Automatic Verification of Epistemic Specifications under Convergent Equational Theories

Ioana Boureanu
Ecole Polytechnique Federal de Lausanne
ioana.boureanu@epfl.ch

Andrew V. Jones
Department of Computing
Imperial College London
andrew.jones@ic.ac.uk

Alessio Lomuscio
Department of Computing
Imperial College London
a.lomuscio@ic.ac.uk

ABSTRACT

We present a methodology for the automatic verification of multi-agent systems against temporal-epistemic specifications derived from higher-level languages defined over convergent equational theories. We introduce a modality called *rewriting knowledge* that operates on local equalities. We discuss the conditions under which its interpretation can be approximated by a second modality that we introduce called *empirical knowledge*. Empirical knowledge is computationally attractive from a verification perspective. We report on an implementation of a technique to verify this modality with the open source model checker MCMAS. We evaluate the approach by verifying multi-agent models of electronic voting protocols automatically extracted from high-level descriptions.

Categories and Subject Descriptors

D.4.6 [**Security and Protection**]: Verification

General Terms

Security, Verification

Keywords

Epistemic Logic, Equational Rewriting, Model Checking

1. INTRODUCTION

Over the past decade there has been increased interest in developing methodologies for the verification of multi-agent systems (MAS). An approach that has been shown effective is that of symbolic model checking [15, 18] for MAS specified in semantics for temporal-epistemic logic [12]. This has been effectively used in a number of practical applications, including autonomous underwater vehicles [11] and cryptographic protocols [23].

A clear advantage of MAS-based approaches using temporal-epistemic logic is the intuitiveness of the resulting specifications to be checked. Concepts emerging from the MAS community are now being exported to other close disciplines that increasingly see the benefit of using powerful, expressive languages.

One of these areas is *security*. It has long been recognised [5] that cryptographic protocols can benefit from specifications in which knowledge-based concepts feature prominently. Concepts such as anonymity, privacy and non-repudiation can be both naturally and powerfully expressed in epistemic languages. Influential works in the area include the formulation of secrecy by Halpern *et al.* [14], advances on algorithmic knowledge [20] and the epistemic modelling of unlinkability [23]. These have found applications in MAS in a variety of ways, including attack detectability [4].

Still, fundamental problems remain. Firstly, the indistinguishability relations to be used when interpreting the knowledge modalities need to account for the cryptographic primitives used in the messages exchanged. For instance, the set of indistinguishable states should be computed by taking into account the agent's ability to decipher a given message. While some approaches (e.g., [7]) support cryptographic primitives such as encryption and decryption, existing approaches fall short of addressing the more general classes including digital signatures and bit-commitments. Yet, these primitives are prominent in several classes of protocols, e.g., e-voting or zero-knowledge.

Secondly, little or no support for cryptography-inspired modalities is currently provided in existing tools. An extension to MCMAS [15] that caters for explicit knowledge exists [16], but we are unaware of any model checker supporting epistemic modalities for cryptographic concepts, or, indeed, other application-driven epistemic modality of use in many MAS settings. In fact, recent approaches [4] have been restricted to protocols in which receivers can decode all messages down to their atomic constituents immediately upon their receipt. This assumption is not natural in many settings including e-voting, where principals are often only able to decipher messages only at the end of a run.

In this paper we develop an approach aimed at overcoming these limitations. Specifically, in Section 2 we define a novel epistemic modality that is interpreted with respect to *a general equational theory* defining the system. This differs from the standard approach in which the agents' knowledge is interpreted on the equality of the local states. The high computational cost of deducing equivalence under equational theories has been previously discussed [8]. Thus, in Section 3, we put forward a computationally efficient approximation. Section 4 discusses the implementation of this revised modality on top of the model checker MCMAS. In Section 5 we evaluate the techniques presented by verifying e-voting protocols modelled as MAS as per the formalism developed in Section 2. We discuss the results and conclude in Section 6.

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), 4-8 June 2012, Valencia, Spain.

2. A TEMPORAL-EPISTEMIC LOGIC FOR SECURITY PROTOCOLS

In this section we introduce a MAS-based semantics for an epistemic logic under convergent equational theories.

2.1 Preliminaries

We assume familiarity with the concepts presented in this subsection; the following is intended to fix the notation only.

Interpreted Systems. The interpreted systems (IS) formalism [19] is a MAS-based semantics for temporal-epistemic logic (CTLK) [12]. We assume a set $Ag = \{1, \ldots, n\}$ of agents and an *Environment* denoted by *Env*. An agent i is described by a set L_i of possible *local states*, a set Act_i of *local actions*, a *local protocol* function $P_i : L_i \rightarrow 2^{Act_i}$ and a *local evolution* function $t_i : L_i \times Act_1 \times \ldots \times Act_n \times Act_{Env} \rightarrow L_i$. An IS is defined by the set $G \subseteq \prod_{1 \leq i \leq n} L_i \times L_E$ of *global states*, the set $Act = Act_1 \times \ldots \times Act_n \times Act_{Env}$ of *joint actions*, the *joint protocol* $\overline{P} = (P_1, \ldots, P_n, P_{Env})$ and the *global evolution* function $\overline{t} = (t_1, \ldots, t_n, t_{Env})$. For a global state $g \in G$ and a joint action $a \in Act$, we use g_i and a_i to denote the local state and the local action of agent i, respectively. For more details on interpreted systems, we refer the reader to [12].

Equational Theories [10]. For ease of reference, consider the following equational theory aimed at checking whether one integer is smaller or equal to another.

An Equational Theory (Σ_1, E_1) *for Illustrative Purposes.*

Signature Σ_1: Sorts: $S = \{nat, bool\}$; Variables: $X_{nat} = \{x, y\}$; Function Symbols: $\Sigma_{\lambda,bool} = \{true, false\}$; $\Sigma_{[(nat,nat),bool]} = \{\leq\}$; $\Sigma_{[bool,bool]} = \{\neg\}$ $\Sigma_{[\lambda,nat]} = \{0\}$; $\Sigma_{[nat,nat]} = \{succ\}$ $\Sigma_{[\omega,s]} = \emptyset$, otherwise (i.e., for other $\omega \in S^*, s \in S$);
The set E_1 of Σ_1-Equations : $((\neg true) = false)$; $((\neg false) = true)$; $(\leq(0, x) = true)$; $(\leq(succ(x), 0) = false)$; $(\leq(succ(x), succ(y)) = \leq(x, y))$;

Let S be a non-empty set of *sorts* (i.e., simple types such as *nat* and *bool* in the example above). Let $\Sigma = \{\Sigma_{(\omega,s)} | \omega \in S^*, s \in S\}$ be an *S-sorted signature*, i.e., a collection of *functional symbols* of *type* $[\omega, s]$. Generally, $\sigma \in \Sigma_{(\omega,s)}$ denotes a *function symbol* (e.g., $\neg, \leq$, *succ* in the example above). Let X be an S-sorted set of variables (e.g., x and y above), where X_s are the variables of sort s (e.g., X_{nat} is still $\{x, y\}$ above). Let $T_{\Sigma,X}$ be the S-sorted set of terms over X and Σ (e.g., $succ(x)$ is a term over signature Σ_1 in the example above), and T_Σ be the set of *ground* terms, i.e., terms without variables. An *equational theory* is a tuple (Σ, E), where Σ is a signature and E is a set of Σ-equations. The notation $t = t'$ or $t \rightarrow_E t'$ denotes a Σ-*equation*, i.e., a pair (t, t') of terms equal under E.

