\mathrm{PL}})$$

The rules $(\mathrm{D}^{\wedge}_{\mathrm{PL}}, \mathrm{emp}_{\mathrm{PL}}, \mathrm{refa}_{\mathrm{PL}})$ are motivated by the need to resolve delayed inferences: In the interpretation of "beliefs as bag", the latter rules condense the application of two rules in one: $\mathrm{D}^{\wedge}_{\mathrm{PL}} = (\mathrm{D}_\wedge, \mathrm{C}_\wedge)$. The full second order rewrite theory including quantification T_{SO} is given by the rules

$$T_{\mathrm{SO}} = \{\mathrm{SEND}, \mathrm{REC}_2, \mathrm{C}, \mathrm{D}^{\wedge}_{\mathrm{PL}}, \mathrm{emp}_{\mathrm{PL}}, \mathrm{refa}_{\mathrm{PL}}, \mathrm{D}^{\wedge}_{\mathrm{SO}}, \mathrm{emp}_{\mathrm{SO}}, \mathrm{refa}_{\mathrm{SO}}\}.$$

The requirement that each deduced sentence is not contained in bel, as in case of $\mathrm{D}^{\hat{}}_{\mathrm{SO}}$, $\mathrm{emp}_{\mathrm{SO}}$, and $\mathrm{refa}_{\mathrm{SO}}$ is necessary since without the requirements the rules would be trivially non-terminating. Unfortunately most specifications will not terminate, even for fair executions:

Observation 2 *There are specifications interpreted in T_{SO}, such that second order universal instantiation is non-terminating.*

Proof: Suppose that a, b, and c denote three agents in a configuration $\mathscr{C}$. Assume furthermore that c believes that a trusts c, that c recently sent a message to a, and finally that c believes that a and b have reflective beliefs about each other. The assumptions are specified as follows:

$$
\begin{array}{ll}
(1) & \mathsf{Bel}_c(\mathrm{Trust}(a,c)) \\
(2) & \mathsf{Bel}_c(\mathsf{Transmit}(c,a,\varphi)) \\
(3) & \mathsf{Bel}_c(\forall X(\mathsf{Bel}_a(X) \rightarrow \mathsf{Bel}_b(\mathsf{Bel}_a(X)))) \\
(4) & \mathsf{Bel}_c(\forall X(\mathsf{Bel}_b(X) \rightarrow \mathsf{Bel}_a(\mathsf{Bel}_b(X)))).
\end{array}
$$

Then any fair execution of $\mathscr{C}$ will be non-terminating: From (1) and (2) by expansion and $\mathrm{emp}_{\mathrm{SO}}$, we have $\mathsf{Bel}_c(\mathsf{Bel}_a(\varphi))$ and expansion of $\mathsf{Bel}_a(\varphi)$ gives the instance $\mathsf{Bel}_b(\mathsf{Bel}_a(\varphi))$ from (3). Yet another expansion according to $\mathrm{emp}_{\mathrm{SO}}$ on (4) gives $\mathsf{Bel}_a(\mathsf{Bel}_b(\mathsf{Bel}_a(\varphi)))$, and so on. Since execution is required to be fair, each expansion can be performed eventually. ■

The epistemic axiom 4_{bel} that might give non-terminating specification is definable by the second order formula $\forall X(\mathsf{Bel}_a(X) \rightarrow \mathsf{Bel}_a(\mathsf{Bel}_a(X)))$. If we assume that executions might be *unfair*, it could happen that 4_{bel} is always applied. Then there could be infinitely many instantiations of the $\forall X$ quantifier, resulting in an infinite deduction.

5 Conclusion

Modal logic is commonly regarded as powerful with respect to expressibility, but unfeasible from a practical point of view. Our case study about local deductions of trust supports this: standard axioms have been excluded and a couple of new reflection axioms was added to the theory. We have shown how agents can perform logical inferences and universal instantiations, based on their trust relations. Yet, in order to be able to perform inferences based on more intricate security properties, a recursive instantiation algorithm is required. Unfortunately, it is difficult to know whether a set of beliefs is closed under finite deductions or whether it is non-terminating in the general case. Although the current implementation covers several standard examples, there are several open problems: The two most important are extend the logic to include multi-agent reasoning and find the exact border of termination for local theorem provers.

References

[1] Abadi, M., Burrows, M., Lampson, B., and Plotkin, G. A calculus for access control in distributed systems. *ACM Transactions on Programming Languages and Systems*, 15(4):706–734, September 1993.

[2] Bieber, P. and Cuppens, F. Expressions of Confidentiality Policies with Deontic Logic. In *Deontic Logic in Computer Science: Normative System Specification*, pages 103–123. John Wiley and Sons Ltd, 1993.

[3] Burrows, B., Abadi, M., and Needham, R. A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, February 1990.

[4] M. Cohen and M. Dam. Logical omniscience in the semantics of BAN logic. In *Proc. FCS'05*, 2005.

[5] Moses Y. Fagin R., Halpern J.Y. and Vardi. *Reasoning about knowledge*. The MIT Press, 1995.

[6] Frege, G. *Begriffschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*. Halle: L. Nebert, 1879. Available in several translations.

[7] Glasgow, J. and Macewen, G. A Logic for Reasoning About Security. *ACM Transactions on Computer Systems*, 10(3):226 – 264, August 1992.

[8] Hagalisletto, A. M. Attacks are Protocols Too. In *Proc. of The Second Int. Conf. on Availability, Reliability and Security (IEEE ARES 2007)*, pages 1197 – 1206.

[9] Hagalisletto, A. M. and Haugsand, J.. A Formal Language for Specifying Security Properties. In *Proc. for the Workshop on Specification and Automated Processing of Security Requirements - SAPS'04*. Austrian Computer Society, 2004.

[10] Lowe, G. Breaking and Fixing the Needham-Schroeder Public-Key Protocol Using FDR. In *Proc. of the Second Int. Workshop on Tools and Algorithms for Construction and Analysis of Systems*, pages 147–166. Springer-Verlag, 1996.

[11] Paulson, L. The inductive approach to verifying cryptographic protocols. *Journal of Computer Security*, 6(1–2):85–128, 1998.

[12] Fagin R. and Halpern J.Y. Belief, Awareness and Limited Reasoning. *Artificial Intelligence*, 34:39–76, 1988.

[13] Shapiro, S. *Foundations without Foundationalism: A Case Study for Second Order Logic*. Oxford Science Publication, 1991.