Bisimulations for Verifying Strategic Abilities with an Application to ThreeBallot

Francesco Belardinelli
IBISC, Université d'Evry & IRIT
Toulouse, France
belardinelli@ibisc.fr

Rodica Condurache
LACL, Université Paris-Est Créteil,
France
rodica.bozianu@gmail.com

Cătălin Dima
LACL, Université Paris-Est Créteil,
France
dima@u-pec.fr

Wojciech Jamroga
Institute of Computer Science,
Polish Academy of Sciences
w.jamroga@ipipan.waw.pl

Andrew V. Jones
Vector Software, Inc., London,
United Kingdom
andrew.jones@vectorcast.com

ABSTRACT

We propose a notion of alternating bisimulation for strategic abilities under imperfect information. The bisimulation preserves formulas of ATL for both the *objective* and *subjective* variants of the state-based semantics with imperfect information, which are commonly used in the modeling and verification of multi-agent systems. Furthermore, we apply the theoretical result to the verification of coercion-resistance in the three-ballot voting system, a voting protocol that does not use cryptography. In particular, we show that natural simplifications of an initial model of the protocol are in fact bisimulations of the original model, and therefore satisfy the same ATL properties, including coercion-resistance. These simplifications allow the model-checking tool MCMAS to terminate on models with a larger number of voters and candidates, compared with the initial model.

1. INTRODUCTION

The realm of formal languages for expressing strategic abilities of rational agents has witnessed a steady growth in recent years [8, 9, 23]. Among the most significant contributions we mention alternating-time temporal logic [3], strategy logic [13, 32], coalition logic [37]. These languages include modal operators, indexed to coalitions $A \subseteq Ag$ of agents, to express that the agents in A have a strategy to enforce a certain outcome, regardless of the behavior of the agents in $Ag \setminus A$. These syntactical features allow us to express winning conditions in multi-player games, notions of equilibrium (e.g. Nash), strategy-proofness [13, 33].

However, if these logics for strategies are to be applied to the specification and verification of multi-agent systems [21, 27, 30], they need to be coupled with efficient model checking techniques. Unfortunately, while in contexts of perfect information we benefit from tractable algorithms for model checking [3], the situation is rather different once we consider imperfect information. In contexts of imperfect information the complexity of the verification task ranges between Δ_2^P-completeness to undecidability, depending on whether we allow for perfect recall [20, 25]. In this setting it is crucial to develop complementary model checking techniques, in order to make the problem amenable.

In this line of research abstractions have proved to be a valuable tool for efficient verification [14, 15]. In this approach the concrete system S to be verified is abstracted into a "simpler" model S^A, which typically contains "less" transitions and therefore is "easier" to check in principle. Then, the verification result is transferred from the abstract S^A to the concrete S by virtue of some preservation result. Normally, preservation is guaranteed by proving that the abstract S^A is *(bi)similar* to S. (Bi)simulations are a powerful tool to analyze the expressiveness of modal languages, starting with van Benthem's result on modal logic as the bisimulation-invariant fragment of first-order logic [6]. However, (bi)simulations are a lot less understood in logics for strategies, where they have been studied mostly for contexts of perfect information [4, 22, 2].

In this paper we advance the state-of-the-art by introducing (bi)simulations for alternating-time temporal logic (ATL) under imperfect information. We prove that these (bi)simulations preserve the interpretation of formulas in ATL, when interpreted with imperfect information and imperfect recall, for both the objective and subjective semantics [8, 9]. Most interestingly for MAS verification, we apply these (bi)simulations to the abstraction of a class of electronic voting protocols without encryption.

Electronic voting has increasingly been considered as a robust alternative to paper-based voting due to a number of advantages it offers: accessibility, availability, voter turnout, less expensive and easier to use than paper voting, faster and more accurate ballot counting and results. However, electronic voting poses a number of challenges, some of which are common also to paper voting, but in a more technological setting: resistance and resilience to coercion and other types of fraud, secrecy, anonymity, verifiability, democracy (the right to vote at most once), accountability. Other issues are specific to electronic voting: access to internet, privatization, as well as public understanding and trust [40].

An increasing amount of research has focused recently on the verification of many of these properties for various types of voting protocols [5, 16]. The frameworks used for modeling and verifying security properties of voting protocols include, to mention only a few, process calculi such as the *applied π-calculus* or *CSP* [18, 24, 42], rewriting-based ap-

Appears in: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

proaches [11, 19, 7], approaches based on flat transition systems etc.

Here we develop a verification procedure for voting protocols that is based on a multi-agent logics approach. The main advantage of an approach based on multi-agent logics is the provision of a unified specification language for a variety of properties. A simple example is the variety of english statements of (non-probabilistic) coercion resistance that is around in the literature, which are usually implemented as behavioral equivalence properties involving some process algebraic model of the system [16]. However such approaches do not make it clear what is the system model and what is the property to be verified on the system. Multi-agent logics allow a clear separation of these two, as well as a wider variety of properties, involving the existence of attacker strategies. Our results, while only preliminary and adressing a simplified version of the Three Ballot protocol [39], allow the verification of systems with an increasing number of voters and candidates when compared with the approach based on process calculi from [34, 35].

Scheme of the Paper. In Section 2 we introduce the syntax and semantics of ATL interpreted under imperfect information and imperfect recall. In Section 3 we define (bi)simulation relations in this setting and prove that they preserve the interpretation of formulas in ATL. Then, in Section 4 we present the three-ballot voting protocol and formalize it as a game structure. In particular, we provide two abstractions of the three-ballot voting protocol and show that all systems are indeed bisimilar. Finally, in Section 5 we evaluate the gains in verification time and resources of model checking these abstractions in comparison to the original model. We conclude in Section 6 by discussing related works and by pointing to future directions of research.

2. THE FORMAL SETTING

In this section we introduce the syntax of ATL and its semantics defined on concurrent games structures with imperfect information. The following definitions and notation are taken from [20]. Concurrent game structures have been introduced in [3] in a perfect information setting. Here we consider their version for contexts of imperfect information [26].

DEFINITION 1. *A* concurrent game structure with imperfect information*, or iCGS, is a tuple* $\mathcal{G} = \langle Ag, AP, S, s_0, \{\sim_i\}_{i\in Ag}, Act, d, \rightarrow, \pi\rangle$ *such that*

- Ag *is a nonempty and finite set of* agents*. Subsets* $A \subseteq Ag$ *of agents are called* groups*.*
- S *is a non-empty set of* states *and* $s_0 \in S$ *is the* initial state *of* $\mathcal{G}$*.*
- *For each agent* $i \in Ag$*,* $\sim_i$ *is an equivalence relation on* S*, called the* indistinguishability relation *for* i*.*
- Act *is a finite non-empty set of* actions*. A tuple* $\vec{a} = (a_i)_{i\in Ag} \in Act^{Ag}$ *is called a* joint action*.*
- $d : Ag \times S \to (2^{Act} \setminus \{\varnothing\})$ *is the* protocol function*. For every* $i \in Ag$*,* $d(i)$ *returns the set of actions available to agent* i *at each state. Protocol* d *satisfies the property that, for all states* $s, s' \in S$ *and any agent* i*,* $s \sim_i s'$ *implies* $d(i, s) = d(i, s')$*, that is, the same actions are available to agent* i *in indistinguishable states.*
- $\rightarrow \subseteq S \times Act^{Ag} \times S$ *is the* transition relation *such that, for every state* $s \in S$ *and joint action* $\vec{a} \in Act^{Ag}$*. We write* $s \xrightarrow{\vec{a}} r$ *for* $(s, \vec{a}, r) \in \rightarrow$*. Moreover,* $s \xrightarrow{\vec{a}} r$ *only if* $a_i \in d(i, s)$ *for every agent* $i \in Ag$*.*
- AP *is a set of* atomic propositions *and* $\pi : S \to 2^{AP}$ *is the* state-labeling function*.*

By Def. 1 in a given state s, each agent $i \in Ag$ can perform the enabled actions in $d(i, s)$. A joint action $\vec{a}$ fires a transition from state s to some state s' only if each a_i is enabled for agent i in s. Further, each agent i is equipped with an indistinguishability relation $\sim_i$, with $s \sim_i s'$ meaning that i cannot tell state s from state s', i.e., agent i possesses the same information in the two states. In particular, the same actions are enabled in indistinguishable states.

Given an iCGS $\mathcal{G}$ as above, a *run* is a finite or infinite sequence $\lambda = s_0\vec{a}_0 s_1 \ldots$ in $((S{\cdot}Act^{Ag})^*{\cdot}S) \cup (S{\cdot}Act^{Ag})^\omega$ such that for every $j \geqslant 0$, $s_j \xrightarrow{\vec{a}_j} s_{j+1}$. Given a run $\lambda = s_0\vec{a}_0 s_1 \ldots$ and $j \geqslant 0$, $\lambda[j]$ denotes the $j+1$-th state s_j in the sequence. For a group $A \subseteq Ag$ of agents, a *joint A-action* denotes a tuple $\vec{a}_A = (a_i)_{i\in A} \in Act^A$ of actions, one for each agent in A. For groups $A \subseteq B \subseteq Ag$ of agents, a joint A-action $\vec{a}_A$ *is extended* by a joint B-action $\vec{b}_B$, denoted $\vec{a}_A \sqsubseteq \vec{b}_B$, if for every $i \in A$, $a_i = b_i$. Also, a joint A-action $\vec{a}_A$ is *enabled* at state $s \in S$ if for each agent $i \in A$, $(a_A)_i \in d(i, s)$.