The semantics for equational theories can be given through the *S-sorted* Σ-*algebra* $\mathbb{A} = (A, \Sigma^A)$ [10], where the set $A = (A_s \mid s \in S)$ is an indexed set of values (i.e., the sort *nat* above is mapped under $\mathbb{A}$ onto the set A_{nat} of concrete values, e.g., natural numbers). For each sort s, the set A_s is called the *support-set* for s. The set Σ^A is a set of functions f^A from A_ω to A_s corresponding to function symbols f in Σ, $f \in \Sigma_{(\omega,s)}$, $\omega \in S^*$, $s \in S$ (e.g., the symbol *succ* corresponds to a concrete successor function operating on natural numbers). The indexed set $\delta = (\delta_s \mid s \in S)$ of maps is an *assignment* of X into the algebra $\mathbb{A}$; the tuple $\delta = [x_1/_{v_1}, \ldots, x_n/_{v_n}]$ represents an assignment where the variable x_i is set to the value v_i, for $i \in \{1, \ldots, n\}$. Moreover, the notation $\delta[x/_v]$ represents the assignment obtained from δ when $\delta_s(x)$ is replaced by the value v, for some $x \in X_s$, $v \in A_s$, $s \in S$.

The relation $\rightarrow_E^*$ is the transitive closure of $\rightarrow_E \cup =$. Let $t \in T_{\Sigma,X}$. The *normal term* of t (denoted by $t{\downarrow_E}$) is the unique term $t' \in T_{\Sigma,Y}$ such that $t \rightarrow_E^* t'$, where $Y \subseteq X$. Finally, recall that an equational theory E is *convergent* if the algebra of its semantics can be mechanised into a rewriting system $\rightarrow_E$ which is convergent (i.e., *confluent* and *terminating* [2]). A theory is *subterm convergent* if all the subterm in the left-hand side of the rewriting rule also appear on the right-hand side of it. For more details on equational theories, we refer to [10].

Protocols. Security protocols often rely on primitives such as encryption and hashing to establish some security property, e.g., authentication. These primitives can be formally described by equational theories. Consider the simple protocol below constructed on the theory (Σ_1, E_1).

A Communication Protocol Pr_1.

$$1.\ A \rightarrow B : n$$
$$2.\ B \rightarrow A : m$$
$$3.\ A \rightarrow B :\ \leq(n, m)$$

The protocol Pr_1 describes a set of send-receive rules of the two *roles*: the *A-role* and the *B-role*. An agent assuming the *A-role* initiates the protocol by sending the term n to its *B*-role partner agent. This receiver replies with the term m. The initiator terminates the protocol with the acknowledgement $\leq(n, m)$, where $\leq$ is a publicly known function symbol. This symbol is described by the equational theory (Σ_1, E_1) aforementioned. The protocol Pr_1 is purposely simple to exemplify the material that follows.

High-level security languages such as Common Authentication Protocol Specification Language (CAPSL) [9] provide precise descriptions of security protocols, including the underlying equational theory formalising the effects of the protocol-primitives executed by each role (i.e., in our presentation, by some agent assuming that role). We assume that all protocols referred to henceforth are specified in CAPSL.

2.2 Protocol Model

In this subsection we put forward a technique for producing a fully instantiated interpreted system that models a finite number of protocol sessions running concurrently. The aim of this construction is to obtain interpreted systems in which the epistemic relation for each agent is an equivalence relation under the underlying equational theory of the protocol.

Consider a generic security protocol Pr that is specified by an equational theory (Σ, E). Its execution generates an instantiation of the protocol. To model this, let a Σ-algebra $\mathbb{A}$, together with a finite set Δ of assignments of variables X in $\mathbb{A}$, be the interpretation of the protocol's theory (Σ, E). Importantly, assume that the rewriting sequence $t \rightarrow_{E^*} t{\downarrow_E}$ mechanising the term algebra $\mathbb{T}_{\Sigma,X}$ is somehow provided for each $t \in T_{\Sigma,X}$, e.g., by using a rewriting engine or a CAPSL compiler [9] a priori. Since the normal terms are at hand, to obtain the equivalence between states we will not need to express the message-deducibility relation beforehand, as required in other approaches [8]. Also, we only consider a bounded number of protocol instantiations. By doing so

we obtain decidable state-equivalence modulo convergent equational theories.

On the algebra $\mathbb{A}$, let the set $T_{\Sigma,X}|_{Pr}$ denote the terms used only in the actual description of the protocol Pr. When Pr is implicit, we simply write simply $T_{\Sigma,X}$ to mean $T_{\Sigma,X}|_{Pr}$. In doing so, we underline the protocol-terms (i.e., messages and their subparts) and, later, their values. Thus, for the protocol Pr_1, the set $T_{\Sigma_1,X}|_{Pr_1}$ of terms is $\{A, B, n, m, \leq(n,m)\}$. To highlight variables describing a role of the protocol (i.e., variables A and B in Pr_1), we introduce an additional sort called *role*. Variables of sort *role* (i.e., X_{role}) range over support set A_{role}. This is taken to be a set of strings, for example $\{alice, bob, greg, \ldots\}$.

Protocol Roles to Agents.

An entire protocol-role (e.g., a sender) is described by a variable of sort *role* together with all the terms and actions that inherently characterise it.

Assume a particular assignment $\delta \in \Delta$. An assignment δ (homomorphically) instantiates all the terms in $T_{\Sigma,X}|_{Pr}$. As such, an assignment symbolically corresponds to a *protocol session* (e.g., a session of Pr_1 is given by $\delta = [n/_3, m/_2, A/_{alice}, B/_{bob}]$). However, to model the development of the protocol execution more clearly, we will use the projection of each such assignment $\delta \in \Delta$ per each role. Note that when the sender A starts the protocol Pr_1, it does not possess any value for the variable m or for the term $\leq(n,m)$ (as it will only receive m after the protocol starts, from some interlocutor of *B-role*). To formalise this, we will say that a variable is *free in* or *bound to a role* (see formalisation below).

Variables in a Role. A variable x is *bound* to a role R, written $x \in \mathcal{B}_R$, if the (CAPSL) protocol description stipulates the variable x as private to the role R. Otherwise, a variable y is *free* in a role R, written $y \in \mathcal{F}_R$. The extension to non-atomic terms is as usual: if $t \in T_{\Sigma,X'}$ and $X' \cap \mathcal{F}_R \neq \emptyset$ then $t \in \mathcal{F}_R$, otherwise $t \in \mathcal{B}_R$.

In the Pr_1 protocol the variable n is bound to the role of A, while the variable m is free in the role of A. Therefore, the term $\leq(m,n)$ is free in the role of A.