We now introduce a notion of strategy adapted to iCGS with imperfect information [26].

DEFINITION 2. *A* (uniform) strategy *for an agent* $i \in Ag$ *is a function* $\sigma : S \to Act$ *that is compatible with* d *and* $\sim_i$*, i.e.,*

- *for every state* $s \in S$*,* $\sigma(s) \in d(i, s)$*;*
- *for all states* $s, r \in S$*,* $s \sim_i r$ *implies* $\sigma(s) = \sigma(r)$*.*

By Def. 2 a strategy in an iCGS has to be uniform in the sense that in indistinguishable states it must return the same action. Such strategies are also known as *observational* in the literature on game theory. Note that in this paper we use memoryless strategies, whereby only the current state determines the action to perform. This choice is dictated by the application in hand, namely voting protocols, in which each agent's memory is encoded in the agent's state[1]. Perfect recall strategies with imperfect information can be defined similarly, as memoryless strategies on tree unfoldings of iCGS. We leave this extension for future work.

A strategy for a group A of agents is a family $\sigma_A = \{\sigma_a \mid a \in A\}$ of strategies, one for each agent in A. Given groups $A \subseteq B \subseteq Ag$, a strategy σ_A for group A, a state $s \in S$, and a joint B-action $\vec{b}_B \in Act^B$ that is enabled at s, we say that $\vec{b}_B$ is *compatible with* σ_A *(in* s*)* whenever $\sigma_A(s) \sqsubseteq \vec{b}_B$. For states $s, r \in S$ and strategy σ_A, we denote $s \xrightarrow{\sigma_A(s)} r$ if $s \xrightarrow{\vec{a}} r$ for some joint action $\vec{a} \in Act^{Ag}$ that is compatible with σ_A.

We define two notions of *outcomes* of strategy σ_A at state s, corresponding to the *objective* and *subjective* interpretation of ATL operators. Fix a state s and a strategy σ_A for group A.

1. The set of *objective outcomes of* σ_A *at* s is defined as $out^{\mathcal{G}}_{obj}(s, \sigma_A) = \{\lambda \in Run(\mathcal{G}) \mid \forall j \geqslant 0, \lambda[j] \xrightarrow{\sigma_A(\lambda[j])} \lambda[j+1]\}$.

[1]Therefore memoryless strategies already encode the agent's memory of all her past observations.

2. The set of *subjective outcomes of* σ_A *at* s is defined as $out^{\mathcal{G}}_{subj}(s,\sigma_A) = \bigcup_{i\in A, s'\sim_i s} out^{\mathcal{G}}_{obj}(s',\sigma_A)$.

DEFINITION 3. *The set of ATL formulas* φ *is defined by the following BNF:*

$$\varphi \quad ::= \quad p \mid \neg\varphi \mid \varphi \to \varphi \mid \langle\langle A\rangle\rangle X\varphi \mid \langle\langle A\rangle\rangle \varphi U \varphi \mid \langle\langle A\rangle\rangle \varphi R \varphi$$

where $p \in AP$ *and* $A \subseteq Ag$.

The ATL operator $\langle\langle A\rangle\rangle$ intuitively means that 'the agents in group A have a (collective) strategy to achieve ...', where the goals are LTL formulas built by using operators 'next' X, 'until' U, and 'release' R. Note that the 'release' operator R cannot be defined in ATL with imperfect information by using 'until' U and 'globally' G, as it is the case in perfect information contexts [29], so we include it for completeness. We define A-formulas as the formulas in ATL in which A is the only group appearing in ATL modalities.

Traditionally, ATL under imperfect information has been given either state-based or history-based semantics, and several variations have been considered on the interpretation of strategy operators. Here we present both the objective and subjective variants of the state-based semantics with imperfect information and imperfect recall.

DEFINITION 4. *Given an iCGS* $\mathcal{G}$*, an ATL formula* φ*, the* subjective *(resp.* objective*) semantics of* φ *at state* s*, denoted* $(\mathcal{G},s) \vDash_x \varphi$ *for* $x = subj$ *(resp.* $x = obj$*), is defined recursively as follows:*

$(\mathcal{G},s) \vDash_x p$ *iff* $p \in \pi(s)$

$(\mathcal{G},s) \vDash_x \neg\varphi$ *iff* $(\mathcal{G},s) \nvDash_x \varphi$

$(\mathcal{G},s) \vDash_x \varphi \wedge \varphi'$ *iff* $(\mathcal{G},s) \vDash_x \varphi$ *and* $(\mathcal{G},s) \vDash_x \varphi'$

$(\mathcal{G},s) \vDash_x \langle\langle A\rangle\rangle X\varphi$ *iff* $\exists\sigma_A\ \forall\lambda \in out^{\mathcal{G}}_x(s,\sigma_A), (\mathcal{G},\lambda[1]) \vDash_x \varphi$

$(\mathcal{G},s) \vDash_x \langle\langle A\rangle\rangle \varphi U \varphi'$ *iff* $\exists\sigma_A\ \forall\lambda \in out^{\mathcal{G}}_x(s,\sigma_A), \exists j \geqslant 0$ *with* $(\mathcal{G},\lambda[j]) \vDash_x \varphi'$ *and* $\forall 0 \leqslant k < j, (\mathcal{G},\lambda[k]) \vDash_x \varphi$

$(\mathcal{G},s) \vDash_x \langle\langle A\rangle\rangle \varphi R \varphi'$ *iff* $\exists\sigma_A\ \forall\lambda \in out^{\mathcal{G}}_x(s,\sigma_A)$, *either* $\forall j \geqslant 0$, $(\mathcal{G},\lambda[j]) \vDash_x \varphi$, *or* $\exists k \geqslant 0$ *with* $(\mathcal{G},\lambda[k]) \vDash_x \varphi'$ *and* $\forall 0 \leqslant l \leqslant k, (\mathcal{G},\lambda[l]) \vDash_x \varphi$

REMARK 5. *The knowledge operator* K_i *can be appended to the syntax of ATL with the following semantics:*

$$(\mathcal{G},s) \vDash_x K_i\varphi \text{ iff } \forall s' \in S, s' \sim_i s \text{ implies } (\mathcal{G},s') \vDash_x \varphi$$

By considering the subjective interpretation of ATL, this operator can be derived: $(\mathcal{G},s) \vDash_{subj} K_i\varphi$ *iff* $(\mathcal{G},s) \vDash_{subj} \langle\langle i\rangle\rangle \varphi U \varphi$. *There exists no such definition for the knowledge operator in ATL with the objective semantics.*

3. SIMULATIONS AND BISIMULATIONS

In this section we define simulation and bisimulation relations on iCGS with imperfect information and perfect recall. The main result we prove is that bisimulations preserve the interpretation of formulas in ATL. We start by introducing relevant notions that will be used in the rest of the paper.

A *partial strategy* for agent $i \in Ag$ is a partial function $\sigma : S \to Act$ such that for each $s_1, s_2 \in S$, if $s_1 \sim_i s_2$ then $\sigma(s_1) = \sigma(s_2)$. We denote the domain of the partial strategy σ as $dom(\sigma)$. Given a group $A \subseteq Ag$, a *partial strategy profile for* $A \subseteq Ag$ is a tuple $(\sigma_i)_{i\in A}$ of partial strategies, one for each agent $i \in A$. The set of partial strategy profiles for A is denoted $PStr_A$. Given a set $U \subseteq S$ of states and a group $A \subseteq Ag$, we denote $PStr_A(U)$ the set of partial strategies whose domain is U:

$$PStr_A(U) = \left\{(\sigma_i)_{i\in A} \in PStr_A \mid dom(\sigma_i) = U \text{ for all } i \in A\right\}$$

Given a group $A \subseteq Ag$ of agents, the *collective knowledge relation* $\sim^E_A$ is defined as $\bigcup_{i\in A} \sim_i$, while the *common knowledge relation* $\sim^C_A$ is the transitive closure $(\bigcup_{i\in A} \sim_i)^+$ of $\sim^E_A$. Then, $E^{\mathcal{G}}_A(q) = \{q' \in S \mid q' \sim^E_A q\}$ and $C^{\mathcal{G}}_A(q) = \{q' \in S \mid q' \sim^C_A q\}$ are respectively the *collective* and *common knowledge neighbourhoods* of state q for group A in the iCGS $\mathcal{G}$.