To denote the initial ignorance of some concrete values within a role, we will use designated values, called *null values*, in the assignment of the variables and terms which are free in a role. To denote these null values, we use $(\bot_s \mid s \in S)$, an S-sorted set of constant function symbols. When the sort s is implicit, we simply write $\bot$ instead of $\bot_s$. All constant function symbols over $\omega \in S^*$ in which at least one component is $\bot$ are denoted by $\bot_\omega$, i.e., if n has value $\bot_{nat}$ within $\leq(m,n)$, then the whole value of $\leq(m,n)$ is also $\bot$, specifically $\bot_{[(nat,nat),bool]}$. Bound variables are assigned to concrete, non-null values, chosen arbitrarily over the a given range, e.g., integers, etc. Let the universal algebra $\mathbb{A}$ be the denotational interpretation of the theory (Σ, E). To be able to define operations on null values, we naturally extend the denotation $\mathbb{A}$ to $\mathbb{A}_\bot$, which has $A_{s_\bot} = A_s \cup \{\bot_s\}$ as support-sets and it operates over A_s like $\mathbb{A}$ and it returns $\bot$ whenever it operates over $\bot$, for any $s \in S$.

Initial Instantiation of Roles. Let $\delta \in \Delta$ be an assignment. The *initial R-role instantiation* $\delta|_R$ is the projection of the assignment δ on a role R, extended to $\mathbb{A}_\bot$ to enforce the assignment of all terms in the *R-role* to values, including null values: $\delta|_R = (t/_{\delta(t)},\ t'/_{\bot_s} \mid t \in (\mathcal{B}_R)_s,\ t' \in (\mathcal{F}_R)_s,\ s \in S)$.

For the protocol Pr_1, let $\delta = [n/_3, m/_2, A/_{alice}, B_{bob}]$ be an assignment. Then, by the definition above, the initial role instantiations are as follows:
$\delta|_A = [n/_3, m/_\bot, A/_{alice}, B/_{bob}, \leq(n,m)/_\bot]$,
$\delta|_B = [n/_\bot, m/_2, A/_{alice}, B/_{bob}, \leq(n,m)/_\bot]$.

For each role R, we map each initial instantiation $\delta|_R$ of the *R-role* into an agent ag_R^δ. This gives the *set* $Ag = \bigcup_{\delta\in\Delta} \bigcup_{R\in X_{role}} \{ag_R^\delta\}$ *of agents.*

An IS Protocol Semantics.

In the following let the agent ag_R^δ represent an arbitrary *R-role* under the assignment δ of a Pr protocol. In particular, let ag_A^δ correspond to the *A-role* under $\delta|_A$, for an assignment $\delta = [n/_3, m/_2, A/_{alice}, B/_{bob}]$ of the protocol Pr_1. We now present the formal description of the agent ag_R^δ.

We consider several concurrent instantiations of each role by different agents. So, a free term ($\bot$) representing the role of the sender can later be instantiated to potentially different values, depending on the value received from other agents. A receipt may trigger the instantiations of other local terms as prescribed by the equational theory of the protocol. For instance, in Pr_1 with $\delta = [n/_3, m/_2, A/_{alice}, B/_{bob}]$ an *A-role* participant may receive the value 2 for m from a *B-role* agent. Following this, the *A-role* agent will "apply" the equational theory E_1 to rewrite the term $\leq(m,n)$ to $\leq(3,2)$, $\leq(2,1)$, $\leq(1,0)$ and, finally, to $.F.$[1]. To permit this, the term $\leq(m,n)$ of sort *Bool*, which is free in the role of A, should range over $(A_{nat} \times A_{nat}) \cup A_{bool_\bot} = (\mathbb{N} \times \mathbb{N}) \cup \{.T., .F.\}_\bot$. However, the term n should efficiently range only over $A_{nat}{=}\mathbb{N}$ for this agent, since n is bound to the *A-role* and its initial value cannot be changed. These value-range restrictions optimise the size of the fully instantiated model. The following definition formalises this by giving the possible values of a portion of a message held by an agent during the run.

Range of a Term for an Agent. The *range* $Range_R(t)$ *of a term* $t \in T_{\Sigma,X}|_{Pr}$ *for an* ag_R^δ *agent* is as follows:
$Range_R(t) =$

$$\begin{cases} A_s & t \in (\mathcal{B}_R)_s \cap X_s \quad (1) \\ A_{s_\bot} & \text{if } t \in (\mathcal{F}_R)_s \cap X_s \quad (2) \\ (A_{s_1} \times \ldots \times A_{s_n}) \cup A_s & \text{if } t \in (\mathcal{B}_R)_s,\ t = \sigma(t_1, \ldots, t_n), \\ & \sigma \in \Sigma_{(s_1,\ldots,s_n),s},\ t_i \in T_{\Sigma,X,s_i} \quad (3) \\ (A_{s_1} \times \ldots \times A_{s_n}) \cup A_{s_\bot} & \text{if } t \in (\mathcal{F}_R)_s,\ t = \sigma(t_1, \ldots, t_n), \\ & \sigma \in \Sigma_{(s_1,\ldots,s_n),s},\ t_i \in T_{\Sigma,X,s_i} \quad (4) \end{cases}$$

Stores and Views for an Agent. A *store for an agent* ag_R^δ is a relation $(t :: Range_R(t) \mid t \in T_{\Sigma,X}|_{Pr})$ between terms and their respective ranges for the agent ag_R^δ.

An *initial view for an agent* ag_R^δ encodes an a initial R-role instantiation δ. A *non-initial view for* ag_R^δ encodes an actual assignment $\delta[y/_v]$, for some $y \in (\mathcal{F}_R)_s$, $v \in A_s$ (i.e., $v \neq \bot_s$).

The store of ag_A^δ in a model for Pr_1 is as follows:
$store_{ag_A^\delta} = (A :: String, B :: String, n :: \mathbb{N}, m :: \mathbb{N}_\bot,$
$\leq(n,m) :: (\mathbb{N} \times \mathbb{N}) \cup \{.T., .F.\}_\bot)$.

A possible non-initial view for ag_A^δ in a model for Pr_1 is:
$view_{ag_A^\delta} = (A \mapsto alice, B \mapsto bob, n \mapsto 3, m \mapsto 2,$
$\leq(n,m) \mapsto \bot)$.

The non-initial view $view_{ag_A^\delta}$ shows that ag_A^δ has updated the value for m in its view from the initially held $\bot$ to the received value 2. In the above, ag_A^δ has not yet "calculated" the value of $\leq(n,m)$, i.e., $\leq(n,m)$ is still $\bot$ in $view_{ag_A^\delta}$.

[1] $.T.$ and $.F.$ are the concrete values for *true* and *false*.

To complete the description of instantiated protocol roles, we introduce the set of *adjacent terms* Adj. These terms are unrelated to the equational theory E, but are induced by the (CAPSL) protocol description (e.g., flag variables to denote protocol steps, stages of rewriting, etc.).

Local States of Agents. An *(initial) local state* is an (initial) view together with certain protocol-driven *adjacent* terms and their assigned values. The set $L_{ag_R^\delta}$ is the set of all *possible local states* of ag_R^δ.

Let $step \in Adj$ be an adjacent atomic term of sort nat (i.e., denoting protocol steps). Then, let $Range_R(step) = \mathbb{N}$ and $step \mapsto 1$ in an initial setup, for any role R. An initial state i_l and local state l of agent ag_A^δ in a model for Pr_1 are as follows:
$i\_l = (step \mapsto 1, A \mapsto alice, B \mapsto bob, n \mapsto 3, m \mapsto \bot_{\mathbb{N}},$
$\leq(n,m) \mapsto \bot_{\{.T.,.F.\}})$;
$l = (step \mapsto 5, A \mapsto alice, B \mapsto bob, n \mapsto 3, m \mapsto 2,$
$\leq(n,m) \mapsto .F.)$.