DEFINITION 6 (SIMULATION). *Given two iCGS* $\mathcal{G} = \langle Ag, AP, S, s_0, \{\sim_i\}_{i\in Ag}, Act, d, \to, \pi\rangle$ *and* $\mathcal{G}' = \langle Ag, AP, S', s'_0, \{\sim'_i\}_{i\in Ag}, Act', d', \to', \pi'\rangle$ *sharing the set of agents* Ag *and the set of atoms* AP*, and a group* $A \subseteq Ag$ *of agents, a relation* $\Rrightarrow_A \subseteq S \times S'$ *is a* simulation for A *iff* $q \Rrightarrow_A q'$ *implies that*

1. $\pi(q) = \pi'(q')$;
2. *For every* $i \in A$ *and* $r' \in S'$, *if* $q' \sim'_i r'$ *then for some* $r \in S$ *we have that* $q \sim_i r$ *and* $r \Rrightarrow_A r'$.
3. *By denoting* $C_A(q) = C^{\mathcal{G}}_A(q)$ *and* $C'_A(q) = C^{\mathcal{G}'}_A(q)$, *there exists a mapping* $ST = ST_{C_A(q),C'_A(q')}$ *with* $ST : PStr_A(C_A(q)) \to PStr_A(C'_A(q'))$ *such that for any two states* $r \in C_A(q)$, $r' \in C'_A(q')$, *if* $r \Rrightarrow_A r'$ *then the following two properties hold:*
   (a) *For every partial strategy* $\sigma_A \in PStr_A(C_A(q))$ *and state* $s' \in S'$, *if* $r' \xrightarrow{ST(\sigma_A)(r')} s'$ *then there exists some state* s *such that* $r \xrightarrow{\sigma_A(r)} s$ *and* $s \Rrightarrow_A s'$.
   (b) $ST_{C_A(q),C'_A(q')} = ST_{C_A(r),C'_A(r')}$.

A relation $\Lleftrightarrow_A$ is a *bisimulation* iff both $\Rrightarrow_A$ and $\Rrightarrow_A^{-1} = \{(q',q) \mid q \Rrightarrow_A q'\}$ are simulations.

Intuitively, by Def. 6 state q' *simulates* q, i.e., $q \Rrightarrow_A q'$ implies that (1) q and q' agree on the interpretation of atoms; (2) q simulates the epistemic transitions from q'; and (3) for every partial strategy σ_A, defined on the common knowledge neighborhood $C_A(q)$, we are able to find some partial strategy $ST(\sigma_A)$ (the same for all states in $C_A(q)$) such that the transition relations $\xrightarrow{ST(\sigma_A)}$ and $\xrightarrow{\sigma_A}$ commute with the simulation relation $\Rrightarrow_A$.

REMARK 7. *The problem of checking for the existence of a bisimulation between two iCGS, for some set of agents* A *is in PSPACE.*

In order to prove that bisimilar states satisfy the same formulas in ATL, we prove the following auxiliary result.

PROPOSITION 8. *If* $q \Rrightarrow_A q'$ *then for every uniform strategy* σ_A*, there exists a uniform strategy* σ'_A *such that*

(*) *For every run* $\lambda' \in out^{\mathcal{G}'}_x(q',\sigma'_A)$, *for* $x \in \{subj, obj\}$, *there exists an infinite run* $\lambda \in out^{\mathcal{G}}_x(q,\sigma_A)$ *such that* $\lambda(i) \Rrightarrow_A \lambda'(i)$ *for every* $i \geqslant 0$.

PROOF. We will inductively define a sequence of partial uniform strategy profiles $(\overline{\sigma}^n_A)_{n\in\mathbb{N}} \in PStr_A$ with $\overline{\sigma}^n_A = (\overline{\sigma}^n_i)_{i\in A}$ and $dom(\overline{\sigma}^n_A) \subseteq dom(\overline{\sigma}^{n+1}_A)$ for each $n \in \mathbb{N}$. These partial

strategies will be constructed using the strategy σ_A and the mapping ST from point 3 in Def. 6 of simulation.

Define first the sequence $dom^n(\sigma_A, q)$, for $n \in \mathbb{N}$, of sets of $\mathcal{G}$-states such that $s \in dom^n(\sigma_A, q)$ if s can be reached in at most n steps from q by applying actions compatible with strategy σ_A:

$$dom^0(\sigma_A, q) = \varnothing, \qquad dom^1(\sigma_A, q) = C_A(q)$$

$$dom^{n+1}(\sigma_A, q) = dom^n(\sigma_A, q) \cup \{r \mid \exists s \in dom^n(\sigma_A, q), s \xrightarrow{\sigma_A(s)} r\}$$

Also, denote σ_A^n the partial strategy resulting from restricting σ_A to $dom^n(\sigma_A, q)$ (and setting it as undefined on the complement of this set).

Further, consider some total order $\precsim$ on S. Then, $\min_{\precsim} U$ is the minimum of U w.r.t. $\precsim$. Similarly, we assume that S' is endowed with a total order $\precsim'$.

The desired sequence of partial uniform strategy profiles $\overline{\sigma}_A^n$, for $n \geqslant 1$, is defined as $\overline{\sigma}_A^1(r)' = ST_{C_A(u), C'_A(u')}(\sigma_A^1)(r')$ and

$$\overline{\sigma}_A^{n+1}(r') = \begin{cases} \overline{\sigma}_A^n(r'), \text{for } r' \in dom(\overline{\sigma}_A^n) \\ ST_{C_A(u), C'_A(u')}(\sigma_A^{n+1})(r'), \text{ for } C'_A(r') \cap ran(\overline{\sigma}_A^n) \neq \varnothing, \\ \quad r' \notin dom(\overline{\sigma}_A^n), u' = \min_{\precsim'} C'_A(r') \cap ran(\overline{\sigma}_A^n) \\ \quad \text{and } u = \min_{\precsim}\{v \in dom^n(\sigma_A, q) \mid v \Rightarrow_A u'\} \end{cases}$$

By induction on n we may easily observe that whenever $C'_A(r') \cap dom(\overline{\sigma}_A^n) \neq \varnothing$, then $C'_A(r') \subseteq dom(\overline{\sigma}_A^n)$.

We may also show that, whenever we take some $u' \in C'_A(r') \cap ran(\overline{\sigma}_A^n) \neq \varnothing$ with $r' \notin dom(\overline{\sigma}_A^n)$, we have $\{u \in dom^n(\sigma_A, q) \mid u \Rightarrow_A u'\} \neq \varnothing$. To see this, take some $u' \in C'_A(r') \cap ran(\overline{\sigma}_A^n)$, which implies that there exists $v' \in dom(\overline{\sigma}_A^n)$ with $v' \xrightarrow{\overline{\sigma}_A^n(v')} u'$. But this means, by definition, that $\overline{\sigma}_A^n(v') = ST_{C_A(v), C'_A(v')}(\sigma_A^n)(v')$ for some $v \Rightarrow_A v'$ with $v \in dom(\sigma_A^{n-1})$. From property 3.a, this implies the existence of u with $u \Rightarrow_A u'$ and $v \xrightarrow{\sigma_A(v)} u$, hence $v \in dom(\sigma_A^n)$.

We then prove by induction on n that $\overline{\sigma}_A^n$ is uniform. The case $n = 1$ is trivial. As regards the induction step, note first that, if $r'_1, r'_2 \in dom(\overline{\sigma}_A^{n+1})$, $C'_A(r'_1) = C'_A(r'_2)$ and if $r'_1 \in dom(\overline{\sigma}_A^n)$ then $r'_2 \in dom(\overline{\sigma}_A^{n+1})$ as well.

On the other hand, for $r'_1, r'_2 \in dom(\overline{\sigma}_A^{n+1}) \setminus dom(\overline{\sigma}_A^n)$, if $r'_1 \sim_i r'_2$ for some $i \in A$, we have that $C'_A(r'_1) = C'_A(r'_2)$. Therefore, if we take $r' = \min_{\precsim'} C'_A(r'_1) \cap ran(\overline{\sigma}_A^n)$ and $r = \min_{\precsim}\{u \in dom^n(\sigma_A, q) \mid u \Rightarrow_A r'\}$ we have, by definition, $\overline{\sigma}_A^{n+1}(r'_1) = ST_{C_A(r), C'_A(r')}(\sigma_A^{n+1})(r'_1)$ and $\overline{\sigma}_A^{n+1}(r'_2) = ST_{C_A(r), C'_A(r')}(\sigma_A^{n+1})(r'_2)$ and these values are identical, since $ST_{C_A(r), C'_A(r')}(\sigma_A)$ is a uniform strategy in $\mathcal{G}'$.

As a result, the "limit" partial strategy profile $\overline{\sigma}_A = \bigcup_{n \in \mathbb{N}} \overline{\sigma}_A^n$, defined by $\overline{\sigma}_A(q) = \overline{\sigma}_A^n(q)$ whenever $q \in dom(\overline{\sigma}_A^n)$, is clearly uniform and has $dom(\overline{\sigma}_A) = ran(\overline{\sigma}_A)$. We then only need to transform it into a (total) uniform strategy profile by imposing a fixed action $a_0 \in Act$ wherever $\overline{\sigma}_A^n$ was undefined, that is, defining the following uniform strategy profile σ'_A:

$$\sigma'_A(r') = \begin{cases} \overline{\sigma}_A(r') & \text{for } r' \in dom(\overline{\sigma}_A) \\ a_0 & \text{otherwise} \end{cases}$$

To prove property (*) for the objective semantics, consider a run $\lambda' \in out^{\mathcal{G}'}_{obj}(q', \sigma'_A)$ and set $\lambda[0] = q$. We then build inductively the run λ as follows: assume $\lambda[k]$ has been built, with $\lambda[j] \Rightarrow_A \lambda'[j]$ for all $j \leqslant k$. Then we apply point 3.a from Def. 6 of simulation to the pair $\lambda[k] \Rightarrow_A \lambda'(k)$ and by using the fact that $\lambda'(k+1) \in \sigma'_A(\lambda[k]) = ST_{C_A(v), C'_A(v')}(\sigma_A)(\lambda'(k))$, where $v' = \min_{\precsim'} C'_A(\lambda'(k))$ and $v = \min_{\precsim}\{u \in dom^n(\sigma_A, q) \mid u \Rightarrow_A \lambda'(k)\}$, to obtain the existence of a state $u \in \sigma_A(\lambda[k])$ such that $u \Rightarrow_A \lambda'(k+1)$. This is the state that we choose for $\lambda[k+1]$, i.e., $\lambda[k+1] := u$.