Let $l \in L_{ag_R^\delta}$, $t \in T_{\Sigma,X}$ and $x \in Range_R(t)$. In the following, we use the following notations:

$l.view$ denotes the view encoded inside the local state l
$l.t$ denotes that the local state l stores a value for the term t
$l\|_{t=x}$ denotes the fact that $l.t{=}x$
$l\|_\delta$ denotes that $l\|_{t=x}$ and $\delta{=}[t/x]$ for all t in the domain of δ
$l[t/x]$ denotes the fact that $l.t$ is set to x

In the following, let i denote the map ag_R^δ of an initiator role R and i' denote the map $ag_{R'}^{\delta'}$ of a receiver role R'.

Local Actions and Protocol of Agents. Let $step \in Adj$, $j \in \{1,2,3\}$, $nr_j \in Range_R(step)$ and t, x, i', l be as above. The set $LAct_i = \{send(t,x,i'), receive(t,x), rewrite, empty\}$ is the set of *possible local actions* of agent i. The *local protocol* P_i of agent i is as follows: $P_i(l|_{step=nr_1, l.t=x, l.R'=i'.R'}) = \{send(t,x,i')\}$, $P_i(l|_{step=nr_2}) = \{receive(t,x)\}$, $P_i(l|_{step=nr_3}) = \{rewrite\}$.

When a particular protocol is given, the parameters of the actions are restricted to proper subsets of $T_{\Sigma,X}$, e.g., t ranges over certain terms in $receive(t,x)$ for i.

The Environment Agent. We assume that the environment agent Env records all communication. Therefore, the local states of the Env agent are given by maps of the form $(t :: \bigcup_{R \in role} Range_R(t) :: Ag :: Ag \,|\, t \in T_{\Sigma,X})$. This gives the set L_{Env} of *possible local states of the Environment agent*. The environment has only one possible action denoted by $listen$, which is enabled by its protocol at every local state.

Global States and Joint Actions. Let $i \in Ag = \{1, \ldots, n\}$, $l_i \in LAct_i$, $l_{Env} \in L_{Env}$, $a_i \in Act_i$ and $a_{Env} \in Act_{Env}$. A *global state* g is a tuple $(l_1, \ldots, l_n, l_{Env})$. The set G of global states is the set of all possible states g as above. A *joint action* a is a tuple $(a_1, \ldots, a_n, a_{Env})$. The set Act of joint actions is the set of all possible joint actions a as above.

Agents' Local Evolution Function. Let i denote the ag_R^δ agent, i' denote the $ag_{R'}^{\delta'}$ agent as above, let $l \in L_i$ be a local state of agent i and $a \in Act$ be a joint action. The *local evolution function* E_i of agent i is defined below. In this definition, the preconditions for enabling a state-update upon receipt express the following: *1*) the action $receive$ of agent i is synchronised with the action $send$ of agent i' and with the action $listen$ of the Env agent; *2*) agent i is in the step nr where it awaits message t; *3*) the purported sender is the agent of the R'-$role$[2] (i.e., $i.R' = i'.R'$); *4*) the values x_j of certain subterms t_j in the received term t are consistent with agent i's view, i.e., $l|_{t_j=x_j}$. These conditions are inspired by the matching-receive semantics [3, 21].

$$E_i(l,a) = \begin{cases} l[step/_{nr+1}] & \text{if } l|_{t=x,step=nr,R'=i'.R'}, \text{ for } a_i = send(t,x,i'), a_{Env} = listen, a_i' = receive(t,x) \\ l[step/_{nr+1}, t/x] & \text{if } l|_{t_j=x_j,step=nr,R'=i'.R'}, \text{ for } a_i = receive(t,x), a_{Env} = listen, a_i' = send(t,x,i),\ t_j \in Sub(t) \\ l[step/_{nr+1}, t/t'] & \text{if } l|_{step=nr+1}, \text{ for } a_i = rewrite, a_{Env} = listen, a_i' = empty,\ t \in T_{\Sigma,X}, t' = t{\downarrow_E} \end{cases}$$

To illustrate further, let $i = ag_A^\delta$ in the Pr_1 protocol and, by E_i, let the action $receive(m,2)$ be performed at the local state $l|_{step=1}$ of agent i. The implicit rewriting-driven state-update is: $\leq(3,2) \rightarrow \leq(succ(2), succ(1)) \rightarrow \leq(2,1) \rightarrow \leq(succ(1), succ(0)) \rightarrow \leq(1,0) \rightarrow .F.$. For protocols where the intermediate rewriting is not of interest, we collapse such a state-update sequence in one update, i.e., the sequence $l[step/_3, \leq(n,m)/_{(2,1)}]$, $l[step/_4, \leq(n,m)/_{(1,0)}]$ and $l[step/_5, \leq(n,m)/_{.F.}]$ is reduced to $l[step/_3, \leq(n,m)/_{.F.}]$. The above presentation of the local evolution function E_i formalises such optimisations.

The Global Evolution Function. The *global evolution function* $t : G \times Act \rightarrow G$ is such that $t(g,a) = g'$ if $act_i \in P_i(g_i)$, $E_i(g_i, a) = g_i'$, for all $i \in Ag \cup \{Env\}$, for $g, g' \in G$ and $a \in Act$.

A *path* is a sequence of global states described by the global evolution function. Paths naturally define the set of *reachable* states. Henceforth, G refers to the set of reachable states.

Equational Interpreted System for Pr. An *equational interpreted system for Pr*, denoted by Υ_{IS}^E, is a tuple $\mathcal{I} = (G, Act, P, t, I_0, V)$, where the components Act and t are as previously defined, $I_0 \subset G$ is a set of initial global states, $P = (P_i \,|\, i \in Ag \cup \{Env\})$, and $V : G \times PV \rightarrow \{true, false\}$ is a valuation function for the propositions PV of a logic language.

Local Satisfaction of Equational Equalities of Terms. The local state $l \in L_i$ *satisfies* $t =_E t'$, written $l \models (t =_E t')$, if $l|_{t=t'}$, for $t \rightarrow_{E*} t'$ with $t' = t{\downarrow_E}$, $t \in T_{\Sigma,X}|_{Pr}$ (i.e., $t \notin Adj$).

By the definition above, a local state l satisfies the equality $t =_E t'$ of terms modulo E if the term t has been rewritten to the normal term t' in local state l of Υ_{IS}^E.

Equational Indistinguishability. Two local states $l \in L_i$ and $l' \in L_i$ are *i-indistinguishable modulo E*, written $l \approx_i^E l'$, if it is the case that $l \models (t =_E t')$ if and only if $l' \models (t =_E t')$, for all $t \in T_{\Sigma,X}|_{Pr}$, i.e., $t \notin Adj$. Two reachable global states $g, g' \in G$ are *i-indistinguishable modulo E*, written $g \sim_i^E g'$, if $g_i \approx_i^E g_i'$. The relation $\sim_i^E \subseteq G \times G$ is the *quotient-indistinguishability relation*.

By the definition above, two local states are indistinguishable modulo E if they satisfy the same equalities of terms modulo E.

[2] If protocols use anonymous channels, then this condition is dropped.