To prove property (*) for the subjective semantics, consider a run $\lambda' \in out^{\mathcal{G}'}_{subj}(q', \sigma'_A)$. Hence, $\lambda'(0) \in C'_A(q')$. By applying point 2 of the definition of simulation (and a short induction on the length of the indistinguishability path connecting q' with $\lambda'(0)$), we obtain the existence of a state $r \in C_A(q)$ such that $r \Rightarrow_A \lambda'(0)$. Then set $\lambda[0] := r$. The rest of the construction of λ' is identical to the inductive case for the objective semantics, that is, for every k, we set $\lambda[k+1] := u$ where $u \in \sigma_A(\lambda[k])$ and $u \Rightarrow_A \lambda'(k+1)$.

The property (*) is then proved for both semantics. □

By using Proposition 8 we are finally able to prove the main preservation result of this paper.

THEOREM 9. *Given two iCGS $\mathcal{G}$ and $\mathcal{G}'$ and states $q \in S$, $q' \in S'$, suppose that $q \Longleftrightarrow_A q'$. Then for every A-formula φ,*

$$(\mathcal{G}, q) \vDash \varphi \quad \textit{if and only if} \quad (\mathcal{G}', q') \vDash \varphi$$

PROOF. The proof is by induction on the structure of φ.

The case for propositional atoms is immediate as $(\mathcal{G}, q) \vDash p$ iff $p \in \pi(q)$, iff $p \in \pi'(q'))$ by definition of bisimulation, iff $(\mathcal{G}', q') \vDash p$. The inductive case for propositional connectives is also straightforward.

For $\varphi = \langle\!\langle A \rangle\!\rangle \psi_1 U \psi_2$, $(\mathcal{G}, q) \vDash \varphi$ implies that for some strategy σ_A, for all $\lambda \in out^{\mathcal{G}}_x(q, \sigma_A)$, for some $j_1 \geqslant 0$, $(\mathcal{G}, \lambda[j_1]) \vDash \psi_2$ and for every $k < j_1$, $(\mathcal{G}, \lambda[k]) \vDash \psi_1$. Again by Proposition 8, there exists strategy σ'_A s.t. for all $\lambda' \in out^{\mathcal{G}'}_x(q', \sigma'_A)$, there exists $\lambda \in out^{\mathcal{G}}_x(q, \sigma_A)$ with $\lambda[j] \Longleftrightarrow_A \lambda'[j]$ for all $j \geqslant 0$. By the induction hypothesis, we get that for all j, $(\mathcal{G}, \lambda[j]) \vDash \psi_1$ iff $(\mathcal{G}', \lambda'[j]) \vDash \psi_1$ and similarly for ψ_2. Hence, $(\mathcal{G}', \lambda'[j_1]) \vDash \psi_2$ and $(\mathcal{G}', \lambda'[k]) \vDash \psi_1$ for all $k < j_1$, that is, $(\mathcal{G}', h') \vDash \varphi$.

The cases for $\varphi = \langle\!\langle A \rangle\!\rangle X \psi$ and $\varphi = \langle\!\langle A \rangle\!\rangle \psi_1 R \psi_2$ are proved similarly. □

By Theorem 9 we obtain that bisimilar states preserve the interpretation of ATL formulas. More precisely, if states q and q' are A-bisimilar then they satisfy the same A-formulas.

4. THREE-BALLOT VOTING PROTOCOL

ThreeBallot [39, 38] is a voting protocol that strives to achieve some desirable properties, such as anonymity and verifiability of voting, without the use of cryptography. The protocol proceeds as follows. Each voter identifies herself at the poll site, and gets a paper "multi-ballot" to vote with. The multi-ballot consists of three vertical ballots – identical except for ID numbers at the bottom, see Figure 1 (presented

BALLOT		BALLOT		BALLOT	
Alex Jones	○	Alex Jones	○	Alex Jones	●
Bob Smith	●	Bob Smith	●	Bob Smith	○
Carol Wu	○	Carol Wu	●	Carol Wu	○
3147524		7523416		5530219	

Figure 1: Three-ballot showing a vote for Bob Smith

after [39]). The voter fills in the multi-ballot, separates the three parts and casts them in the ballot box. To cast a vote for a candidate, one must mark exactly two (arbitrary) bubbles on the row of the candidate. To not vote for a candidate, one must mark exactly one of the bubbles on the candidate's row (again, arbitrary one). In all the other cases the vote is invalid. The ballots are tallied by counting the number of bubbles marked for each candidate, and then subtracting the number of voters from the count.

While voting, the voter also receives a copy of one of her three ballots, and she can take it home. After the election closes, all the ballots are scanned and published on the web bulletin board. In consequence, the voter can check if her receipt matches a ballot listed on the bulletin board. If no ballot matches the receipt, the voter can file a protest.

Since ThreeBallot is not a cryptographic protocol, it does not heavily rely on computers and counting can be done directly. Moreover, voters have no responsibility to ensure the integrity of cryptographic keys, and the security process in their vote is essentially the same as with traditional ballots.

Properties. ThreeBallot was proposed to provide several properties that reduce the possibility of electoral fraud.

Anonymity (cf. e.g. [34]) requires that no agent should ever know how another voter voted, except in cases when it is inevitable, such as when all the voters voted for the same candidate. Anonymity is important because it limits the opportunities of coercion and vote-buying. *Coercion-resistance* requires that the voter cannot reveal the value of her vote beyond doubt, even if she fully cooperates with the coercer. As a consequence, the coercer has no way of deciding whether to execute his threat (or, dually, pay for the vote). A preliminary formalization of coercion-resistance and receipt-freeness in ATL has been presented in [43].

Finally, *end-to-end voter verifiability* [41, 40] provides a way to verify the outcome of the election by allowing voters to audit the information published by the system. Typically, the focus is on individual verifiability: each voter should be able check if her vote has been taken into account and has not been altered.

4.1 iCGS Model

We present here three iCGS models of the Three-Ballot Voting system. All these models have been specified in ISPL (Interpreted System Programming Language), the input language of MCMAS. Several aspects of the voting system have not been modeled: the ID of each ribbon, the copy of the ribbon which is given back to each voter after casting his/her ballot, the possibility for voters to verify the presence of the ribbon they are given back after voting. We model a single attacker who is also a voter and, as such, must obey the voting protocol and does not interact in any particular way with the other agents.

In the iCGS below, each agent is represented by means of its local variables and their evolution. The vote collector and bulletin board (BB) are modeled by the Environment agent (call it Env). This agent contains local variables modeling the fact that the voting process is open and the values of ribbons on the BB. These variables are observable by all voters, including the attacker. Env also contains private variables used for collecting ribbons and disposes of the three actions $Act_e = \{stop, collect, nop\}$ for waiting closing elections, collecting votes and, finally, looping after the end of the publication of the BB.

Elections are closed immediately after the voting starts. This peculiarity of our models avoids us dealing with a vote collector which never stops the voting process, which may lead to the vacuous falsity of the formulas checked unless some fairness property is enforced – and, for the time being, fairness is not handled by our alternating bisimulation.

The agents representing voters have each a private variable representing their choice for a candidate. Then they share three "ballot" variables with Env. These variables represent the ribbons that are created by the "voting machine". Casting the vote is modeled by creating the three ribbons, compatible with the choice of each candidate. Votes are already cast in the initial state. Being visible by Env, the values of the three ribbons are copied by Env on the (variables represented on the) BB in a random order. Each agent has two actions: *vote*, by which the voter casts his/her vote, and *nop*, a non-voting or idle action. *vote* is enabled only in the initial state, *nope* is enabled everywhere. All agent variables are never modified during the voting process.

In the first model, denoted $\mathcal{G}_{tot}$, for each agent choice, all configurations of the three ribbons which are compatible with the agent's choice may occur. The communication between each agent and Env is entirely at Env's charge, who has direct access to agents' ribbons and copies them onto the BB. Copying is also done at random: Env chooses a non-copied ribbon from a voter who has cast his vote (boolean variables are defined to help Env identify these situations) and copies it onto a free position on the BB.