As an example, let i be ag_A^δ in the $\Upsilon_{IS}^{E_1}$ for the protocol Pr_1. As none of the following states of ag_A^δ satisfy $\leq(m,n) = {}_{E_1}.F.$, it holds that $l_1|_{step=2,\leq(n,m)=(3,2)} \approx_i^{E_1} l_2|_{step=3,\leq(n,m)=(2,1)} \approx_i^{E_1} l_3|_{step=4,\leq(n,m)=(1,0)}$. However, the state $l|_{step/_5,\leq(n,m)/_{.F.}}$ does satisfy $\leq(m,n) =_{E_1}.F.$

Equational Multi-agent System Model for Pr. Let $\mathcal{I}$ be an equational interpreted system for a protocol Pr specified by a convergent equational theory (Σ, E). The *equational multi-agent system model for Pr* $M_{IS}^E = (G, (\sim_i^E)_{i\in Ag}, V)$ is the model generated by the equational interpreted system model $\mathcal{I}$. In M_{IS}^E the relation $\sim_i^E$ is as described above, G is the set of reachable states generated by $\mathcal{I}$, and V is the set of atomic proposition in $\mathcal{I}$.

We use the notation $\mathcal{I}$ both for the equational interpreted system for Pr and the equational multi-agent system model for Pr; the context will disambiguate. In our implementation, we optimise the formalism above when generating the Υ_{IS}^E; this is not discussed here.

2.3 The Epistemic Logic CTLKR

Let $\mathcal{I}$ be the equational multi-agent system model M_{IS}^E of Pr, $p \in PV$ and $i \in Ag \cup \{Env\}$. The specification language CTLKR, used to express the system requirements is defined by the following BNF:

$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_i\varphi \mid R_i\varphi \mid AX\varphi \mid AG\varphi \mid A(\varphi U \varphi)$.

The operator K_i denotes the *knowledge modality* ($K_i\varphi$ reads "agent i knows the fact φ") while the operator R_i is the *rewriting-knowledge modality* ($R_i\varphi$ reads "agent i knows the fact φ modulo the equational theory (Σ, E)"). The semantics for CTL on M_{IS}^E is as on standard interpreted systems [12]. The interpretation of the knowledge modalities is as follows:

$(\mathcal{I}, g) \models K_i\varphi$ if (for all $g' \in G$)($g_i = g'_i$ implies $(\mathcal{I}, g') \models \varphi$)
$(\mathcal{I}, g) \models R_i\varphi$ if (for all $g' \in G$)($g \sim_i^E g'$ implies $(\mathcal{I}, g') \models \varphi$).

The logic CTLKR extends the commonly used logic CTLK by means of the rewriting epistemic modality R.

3. AN APPROXIMATION FOR AUTOMATIC VERIFICATION

We wish to use the logic CTLKR as a specification language for model checking security protocols encoded in the MAS-based formalism presented in the previous section. This would enable us to surpass the significant limitations of the state-of-the-art as discussed in the introduction. However, locally parametrised properties of type $t =_E t'$ make the computation of the indistinguishability relation particularly costly, thereby increasing the verification time. To circumvent this, we approximate the R modality and interpret it over an abstraction of Υ_{IS}^E through the use of local predicates. In the following we will show that important classes of protocols are amenable to analysis through this approximation.

3.1 Empirical Interpreted Systems

An *S-sorted logical signature* contains *logic symbols* of type $[\omega]$, for $\omega \in S^*$. Informally, a (standard) signature specifies symbols related to algebraic operators, e.g., *decrypt*, whereas a logical signature specifies symbols related to facts, e.g., *isDecrypted*.

Logically Extended Signatures. A *logically extended signature* is given by a tuple (Σ, Σ_L), where Σ is an S-sorted signature and Σ_L is an S-sorted logical signature.

The tuple (Σ, Σ_L, E) is the *logically extended equational theory* corresponding to the equational theory (Σ, E). A logically extended equational theory can describe more properties of a protocol than the underlying equational theory alone.

The set $T_{\Sigma,\Sigma_L,X}$ of *logical terms* is defined on logically extended signatures (Σ, Σ_L) in the same way the set $T_{\Sigma,X}$ of terms is defined on the signature Σ.

The denotation of logically extended signatures is given through a *logical extension of the algebra* $\mathbb{A}_\perp$. In this extension the *interpretation* $i\_p^{\mathbb{A}_\perp}(\delta)$ *of a logical term* $p \in T_{\Sigma,\Sigma_L,X}$ *under assignments* $\delta \in \Delta$ is a predicate $p^{\mathbb{A}_\perp}$ evaluated over $\{true, false\}$. When $\mathbb{A}_\perp$ is implicit, we simply write $i\_p(\delta)$ instead of $i\_p^{\mathbb{A}_\perp}(\delta)$.

Let j be an arbitrary agent in an IS formalisation.

Logical Terms and Experiments of Agents. A fixed set $In_j \subseteq T_{\Sigma,\Sigma_L,X}$ denotes *the set of logical terms of agent* j. The set $InEx_j = \{i\_p(\delta) \mid p \in In_j\}$ of predicates contains the *local experiments for agent* j. An $InEx_j$ set is denoted as a *local experiment-set*. The Ag-indexed set $InEx = (InEx_j \mid j \in Ag)$ is the *experiment-set*.

Logical terms are symbols that enrich the agents' stores with "meta-data" representing facts not explicitly included in the protocol. Experiments are predicates that are evaluated on the views or the local states of agents, i.e., interpreting this meta-data. Thus, evaluating these predicates will account for a special kind of knowledge accrued by the agents.

Below we illustrate these notions, with intuitive predicates and symbols: e.g., i_diffOne$(n,m)(\delta)$ is *true* if "the absolute difference between $\delta(n)$ and $\delta(m)$ is 1".

Possible Experiments For Agent ag_A^δ *in* Pr_1. The sets In of logical terms and their respective experiment-sets $InEx$ are as follows:

- $In1_i$ = {smaller(n,m)}; $InEx1_i$=i_smaller$(n,m)(\delta)$;
- $In2_i$ = {diffOne(n,m)}; $InEx2_i$=i_diffOne$(n,m)(\delta)$.

Similarly to [17], we introduce an indistinguishability relation defined over local predicates, i.e., here on local experiments of agents.

Local Empirical Indistinguishability. Two local states $l, l' \in L_j$ are *indistinguishable modulo* $InEx_j$, or $l \approx^{InEx_j} l'$, if $i\_p(\delta) = i\_p(\delta')$ for all $p \in In_j$, where $\delta, \delta' \in \Delta$ respectively describe l and l', i.e., $l|_\delta$ and $l'|_{\delta'}$. Two global states g, $g' \in G$ are *indistinguishable modulo* $InEx_j$, written $g \sim^{InEx_j} g'$, if $g_j \approx^{InEx_j} g'_j$. The relation $\sim^{InEx_j} \subseteq G \times G$ is the *empirical indistinguishability* relation.

Then, two local states l and l' are indistinguishable through experiments if these are evaluated identically at l and at l', as exemplified below.