With the second model, denoted $\mathcal{G}_{lex}$, we model a voting machine which sorts, according to the lexicographic order, the three ribbons produced for the agent's choice, and places the largest one in the first "ballot" variable of the voter, the second largest in the second variable, and the smallest in the third variable. Hence, for each choice of an agent, there are still several configurations of ribbons that are produced, but we no longer produce all permutations of a configuration, but a single representative of that permutation.

Finally, we modify $\mathcal{G}_{lex}$ into a third model, in which Env no longer copies ribbons on the BB, but rather counts the votes for each candidate by peeping at the "ballot" variables of each voter. This model is denoted $\mathcal{G}_{count}$.

Formally, in the case of $\mathcal{G}_{tot}$ for n voters and nc candidates, each global state has the form $(vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (ch_i, v_i)_{1\leqslant i\leqslant n}, (s_{ij}, a_{ij})_{1\leqslant i\leqslant n, 1\leqslant j\leqslant 3})$ where:

1. The local state for voter i is $(vopen, pub, ribb_1, ribb_2, ...ribb_{3n}, v_i, s_{i1}, s_{i2}, s_{i3})$
2. Boolean *vopen* holds true when the vote is opened and *pub* signals that all ribbons of agents that have voted are published on the BB.
3. Integer $1 \leqslant ch_i \leqslant nc$ specifies the choice of agent i.
4. Boolean v_i $(1 \leqslant i \leqslant n)$ tells whether agent i has voted.
5. Integer variables s_{ij} $(1 \leqslant j \leqslant 3)$ represent the "ballots" of voter i. They are shared between each agent and Env, who copies them onto the BB.
6. Integer variables $ribb_\ell$ $(1 \leqslant \ell \leqslant 3n)$ represent the BB.
7. Booleans a_{ij} are used by Env for remembering which ballots s_{ij} have been copied on the BB.

Initial states are such that $vopen = true$, $v_i = false$ for all $i \leqslant n$, variables $ribb_\ell$ are *undefined* value $\perp$, $a_{ij} = false$ and, for variables s_{ij} we have the following rules modeling the creation of a triple of ribbons compatible with a choice of a candidate: for each voter i, let $b_{jk} = b^i_{jk}$ be the bit representing the bubble on the line corresponding with candidate k of the jth ballot of i's vote, as given by ch_i. A tuple $(b_{jk})_{1\leqslant j\leqslant 3,1\leqslant k\leqslant nc}$ is *compatible* with choice ch_i if the following properties hold:

1. if $k = ch_i$ then $\exists p \leqslant 3$ s.t. $b_{pk} = 0$ and $\forall p' \neq p$, $b_{p'k} = 1$
2. if $k \neq ch_i$ then $\exists p \leqslant 3$ s.t. $b_{pk} = 1$ and $\forall p' \neq p$, $b_{p'k} = 0$

Denote $B(ch_i)$ the set of bit tuples $(b_{jk})_{1\leqslant j\leqslant 3,1\leqslant k\leqslant nc}$ compatible with ch_i. Denote further by $R(ch_i)$ the transformation of these bit tuples into integer triples modeling the valid ballots compatible with the choice ch_i, $R(ch_i) = \{(st_j)_{1\leqslant j\leqslant 3} \mid st_j = \sum_{1\leqslant k\leqslant nc} b_{jk}\cdot 2^{k-1}, (b_{jk})_{1\leqslant j\leqslant 3,1\leqslant k\leqslant nc} \in B(ch_i)\}$. (For instance, valid triples of integers compatible with a voting intention for candidate 2 and $nc = 2$ are all permutations of $(3,2,0)$ plus all permutations of $(2,2,1)$.)

Then $(s_{ij})_{1\leqslant j\leqslant 3} \in R(ch_i)$ for each $1 \leqslant i \leqslant n, 1 \leqslant j \leqslant 3$.

Transitions are then of the form: $(vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (ch_i, v_i)_{1\leqslant i\leqslant n}, (s_{ij}, a_{ij})_{1\leqslant i\leqslant n,1\leqslant j\leqslant 3}) \xrightarrow{(a_e,a_1,a_2,\ldots,a_n)} (vopen', pub', (ribb'_\ell)_{1\leqslant\ell\leqslant 3n}, (ch'_i, v'_i)_{1\leqslant i\leqslant n}, (s'_{ij}, a'_{ij})_{1\leqslant i\leqslant n,1\leqslant j\leqslant 3})$ with:

1. $vopen' = false$ if ($a_e = stop$ or $vopen = false$) and $vopen' = true$ otherwise. Action $a_e = stop$ is the only available action for Env if $vopen = true$.
2. For $a_i = vote$, $v'_i = true$, and for $a_i = nop$, $v'_i = v_i$.
3. For $a_e = collect$ and $a_i = nop$ for all i we have the following rules:
   (a) There exists some subset of pairs $A \subseteq \{1,..,n\} \times \{1,..,3\}$ with $a'_{ij} = a_{ij} = true$ for all $(i,j) \in A$.
   (b) There exists $(i_0, j_0) \notin A$ with $a'_{i_0,j_0} = true$, $a_{i_0,j_0} = false$ and for all $(i,j) \notin A \cup \{(i_0,j_0)\}$, $a'_{ij} = false$.
   (c) There exists some $B \subseteq \{1,..,3n\}$ with $card(B) = card(A)$, $ribb'_\ell = ribb_\ell$ for all $\ell \in B$.
   (d) There exists some $k \notin B$, $1 \leqslant k \leqslant 3n$ with $ribb_k = \perp$, $ribb'_k = s_{i_0,j_0}$ and $ribb'_\ell = \perp$ for all $\ell \notin B \cup \{k\}$.
4. Action $a_e = nop$ can only be executed when, for each i, either all $a_{ij} = true$ or $v_i = false$, and its effect is to modify only $pub' = true$, all the other variables remaining unchanged.

In $\mathcal{G}_{lex}$, transitions are identical to the above, the only difference being in the initial states, more specifically in the configuration of variables s_{ij}. These are instantiated such that $(s_{ij})_{1\leqslant j\leqslant 3} \in \{max(Perm((st_j)_{1\leqslant j\leqslant 3})) \mid (st_j)_{1\leqslant j\leqslant 3} \in R_{ch_i}\}$ for each $1 \leqslant i \leqslant n$, the maximum being considered under the lexicographic order and $Perm((st_j)_{1\leqslant j\leqslant 3})$ stands for the set of all permutations of the tuple $(st_j)_{1\leqslant j\leqslant 3}$.

Finally, the iCGS $\mathcal{G}_{count}$ is similar with $\mathcal{G}_{lex}$ but all variables $ribb_\ell$ are replaced with nc variables $(co_k)_{1\leqslant k\leqslant nc}$. The local state for agent i is then $(vopen, pub, co_1, \ldots, co_{nc}, v_i, s_{i1}, s_{i2}, s_{i3})$. The description of transitions is then the same, excepting the case for $a_e = collect$ and items 3.(c)-3.(d) above (defining the updates of variables $ribb_\ell$), which are replaced by the following:

3.(c') For each $1 \leqslant k \leqslant nc$, if $a'_{ij} \neq a_{ij}$ then $co'_k = co_k + b_{ijk}$, where b_{ijk} is the k-th least significant bit of s_{ij}, otherwise $co'_k = co_k$.

4.2 Bisimulations for $\mathcal{G}_{tot}$, $\mathcal{G}_{lex}$ and $\mathcal{G}_{count}$

The three models defined in the previous section seem naturally related w.r.t. some properties – in particular those related with the attacker modifying the outcome of the vote or breaking the anonymity. The interest in simplifying the model is that checking the coercion resistance property can be done faster and with less memory on $\mathcal{G}_{count}$ than on $\mathcal{G}_{lex}$, which, on its turn, requires less time and memory than $\mathcal{G}_{tot}$, as we will see in the last section. In this section we show that the three models are bisimilar for the attacker, for the set of atomic propositions that refer only to choices of the agents. The fact which formalizes the "natural relation" between them and allows us to check a coercion resistance property on the simplest one and then generalizing the results on the two others, in particular on the largest model. Note that this bisimulation works because the properties do not refer to the status of the BB. For instance, these bisimulations would not be useful for simplifying systems for verifiability [18].

Formally, for each choice for an agent i to vote for a candidate j, we utilize an atomic proposition $p_{ch_i=j}$, which holds true only in those states in which $ch_i = j$. Then if we denote the attacker $att = n$ and $AP = \{p_{ch_i=j} \mid 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant nc\}$, the following relation is an $\{att\}$-bisimulation over AP between $\mathcal{G}_{tot}$ and $\mathcal{G}_{lex}$: $(vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (ch_i, v_i)_{1\leqslant i\leqslant n}, (s_{ij}, a_{ij})_{1\leqslant i\leqslant n,1\leqslant j\leqslant 3}) \Longleftrightarrow^1_{\{att\}} (vopen', pub', (ribb'_\ell)_{1\leqslant\ell\leqslant 3n}, (ch'_i, v'_i)_{1\leqslant i\leqslant n}, (s'_{ij}, a'_{ij})_{1\leqslant i\leqslant n,1\leqslant j\leqslant 3})$ iff the following hold:

1. $vopen = vopen'$, $pub = pub'$, $v_i = v'_i$, $ch_i = ch_i$ for all $1 \leqslant i \leqslant n$ and $ribb_\ell = ribb'_\ell$ for all $1 \leqslant \ell \leqslant 3n$.
2. For each $1 \leqslant i \leqslant n$, if we denote b_{jk} the kth least significant bit of s_{ij} and b'_{jk} the kth bit of s'_{ij}, then both $(b_{jk})_{1\leqslant j\leqslant 3,1\leqslant k\leqslant nc}, (b'_{jk})_{1\leqslant j\leqslant 3,1\leqslant k\leqslant nc} \in B(ch_i)$.
3. Denote ρ_i the S_3-permutation of (s_{i1}, s_{i2}, s_{i3}) into $(s'_{i1}, s'_{i2}, s'_{i3})$, i.e. $s_{ij} = s'_{i\rho_i(j)}$. Also when $s_{ij} = s'_{ij} = \perp$ we put $\rho_i = id_{\{1,2,3\}}$. Then $a_{ij} = a'_{i\rho_i(j)}$ for all i, j.

Stated differently, the 3rd item above says that (b'_{jk}) is the largest, in lexicographic order, among all tuples in B_{ch_i} which are permutations of (b_{jk}).

For $\mathcal{G}_{lex}$ and $\mathcal{G}_{count}$, we may consider the following $\{att\}$-bisimulation over AP: $(vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (ch_i, v_i)_{1\leqslant i\leqslant n}, (s_{ij}, a_{ij})_{1\leqslant i\leqslant n,1\leqslant j\leqslant 3}) \Longleftrightarrow^2_{\{att\}} (vopen', pub', (co_k)_{1\leqslant k\leqslant nc}, (ch'_i, v'_i)_{1\leqslant i\leqslant n}, (s'_{ij}, a'_{ij})_{1\leqslant i\leqslant n,1\leqslant j\leqslant 3})$ where:

1. $vopen = vopen'$, $pub = pub'$, $v_i = v'_i$, $ch_i = ch_i$, $s_{ij} = s_{ij}$ and $a_{ij} = a'_{ij}$ for all $1 \leqslant i \leqslant n, 1 \leqslant j \leqslant 3$.
2. For each $1 \leqslant \ell \leqslant 3n$ and $1 \leqslant k \leqslant nc$, if we denote $b_{\ell k}$ the kth least significant bit on the ribbon $ribb_\ell$, then:
$$co_k = \sum \{b_{\ell k} \mid ribb_\ell \neq \perp, 1 \leqslant \ell \leqslant 3n\}$$

To prove that these relations are indeed alternating bisimulations, note that the condition 1 is trivially satisfied as whenever $q \Longleftrightarrow^\iota_{\{att\}} q'$ ($\iota = 1, 2$), we must have that $(ch_i = j) \in q$ iff $(ch_i = j) \in q'$.

To prove properties 2 and 2-dual for $\Longleftrightarrow^1_{\{att\}}$, take states q, r in $\mathcal{G}_{tot}$ and $r \in \mathcal{G}_{lex}$ with $q \Longleftrightarrow^1_{\{att\}} q'$, $q \sim_{att} r$. Then

$q = (vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (ch_i, v_i)_{1\leqslant i\leqslant n}, (s_{ij}, a_{ij})_{1\leqslant i\leqslant n,1\leqslant j\leqslant 3})$

$q' = (vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (ch_i, v_i)_{1\leqslant i\leqslant n}, (s'_{ij}, a'_{ij})_{1\leqslant i\leqslant n,1\leqslant j\leqslant 3})$

$r = (vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (\overline{ch}_i, \overline{v}_i)_{1\leqslant i\leqslant n}, (\overline{s}_{ij}, \overline{a}_{ij})_{1\leqslant i\leqslant n,1\leqslant j\leqslant 3})$

with q, q' related by the definition of $\Longleftrightarrow^1_{\{att\}}$ above and $\overline{ch}_{att} = ch_{att}$, $\overline{s}_{att,j} = s'_{att,j}$ and $\overline{a}_{att,j} = a'_{att,j}$ for all $1 \leqslant j \leqslant 3$. Put then

$$r' = (vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (ch'_i, v'_i)_{1\leqslant i\leqslant n-1}, (\overline{ch}_{att}, \overline{v}_{att}), (s'_{ij}, a'_{ij})_{1\leqslant i\leqslant n-1, 1\leqslant j\leqslant 3}, (\overline{s}_{att,j}, \overline{a}_{att,j})_{1\leqslant j\leqslant 3})$$

and we get the desired result: $q' \Longleftrightarrow^1_{\{att\}} r'$ and $r \sim_{att} r'$. The mirror argument also works: for $q \Longleftrightarrow^1_{\{att\}} q'$ and $q' \sim_{att} r'$ we may choose $q \Longleftrightarrow^1_{\{att\}} r$ and $r \sim_{att} r'$.

Conditions 2 and 2-dual for $\Longleftrightarrow^2_{\{att\}}$ can be proved similarly, by observing that, $q \Longleftrightarrow^2_{\{att\}} q'$ and $q \sim_{att} r$ with $q, r \in \mathcal{G}_{lex}$ and $q' \in \mathcal{G}_{count}$, then:

$$q = (vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (ch_i, v_i)_{1\leqslant i\leqslant n}, (s_{ij}, a_{ij})_{1\leqslant i\leqslant n, 1\leqslant j\leqslant 3})$$
$$q' = (vopen, pub, (co_k)_{1\leqslant k\leqslant nc}, (ch_i, v_i)_{1\leqslant i\leqslant n}, (s'_{ij}, a'_{ij})_{1\leqslant i\leqslant n, 1\leqslant j\leqslant 3})$$
$$r = (vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (\overline{ch}_i, \overline{v}_i)_{1\leqslant i\leqslant n}, (\overline{s}_{ij}, \overline{a}_{ij})_{1\leqslant i\leqslant n, 1\leqslant j\leqslant 3})$$

with the same relationship between variables of q and r; and then, for

$$r' = (vopen, pub, (co_k)_{1\leqslant k\leqslant nc}, (ch'_i, v'_i)_{1\leqslant i\leqslant n-1}, (\overline{ch}_{att}, \overline{v}_{att}), (s'_{ij}, a'_{ij})_{1\leqslant i\leqslant n-1, 1\leqslant j\leqslant 3}, (\overline{s}_{att,j}, \overline{a}_{att,j})_{1\leqslant j\leqslant 3})$$

we get $q' \Longleftrightarrow^2_{\{att\}} r'$ and $r \sim_{att} r'$.

Finally, for conditions 3 and 3-dual, note first that for any state $q \in \mathcal{G}_{tot}$ or $q \in \mathcal{G}_{lex}$, $C_{att}(q)$ is just the equivalence class of q w.r.t. $\sim_{att}$, that is, if $q = (vopen, pub, (ribb_\ell)_{1\leqslant\ell\leqslant 3n}, (ch_i, v_i)_{1\leqslant i\leqslant n}, (s_{ij}, a_{ij})_{1\leqslant i\leqslant n, 1\leqslant j\leqslant 3})$, then $C_{att}(q)$ is composed of all states with local state for att of the form $((ribb_\ell)_{1\leqslant\ell\leqslant 3n}, ch_{att}, v_{att}, (s_{att,j}, a_{att,j})_{1\leqslant j\leqslant 3})$. Similarly, for $q' \in \mathcal{G}_{count}$ with $q' = (vopen, pub, (co'_k)_{1\leqslant k\leqslant nc}, (ch'_i v'_i)_{1\leqslant i\leqslant n}, (s'_{ij}, a'_{ij})_{1\leqslant i\leqslant n, 1\leqslant j\leqslant 3})$, $C_{att}(q')$ is composed of all states with local state for att composed of $((co_k)_{1\leqslant k\leqslant nc}, ch_{att}, v_{att}, (s_{att,j}, a_{att,j})_{1\leqslant j\leqslant 3})$.

Note first that, in all three iCGS, on each neighborhood $C_{att}(q)$, only one or two partial strategies for att can be defined, depending whether the vote is open or not. Therefore, we may define the mapping $ST_{C^{\mathcal{G}_{tot}}_{att}(\cdot), C^{\mathcal{G}_{lex}}_{att}(\cdot)}$ to associate to each partial strategy prescribing nop to att in some $C^{\mathcal{G}_{tot}}_{att}(q)$ the quasi-identical strategy prescribing the same action in $C^{\mathcal{G}_{lex}}_{att}(q')$, and, similarly, to the partial strategy prescribing $vote$ to att in $C^{\mathcal{G}_{tot}}_{att}(q)$ the strategy prescribing $vote$ in $C^{\mathcal{G}_{lex}}_{att}(q')$. The dual mapping ST' is defined similarly, and the same definitions work for the bisimulation $\Longleftrightarrow^2_{\{att\}}$. Note that these definitions already satisfy property 3.(b) of bisimulations.

To prove property 3.(a), consider first the strategy $vote_{att}$ prescribing $vote$ for att on $C^{\mathcal{G}_{tot}}_{att}(q)$ and take some state r with $q' \xrightarrow{vote_{att}} r'$. Since when voting is enabled, Environment does not collect votes, r' has the same BB as q' and all booleans a_{ij} are false. Therefore, we may choose the state r which has the same local variables as r' for all voters and the same BB as q, and get $q' \xrightarrow{ST(vote_{att})} r'$ and $r \Longleftrightarrow^1_{\{att\}} r'$. A similar proof works for $\Longleftrightarrow^2_{\{att\}}$.