Empirical Indistinguishability in a Model for Pr_1. Let i be the agent ag_A^δ in a model for Pr_1 and two local states of ag_A^δ respectively described by $\delta[n/_9, m/_8]$ and $\delta[n/_9, m/_5]$, i.e., $l_{ag_A}|_{\delta[n/_9,m/_8]}$ and $l'_{ag_A}|_{\delta[n/_9,m/_5]}$. Let $InEx1_i$ and $InEx2_i$ be the experiment-sets above. Then,

- i_smaller$(n,m)(\delta[n/_9, m/_8])$=$false$ and i_smaller$(n,m)(\delta[n/_9, m/_5])$=$false$;
- i_diffOne$(n,m)(\delta[n/_9, m/_8])$=$true$ and i_diffOne$(n,m)(\delta[n/_9, m/_5])$=$false$.

Therefore, $l \approx^{InEx1_i} l'$ holds, but $l \approx^{InEx2_i} l'$ does not hold, i.e., $l \not\approx^{InEx2_i} l'$.

Therefore, we have just augmented the Υ_{IS}^E formalisation of protocol executions with local experiments. The resulting models are formally defined below.

Empirical Equational Interpreted System. Let Pr be a protocol specified by (Σ, Σ_L, E) and $\mathcal{I}$ be the equational interpreted system Υ_{IS}^{E} for Pr. An *empirical equational interpreted system* is the tuple $\mathcal{I}'=(\mathcal{I}, InEx)$, where $InEx = (InEx_j \,|\, j \in Ag \cup \{Env\})$.

3.2 Extended Protocol Logic

We use P_i as the *empirical knowledge modality* ($P_i\varphi$ reads "agent i empirically knows the fact φ"). On an empirical equational multi-agent system model $\mathcal{I}' = (\mathcal{I}, InEx)$, consider the language $\mathcal{L}' = CTLKR \cup P_i$, for $i \in Ag \cup Env$.

The language CTLKR is interpreted on $\mathcal{I}'$ as on $\mathcal{I}$. Let $g \in G$ be an arbitrary reachable state of $\mathcal{I}'$. The interpretation of the empirical knowledge modality is as follows:

$(\mathcal{I}', g) \models P_i\varphi$ if
(for all $g' \in G$)($g \sim^{InEx_i} g'$ implies $(\mathcal{I}', g') \models \varphi$).

The empirical knowledge of an agent refers to the information obtained only by theoretically enquiring the agent's local predicates, i.e., its experiments.

3.3 Experiments Sets of Convergent Equational Theories

In this section we explicitly link the convergence of the underlying equational theory to the experiments of the agents. Once again, we assume that the normal terms of the theory are encoded in the model, i.e., by using a rewriting system, and that the number of protocol instantiations considered is bounded.

Let Pr be a protocol specified by a *convergent* equational theory (Σ, E), $\mathcal{I}$ be the equational interpreted system for Pr and j denote the ag_R^δ agent as before. Let Σ_L be a logical signature containing the (special) logical symbols $pred \in \Sigma_L$ of type ω, for all $\omega \in S^*$. Let t be an arbitrary term of type ω, i.e., $t \in T_{\Sigma,X}$.

Predicates for Terms. A logical term $pred(t) \in T_{\Sigma,\Sigma_L,X}$ is a *logical term for* $t \in T_{\Sigma,X}$. The *interpretation* $i\_pred^E(t)(\delta)$ *of a predicate for* t *in* E is always $true$, i.e., $i\_pred^E(t)(\delta) = true$, for all $\delta \in \Delta$.

By the definition above, a predicate $i\_pred^E(t)$ for a term $t \in T_{\Sigma,X}$ is $true$ under all assignments $\delta \in \Delta$ for t. Since δ is in Δ, i.e., δ is not an role instantiation, it means that $\delta(t) \neq \bot$. In the next definition we use predicates for terms to express special experiments, which simulate the recording of the normal terms.

Local Experiments of Convergent Theories.
$In_j^E = \bigcup_{t \in T_{\Sigma,X}} \{pred(t') \,|\, t' = t\downarrow_E\}$ is the set of logical terms *for the convergent theory* E of agent j. $InEx_j^E = \{i\_pred^E(t)(\delta) \,|\, pr(t) \in In_j^E\}$ is set of local experiments *of the convergent theory* E for agent j.

Importantly, one can automatically produce the exact set of experiments of a convergent theory E for a protocol Pr by using the CAPSL description of the protocol, the finite set of instantiations given and the normal terms implied by E.

Empirical Equational IS for Convergent Theories.
Let $InEx_j^E$ be the local experiments *for the convergent theory* E of agent j and $InEx^E = (InEx_j^E \,|\, j \in Ag)$. An empirical equational interpreted system $\Upsilon_{IS}^{\mathbb{I}E}$ *for the convergent theory* E is given by the tuple $\mathcal{I}'=(\mathcal{I}, InEx^E)$.

The unwinding of $\Upsilon_{IS}^{\mathbb{I}E}$ follows as in previous definitions. By the above, the system $\Upsilon_{IS}^{\mathbb{I}E}$ is a special empirical system, i.e., agents "track" normal terms under E. We now prove that in these systems the P_i modality coincides with R_i.

THEOREM 3.1. *Let Pr be a protocol specified by a convergent theory (Σ, E) and $\mathcal{I}$ be an $M_{IS}^{\mathbb{I}E}$ model for E. Then, $\mathcal{I} \models \varphi$ if and only if $\mathcal{I} \models \overline{\varphi}$, for any $\varphi \in \mathcal{L}$, where $\overline{\varphi} \in \mathcal{L}'$ is obtained from φ by uniformly substituting R_j for P_j, for any $j \in Ag$.*

Proof (sketch). We only need to prove that $\mathcal{I} \models R_j\psi$ iff $\mathcal{I} \models P_j\overline{\psi}$, for some arbitrary $\psi \in \mathcal{L}$ and an agent $j = ag_R^\alpha$ under an initial R-role instantiation α.

Thus, $\mathcal{I} \models R_j\psi \overset{\text{def. of } \models_{\mathcal{L}}}{\Leftrightarrow}$ (for all $g' \in G$)($g \sim_j^E g'$ implies $(I, g') \models \psi$) $\overset{\text{def. of } \approx^E, \models_{\text{eq}}}{\Leftrightarrow}$ (for all $g' \in G$) (for all $t \in T_{\Sigma,X}, t' = t\downarrow_E$) (($g_j|_{t=t'}$ iff $g'_j|_{t=t'}$) implies $(\mathcal{I}, g') \models \psi$) (**1**). Let δ, δ' be assignments extending the initial R-role instantiation α and w.l.o.g. denote the local states in (1) as $g_j|_\delta$, $g'_j|_{\delta'}$ (**2**). Then, $In_j^E = \bigcup_{t \in T_{\Sigma,X}} \{pred(t\downarrow_E)\}$, $InEx_j^E = \cup_{t \in T_{\Sigma,X}} \{i\_pred(t\downarrow_E)(\delta) = true\}$ (**3**). By the definition of $\approx^{InEx_j^E}$, (2) and (3), the following holds in (1): $g_j \approx^{InEx_j^E} g'_j$ (**4**). From (1) with the above and (4), it follows that: $\overset{\text{def. of } \models_{\mathcal{L}'}}{\Leftrightarrow} (\mathcal{I}, g) \models P_j\psi$. □

4. MODEL CHECKING KNOWLEDGE OF PROTOCOL PARTICIPANTS

In this section we present a procedure for model checking empirical knowledge that allows for the specification and verification of standard interpreted systems equipped with local experiment-sets, i.e., not only for the equationally-driven $\Upsilon_{IS}^{\mathbb{I}E}$.