Consider now the strategy $none_{att}$ prescribing action nop for att on $C^{\mathcal{G}_{tot}}_{att}(q)$ and take again some state r' with $q' \xrightarrow{nop_{att}} r'$. Note that this action is only enabled when the vote is closed. Then the only agent which executes a non-idle action on the above transition is Environment. By this transition, Environment copies one of the ribbons onto the BB. This transition can then be simulated in $\mathcal{G}_{tot}$ by copying the same ribbon (but which might be stored at a different position in q than in q') onto the BB. A mirror argument can be used to prove 3.(a) for ST'.

Finally, for proving point 3.(a) for $q \Longleftrightarrow^2_{\{att\}} q'$, note that the same considerations above apply for the case of a transition from a state q' in which the voting is open. For the case of states q, q' where the voting is closed and hence only strategy nop_{att} is available to the attacker, we note that the only action which is compatible with $q \xrightarrow{nop_{att}} r$ in both models is Environment collecting votes. This corresponds in $\mathcal{G}_{count}$ with an action in which Environment counts votes, and hence we may find a state r which is an nop_{att}-compatible successor of q, which has the same local states for voters as r' and in which each counter co_k of r' keeps the sum of the bullets on kth line on the copied ribbons from r. This will ensure $r \Longleftrightarrow^2_{\{att\}} r'$.

5. EXPERIMENTAL RESULTS

In this section, we exhibit the improvements in running time when checking the same properties over the three bisimilar models. The three models are checked with growing number of voters and candidates. For our experiments, we have used the last version of MCMAS (1.2.2) [30]. Tests were made on a virtual machine running Ubuntu 16.04.1 LTS on a Dell PowerEdge R720 server with two Intel Xeon E5-2650 8 core processors at 2GHz, and 128 GB of RAM. The .ispl files containing the tested models of the voting system are available at [1].

The formulas that are verified on all these models represent a variant of coercion resistance [43]. They specify the fact that the attacker att has no strategy by which he could know how agent i has voted ($i \neq att$):

$$\varphi_i = \langle\!\langle att \rangle\!\rangle F\big(pub \wedge (v_i \rightarrow \bigvee_{1\leqslant j\leqslant nc} K_{att}(j = ch_i))\big)$$

(Recall that, in our model the attacker is also a voter, which corresponds with situations in which a voter fully cooperates with the attacker).

MCMAS provides two options, `-atlk 2` or `-uniform`, for checking ATL formulas with uniform strategies, with some differences in the semantics of ATL formulas (`-uniform` is similar with *"irrevocable strategies"* of [2]). We observed that neither of these options were stable, and lead to a number of experiments ending with inconsistent results or MCMAS terminating abnormally. We refer the interested reader to [10].

We then checked the coercion resistance property with `-atlk 1` option, which utilizes ATL with perfect information. This is nevertheless consistent with our theoretical setting since all tests show that the formulas are false, and whenever a positive ATL formula is false under the perfect information semantics, it is also false under the imperfect information semantics, and hence preserved by alternating bisimulations.

For the *total model* $\mathcal{G}_{tot}$ the only configurations for which MCMAS produces results in reasonable time are shown in Table 1, which gives running times and state space (denoted $|S|$). For $\mathcal{G}_{lex}$, the state space is smaller and, therefore, the model with three voters and three candidates gives a also reasonable running time. For all the other cases, MCMAS outputs the result faster than for $\mathcal{G}_{tot}$. Statistics are given in Table 2. Finally, the models $\mathcal{G}_{count}$ can be verified much faster, the number of reachable states decreasing substantially, allowing for verifying the formula for 4 voters and 3 candidates in 44 seconds. Statistics are given in Table 3.

In all these tables, NA means a 2 hours timeout has been reached without obtaining any result.

# candid.		# voters 2v	3v	4v
	2c	0.93 s |S| = 3.49091e+06	7.765 s |S| = 1.46625e+10	NA
	3c	23.61 s |S| = 2.44048e+08	NA	NA

Table 1: MCMAS statistics for $\mathcal{G}_{tot}$

# candid.		# voters 2v	3v	4v
	2c	0.38 s |S| = 196388	3.42 s |S| = 1.92068e+08	823.12 s |S|= 2.26211e+11
	3c	15.32 s |S| = 8.09895e+06	4807.79 s |S| = 1.03982e+11	NA

Table 2: MCMAS statistics for $\mathcal{G}_{lex}$

# candid.		# voters 2v	3v	4v	5v
	2c	0.15 s |S| = 4406	0.72 s |S| = 39201	2.39 s |S|= 3.08043e+06	17.03 s |S| = 6.57133e+07
	3c	0.44 s |S| = 101993	4.29 s |S| = 3.81446e+06	44.18 s |S| = 2.17425e+09	NA

Table 3: MCMAS statistics for $\mathcal{G}_{count}$

We also verified an anonymity property, specified in CTLK, with the same aim at showing the improvements obtained with bisimulations. Note that, for any group A, an A-bisimulation is also a bisimulation of the epistemic labeled transition systems, hence the two systems satisfy the same CTLK formulas. The CTLK formula that we tested is $\varphi_i^c = AG(not\_same \rightarrow (\bigwedge_{1 \leqslant j \leqslant nc} \neg K_{att} p_{ch_i=j}))$ which utilizes an atomic proposition *not_same* which avoids unanimity. Note that *not_same* can be defined using only the atoms in AP.

6. CONCLUSIONS

In this paper we advanced the state-of-the-art in the model theory of the strategy logic ATL under imperfect information and imperfect recall. Specifically, we introduced a novel notion of (bi)simulation on iCGS that preserves the interpretation of ATL formulas (Theorem 9). Then, we applied this theoretical result to the verification of the ThreeBallot voting system, a relevant voting protocol without cryptography. In particular, we model check the "simpler" bisimilar abstractions of the ThreeBallot system, and then transfer the result to the original model in virtue of Theorem 9. As reported in the experimental results, the gains in terms of both time and memory resources are significant.

The literature on both logics for strategies and the formal verification of voting protocols is extensive and rapidly growing. Hereafter we only consider the works most closely related to the present contribution.

Bisimulations for ATL. An in-depth study of model equivalences induced by various temporal logics appears in [22]. Bisimulations for ATL with perfect information have been introduced in [4]. Since then there have been various attempts to extend these to imperfect information contexts [2, 17]. In [17, 31] non-local model equivalences for ATL with imperfect information have been put forward. However, to our knowledge these works do not deal with the imperfect information/imperfect recall setting here considered, nor do they provide a local account of bisimulations.

# candid.		# voters 2v	3v	4v
	2c	0.776 s |S| = 3.72655e+06	6.531 s |S| = 1.46625e+10	NA
	3c	19.811 s |S| = 2.44048e+08	2628.61 s |S| = 1.69347e+13	NA

Table 4: MCMAS statistics: $\mathcal{G}_{tot}$ and CTLK formula

# candid.		# voters 2v	3v	4v
	2c	0.37 s |S| = 196388	3.035 s |S| = 1.92068e+08	NA
	3c	15.26 s |S| = 8.09895e+06	NA s	NA

Table 5: MCMAS statistics: $\mathcal{G}_{lex}$ and CTLK formula

# candid.		# voters 2v	3v	4v	5v
	2c	0.099 s |S| = 4406	0.553 s |S| = 39201	1.507 s |S|= 3.08043e+06	8.87 s |S| = 6.57133e+07
	3c	0.44 s |S| = 101993	4.29 s |S| = 3.81446e+06	26.078 s |S| = 2.17425e+09	NA

Table 6: MCMAS statistics: $\mathcal{G}_{count}$ and CTLK formula

Verification of Voting Protocols. The present contribution is inspired by recent works on the verification of voting protocols, mostly by using the π-calculus and *CSP* [18, 24, 42]. In [5] the authors define two semantic criteria for single transferable vote (STV) schemes, then show how bounded model-checking and SMT solvers can be used to check whether these criteria are met. In [34] anonymity properties of voting protocols are verified by using CSP. In particular, in [35] the authors construct CSP models of the ThreeBallot system and use them to produce an automated formal analysis of their anonymity properties. One issue we identify with this approach is that the system model and the property to be verified are not clearly distinguished. On the contrary, multi-agent logics allow a clear separation of the two, as well as a wider variety of properties, also involving the existence of attacker strategies. Specifically, in our experiments we are able to model check ThreeBallot systems with 5 voters and 2 candidates, or 4 candidates and 3 voters, while in [35] results are provided for at most 3 voters and 2 candidates.

Future Work. We envisage several extensions of the present contribution. First, it is of interest to develop bisimulations for iCGS with perfect and bounded recall, as in many application domains agents do have some memory of past states and actions. Also for the verification of voting protocols, it is key to extend ATL with epistemic modalities to express naturally properties of anonymity and confidentiality. We remarked that individual knowledge is expressible in the subjective semantics. However, no such result holds for the objective interpretation, nor common knowledge happens to be definable. Finally, we aim at automating and implementing the procedures described in this paper in a model checking tool for the formal verification of (electronic) voting protocols.