Algorithm 1 SAT$_P$(φ : FORMULA, j : AGENT) : *Set of* STATES

```
1: X ← [[¬φ]]
2: Y ← X
3: while X ≠ ∅ do
4:    g ← X.pop()
5:    φ_g ← true
6:    for exp ∈ InEx_j do
7:       if g ∈ [[exp]] then
8:          φ_g ← φ_g ∧ exp
9:       else
10:         φ_g ← φ_g ∧ ¬exp
11:      end if
12:   end for
13:   Y ← Y ∪ [[φ_g]]
14:   X.remove([[φ_g]])
15: end while
16: return ¬Y
```

The approach for calculating the set $[\![P_j\varphi]\!]$, i.e., the set of states that satisfy the formula $P_j\varphi$, is shown in Algorithm 1. Lines 8 and 10 construct the formula ϕ_g representing the conjunction of the evaluation of experiments for the agent j at the current state g. The set Y is constructed iteratively from each $g \in [\![\neg\varphi]\!]$ (the set X). At Line 13, $[\![\phi_g]\!]$ contains the set of states that are empirically indistinguishable from the state g (i.e., $[\![\phi_g]\!] = \{g' \in G \,|\, g' \sim^{InEx_j} g\}$). To calculate $Y = \{g \in G \,|\, (\exists g' \in G)(g' \sim^{InEx_j} g) \wedge (g \models \neg\varphi)\}$ efficiently, we remove $[\![\phi_g]\!]$ from X (Line 14) as these states have an identical experiment-set evaluation. At Line 15, Y contains

Table 1: E-Voting Specifications in CTLKR

VP	$AG(votes(i,v) \rightarrow AG \bigwedge_{v' \neq v} Q_{at}(votes(i,v')))$
VVU	$AG(votes(i,v) \rightarrow \bigwedge_{i' \neq i} \bigwedge_{v' \neq v} [votes(i',v') \rightarrow AGQ_{at}(votes(i,v') \wedge votes(i',v))])$
RF	for $v \in Range_V(vote) \setminus \{v_r\}$, $AG(votes(i_r,v_r) \wedge votes(i,v) \rightarrow \bigwedge_{i' \notin \{i,i_r\}} \bigwedge_{v'} [votes(i',v') \rightarrow AGQ_{at}(votes(i,v_r) \wedge votes(i',v) \wedge votes(i_r,v'))])$
CR	for $v \in Range_V(vote) \setminus \{v_c\}$, $AG(votes(i_c,v_c) \wedge votes(i,v) \rightarrow \bigwedge_{i' \notin \{i,i_c\}} \bigwedge_{v'} [votes(i',v') \rightarrow AGQ_{at}(votes(i,v_c) \wedge votes(i',v) \wedge votes(i_c,v'))])$

the reachable states that either directly refute φ, or are empirically indistinguishable from a state that does. Therefore we obtain that $Y = [\![\overline{P_j}(\neg\varphi)]\!]$, where $\overline{P_j}(\neg\varphi) \equiv \neg P_j(\neg\varphi)$ is the dual of $P_j(\varphi)$. Finally, at Line 16, the algorithm calculates $\neg Y$, i.e., the set difference between the set of global states G and Y. So, it returns $[\![P_j(\varphi)]\!]$.

PROPOSITION 4.1. *Algorithm 1 calculates the set of states* $[\![P_j\varphi]\!]$.

Implementation. We have implemented Algorithm 1 as an experimental extension of the model checker MCMAS [15]. This extension, titled MCMAS-E, is available from [1]. ISPL, The input-language of MCMAS, was extended to allow for the definition of experiments at the agent level, as well as to support the specification of empirical knowledge formulae.

5. VERIFYING E-VOTING PROTOCOLS

The applicability of previous research [3] in this line has been limited to protocols for which the specifications can be expressed by using standard notions of knowledge; this included authentication and key-establishment. We herein analyse e-voting protocols, which were out of the scope of [3].

To illustrate that the models and knowledge modalities introduced so far surpass this limit, we analyse more sophisticated e-voting protocols than previously possible with our extension of MCMAS.

E-Voting in the $\Upsilon_{IS}^{\mathbb{I}E}$ Formalism. Assume a $\Upsilon_{IS}^{\mathbb{I}E}$ model and the propositions $votes(j)$ and $votes(j,x)$, representing that an honest agent j has voted and that agent j has voted x, respectively. Let i, i' be two different agents. We consider only fair paths representing voting sessions in which eventually both agent i and agent i' vote and that the voting is not unanimous.

The specifications for e-voting requirements we consider, i.e., *vote privacy* (VP), *voter-vote unlinkability* (VVU), *receipt-freeness* (RF) and *coercion-resistance* (CR), are formalised in Table 1. We use the notation $Q_j\varphi$ to represent $\neg R_j \neg\varphi$, for any agent j.

VP stipulates that whenever agent i has voted v, there does not exist a point where the attacker *at* can be sure that it was i who voted v. Similarly, *VVU* expresses that the attacker *at* will always consider it possible that agents i and i' have swapped votes. *RF* states that, whenever agent i counterbalances the vote of the receipt-providing agent i_r, the attacker *at* is not at any point able to link any of the voters to their respective votes. *CR* is similar to RF, but it is analysed on a stronger threat-model. The formulae VVU, RF and CR are inspired by the specifications of *total role-interchangeability* [23], whereas VP is inspired by the specifications of anonymity in [14] and their extensions to privacy in [23].

We verify these specifications against the FOO'92 e-voting protocol [13]. We formalise the execution of a finite number of concurrent sessions as three, specialised $\Upsilon_{IS}^{\mathbb{I}E}$ systems. The first model, $\mathcal{M}_1$, is a $\Upsilon_{IS}^{\mathbb{I}E}$ model with an added *Attacker* agent (*at*) representing a passive intruder. This model satisfies the vote-privacy property. A receipt-providing agent i_r and a stronger *Attacker* are modelled in $\mathcal{M}_2$, which specialises $\mathcal{M}_1$ and supports receipt-freeness. To model coercion, the formalisation $\mathcal{M}_3$ extends $\mathcal{M}_2$, with a further enhanced *Attacker* and a coercible agent i_c.

Experiments. The high-level description of the FOO'92 protocol was initially provided in CAPSL [9]. This encoding was then passed to an ISPL translator [1]. The translator is an extension of the PD2IS toolkit [3] where the instantiated, $\perp$-enhanced normal terms are inserted into the ISPL models. In this way we can automatically generate the $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_3$ formalisations of FOO'92, as well as the e-voting requirements in ISPL. The generated ISPL files are in the region of 8000 lines and take approximately 15 seconds to build. MCMAS-E was then used to verify these models. The machine employed was an Intel Core 2 Duo processor 3.00 GHz with a 6144 KiB cache running the 32-bit Linux kernel 2.6.32.10. The averaged results obtained across two runs of MCMAS-E are summarised in Table 2.

Table 2:Averaged Experiments on FOO'92.