Acknowledgements. F. Belardinelli acknowledges the support of the ANR JCJC Project SVeDaS (ANR-16-CE40-0021). W. Jamroga acknowledges the support of the National Centre for Research and Development (NCBR), Poland, under the project VoteVerif (POLLUX-IV/1/2016).

REFERENCES

[1] ISPL Files for ThreeBallot Voting Protocol. https://www.dropbox.com/sh/ferdoqe9hi4cmbx/AAA1hLy0grmCoBVqZf6wbKLWa?dl=0. November 2016.

[2] T. Ågotnes, V. Goranko, and W. Jamroga. Alternating-time temporal logics with irrevocable strategies. In *Proceedings of TARK XI*, pages 15–24, 2007.

[3] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

[4] R. Alur, T. A. Henzinger, O. Kupferman, and M. Y. Vardi. Alternating refinement relations. In *In Proceedings of the Ninth International Conference on Concurrency Theory (CONCUR'98), volume 1466 of LNCS*, pages 163–178. Springer-Verlag, 1998.

[5] B. Beckert, R. Goré, C. Schürmann, T. Bormer, and J. Wang. Verifying voting schemes. *J. Inf. Secur. Appl.*, 19(2):115–129, Apr. 2014.

[6] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*, volume 53 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2001.

[7] I. Boureanu, A. V. Jones, and A. Lomuscio. Automatic Verification of Epistemic Specifications Under Convergent Equational Theories. In *Proceedings of the 11th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'12)*, pages 1141–1148. IFAAMAS, 2012.

[8] N. Bulling, J. Dix, and W. Jamroga. Model checking logics of strategic ability: Complexity. In *Specification and Verification of Multi-agent Systems*, pages 125–159. Springer, 2010.

[9] N. Bulling and W. Jamroga. Comparing variants of strategic ability: how uncertainty and memory influence general properties of games. *Autonomous Agents and Multi-Agent Systems*, 28(3):474–518, 2014.

[10] S. Busard, C. Pecheur, H. Qu, and F. Raimondi. Reasoning about memoryless strategies under partial observability and unconditional fairness constraints. *Information and Computation*, 242:128–156, 2015.

[11] I. Cervesato, N. A. Durgin, P. Lincoln, J. C. Mitchell, and A. Scedrov. A Meta-Notation for Protocol Analysis. In *Proceedings of the 12th IEEE Computer Security Foundations Workshop (CSFW'99)*, pages 55–69. IEEE Computer Society, 1999.

[12] R. Chadha, S. Kremer, and A. Scedrov. Formal analysis of multiparty contract signing. *J. Autom. Reasoning*, 36(1-2):39–83, 2006.

[13] K. Chatterjee, T. Henzinger, and N. Piterman. Strategy logic. In *Proceedings of the 18th International Conference on Concurrency Theory (CONCUR07)*, volume 4703, pages 59–73, 2007.

[14] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement. In *Proceedings of the 12th International Conference on Computer Aided Verification (CAV00)*, volume 1855 of *Lecture Notes in Computer Science*, pages 154–169. Springer, 2000.

[15] E. M. Clarke, O. Grumberg, and D. Long. Model checking and abstractions. *ACM Transactions on Programming Languages and Systems*, 16(5):1512–1542, 1994.

[16] V. Cortier. Formal verification of e-voting: Solutions and challenges. *ACM SIGLOG News*, 2(1):25–34, Jan. 2015.

[17] M. Dastani and W. Jamroga. Reasoning about strategies of multi-agent programs. In *Proceedings of AAMAS2010*, pages 625–632, 2010.

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

[19] G. Denker and J. K. Millen. Modeling Group Communication Protocols Using Multiset Term Rewriting. *Electr. Notes Theor. Comput. Sci.*, 71:20–39, 2002.

[20] C. Dima and F. L. Tiplea. Model-checking ATL under imperfect information and perfect recall semantics is undecidable. *CoRR*, abs/1102.4225, 2011.

[21] P. Gammie and R. van der Meyden. MCK: Model checking the logic of knowledge. In *Proceedings of 16th International Conference on Computer Aided Verification (CAV04)*, volume 3114 of *Lecture Notes in Computer Science*, pages 479–483. Springer, 2004.

[22] U. Goltz, R. Kuiper, and W. Penczek. Propositional temporal logics and equivalences. In *Proceedings of CONCUR '92*, pages 222–236, 1992.

[23] V. Goranko and W. Jamroga. Comparing semantics for logics of multi-agent systems. *Synthese*, 139(2):241–280, 2004.

[24] C. A. R. Hoare. Communicating Sequential Processes. *Commun. ACM*, 21(8):666–677, 1978.

[25] W. Jamroga and J. Dix. Model checking abilities under incomplete information is indeed δ_p^2-complete. In *Proceedings of the 4th European Workshop on Multi-Agent Systems EUMAS'06*, pages 14–15. Citeseer, 2006.

[26] W. Jamroga and W. van der Hoek. Agents that know how to play. *Fundamenta Informaticae*, 62:1–35, 2004.

[27] M. Kacprzak, W. Nabialek, A. Niewiadomski, W. Penczek, A. Pólrola, M. Szreter, B. Woźna, and A. Zbrzezny. Verics 2007 - a model checker for knowledge and real-time. *Fundamenta Informaticae*, 85(1):313–328, 2008.

[28] R. Küsters, T. Truderung, and A. Vogt. Verifiability, privacy, and coercion-resistance: New insights from a case study. In *32nd IEEE Symposium on Security and Privacy, S&P 2011, 22-25 May 2011, Berkeley, California, USA*, pages 538–553. IEEE Computer Society, 2011.

[29] F. Laroussinie, N. Markey, and G. Oreiby. On the expressiveness and complexity of ATL. *Logical Methods in Computer Science*, 4(2:7), May 2008.

[30] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS: A model checker for the verification of multi-agent systems. *Software Tools for Technology Transfer*, 2015. http://dx.doi.org/10.1007/s10009-015-0378-x.

[31] M. Melissen. *Game Theory and Logic for Non-repudiation Protocols and Attack Analysis*. PhD thesis, University of Luxembourg, 2013.

[32] F. Mogavero, A. Murano, G. Perelli, and M. Y. Vardi. Reasoning about strategies: On the model-checking

problem. *ACM Transactions in Computational Logic*, 15(4):34:1–34:47, 2014.

[33] F. Mogavero, A. Murano, and M. Vardi. Reasoning About Strategies. In *Proceedings of the 30th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS10)*, volume 8, pages 133–144. Schloss Dagstuhl, 2010.

[34] M. Moran, J. Heather, and S. Schneider. Verifying anonymity in voting systems using csp. *Formal Aspects of Computing*, 26(1):63–98, 2014.

[35] M. Moran, J. Heather, and S. Schneider. Automated anonymity verification of the ThreeBallot and VAV voting systems. *Software & Systems Modeling*, 15(4):1049–1062, 2016.

[36] M. R. Neuhäußer and J. Katoen. Bisimulation and logical preservation for continuous-time Markov Decision Processes. In *CONCUR 2007 - Concurrency Theory, 18th International Conference, CONCUR 2007, Lisbon, Portugal, September 3-8, 2007, Proceedings*, volume 4703 of *Lecture Notes in Computer Science*, pages 412–427. Springer, 2007.

[37] M. Pauly. A modal logic for coalitional power in games. *Journal of Logic and Computation*, 12(1):149–166, 2002.

[38] W. S. R. Rivest. Three voting protocols: ThreeBallot, VAV, and Twin. In *Proceedings of USENIX/ACCURATE Electronic Voting Technology Workshop (EVT)*, 2007.

[39] R. L. Rivest. The ThreeBallot Voting System, October 2006. `http://theory.lcs.mit.edu/~rivest/Rivest-TheThreeBallotVotingSystem.pdf`.

[40] P. A. Ryan. The computer ate my vote. In P. Boca, J. P. Bowen, and J. Siddiqi, editors, *Formal Methods: State of the Art and New Directions*, chapter 5, pages 148–184. Springer Verlag, 2009.

[41] P. Y. A. Ryan, S. A. Schneider, and V. Teague. End-to-end verifiability in voting systems, from theory to practice. *IEEE Security & Privacy*, 13(3):59–62, 2015.

[42] S. Schneider and A. Sidiropoulos. CSP and Anonymity. In *Proceedings of the 1996 European Symposium on Research in Computer Security (ESORICS'96)*, volume 1146 of *Lecture Notes in Computer Science*, pages 198–218. Springer-Verlag, 1996.

[43] M. Tabatabaei, W. Jamroga, and P. Ryan. Expressing receipt-freeness and coercion-resistance in logics of strategic ability: Preliminary attempt. In *Proceedings of the 1st International Workshop on AI for Privacy and Security PrAISe 2016*, pages 1:1–1:8. ACM, 2016.

[44] J. van Eijck and S. Orzan. Epistemic verification of anonymity. *Electr. Notes Theor. Comput. Sci.*, 168:159–174, 2007.