	Form.	Mem. (KiB)	Time (s)	States
$\mathcal{M}_1$	VP/VVU	176032	66441	$6.69 \cdot 10^{11}$
$\mathcal{M}_2$	(weakened) RF	175496	66168	$6.69 \cdot 10^{11}$
$\mathcal{M}_3$	VP/VVU	181926	70401	$6.69 \cdot 10^{11}$

The leftmost column shows the class of model considered. The *Memory* and *Time* columns respectively show the average memory usage and the average CPU time in each run. *States* reports the number of reachable states in each model.

Discussion of Results. The *Formulae* column reports the strongest e-voting specification that was found to hold on the model (strength grows from VP, VVU, RF to CR); vote privacy (VP and VVU) were found to hold on all three classes of models considered. On models $\mathcal{M}_2$ and $\mathcal{M}_3$ a path was found where eventually the intruder is able to link the receipt-providing agent and its vote, i.e., receipt-freeness (RF) was refuted. Our findings are in-line with known results (i.e., vote-privacy holding for FOO'92). An alternative approach based on applied-pi [8] exhibits similar results. The models verified are of large sizes and are not optimised for e-voting, consequently our verification times are seen as favourable. The complete set of ISPL models and specifications verified are available from [1].

6. CONCLUSIONS AND RELATED WORK

In this paper we have introduced an approach to model checking MAS-based models of security protocols, using spec-

ifications expressed in a specialised temporal-epistemic logic. This work surpasses the current state-of-the-art of temporal-epistemic verification of protocols specified as MAS in two ways. Firstly, it advances a formalism integrating equational theories with epistemic logic. This allows for the modelling of several cryptographic primitives of interest. Secondly, we present an automatic methodology for the verification multi-agent systems-based models against relevant specifications through an open-source dedicated model checker. We emphasise nonetheless that the methodology presented is not directly optimised for e-voting primitives; in fact, it aims at a generic MAS-based verification method.

The empirical indistinguishability relation introduced is related to that of explicit knowledge [17], although in that line no support for cryptographic primitives was available. In [20] agents are empowered with deduction algorithms for generating new local knowledge, but the technical details are different and no automatic technique is discussed. In a theoretical setting of cryptographic modelling, [22] studied the decidability of model checking with respect to an epistemic extension of ATL*; given the specification language, it is clear that the protocol model, the operators and the semantics in [22] differ from those we present.

Semi-decidable tools have been used to show static equivalence of applied-pi frames modulo certain convergent equational theories [6, 8]. Such approaches could be applied to verify symbolically an infinite number of e-voting sessions. However, they focus mainly on the problem of deciding static equivalence in process calculi (thus a comparison on protocol verification cannot be drawn). Comparatively, we assume a bounded number of fully instantiated protocol sessions where the normal terms of the theory are encoded in the model. Thus, we attain a decidable and fully automatic method of MAS-based protocol verification. In the context of bounded size modelling, the epistemic modalities and indistinguishability relations we have introduced can be correlated to process equivalence [6, 8] and, respectively, static frame-indistinguishability in applied-pi calculus.

Our specifications of e-voting requirements follow the formulations of anonymity in [14, 23] and are model-independent (unlike those in [8], where e-voting specifications are expressed as reachability or process equivalence properties in a model-dependent manner).

Acknowledgments.

The authors acknowledge the support of EPSRC grants EP/E035655/1 and EP/I00520X/1, and the Swiss-funded MICS project.

7. REFERENCES

[1] MCMAS-E, Model Checking Equational IS. `http://vas.doc.ic.ac.uk/tools/mcmas_equational/`.

[2] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge Univ. Press, 1998.

[3] I. Boureanu, M. Cohen, and A. Lomuscio. Model Checking Temporal Epistemic Specifications for Authentication Protocols Compiled into Interpreted Systems. *Journal of Applied Non-Clasical Logics*, 19(4):463–487, 2009.

[4] I. Boureanu, A. Lomuscio, and M. Cohen. Model Checking Detectability of Attacks in Multiagent Systems. In *Proc. of AAMAS'10*, pages 691–698. IFAAMAS Press, 2010.

[5] M. Burrows, M. Abadi, and R. Needham. A Logic of Authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990.

[6] Ş. Ciobâcă, S. Delaune, and S. Kremer. Computing Knowledge in Security Protocols under Convergent Equational Theories. In *Proc. of CADE'09*, pages 355–370. Springer, 2009.

[7] M. Cohen and M. Dam. A Complete Axiomatization of Knowledge and Cryptography. In *Proc. of LICS'07*, pages 77–88. IEEE Comp. Soc. Press, 2007.

[8] S. Delaune, S. Kremer, and M. Ryan. Verifying Privacy-Type Properties of Electronic Voting Protocols. *Journal of Computer Security*, 17(4):435–487, 2009.

[9] G. Denker and J. Millen. CAPSL Intermediate Language. In *Proc. of FMSP'99*. ACM New York, 1999.

[10] H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 1: Equations and Initial Semantics*. Springer, 1985.

[11] J. Ezekiel, A. Lomuscio, L. Molnar, S. Veres, and M. Pebody. Verifying Fault Tolerance and Self-Diagnosability of an Autonomous Underwater Vehicle. In *Proc. of IJCAI'11*, 2011.

[12] R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning about Knowledge*. MIT Press, 1995.

[13] A. Fujioka, T. Okamoto, and K. Ohta. A Practical Secret Voting Scheme for Large Scale Elections. In *Proc. of AUSCRYPT'92*, pages 244–251. Springer, 1992.

[14] J. Halpern and K. O'Neill. Anonymity and Information Hiding in Multiagent Systems. *Journal Computer Security*, 13(3):483–514, 2005.

[15] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS: A Model Checker for Multi-Agent Systems. In *Proc. of CAV'09*, volume 5643, pages 682–688. Springer, 2009.

[16] A. Lomuscio, F. Raimondi, and B. Wozna. Verification of the TESLA Protocol in MCMAS-X. *Fundamenta Informaticae*, 79(3-4):473–486, 2007.

[17] A. Lomuscio and B. Woźna. A Combination of Explicit and Deductive Knowledge with Branching Time: Completeness and Decidability Results. In *Proc. of DALT'05*, volume 3904, pages 188–204. Springer, 2005.

[18] R. van der Meyden and P. Gammie. MCK: Model Checking the Logic of Knowledge. In *Proc. of CAV'04*, volume 3114 of *LNCS*, pages 479–483. Springer, 2004.

[19] R. Parikh and R. Ramanujam. Distributed Processes and the Logic of Knowledge. In *Logic of Programs*, pages 256–268, 1985.

[20] S. Petride and R. Pucella. Perfect Cryptography, S5 Knowledge, and Algorithmic Knowledge. In *Proc. of TARK'07*, pages 239–247. ACM Press, 2007.

[21] M. Rusinowitch and M. Turuani. Protocol Insecurity with Finite Number of Sessions is NP-complete. *Proc. of IEEE-S&P'01*, 2001.

[22] H. Schnorr. Deciding Epistemic and Strategic Properties of Cryptographic Protocols. Technical Report TR 1012, Christian-Albrechts-Universitat zu Kiel, Institut fur Informatik, October 2010.

[23] Y. Tsukada, K. Mano, H. Sakurada, and Y. Kawabe. Anonymity, Privacy, Onymity, and Identity: A Modal Logic Approach. In *Proc. of CSE'09*. IEEE.