Verification of Non-Uniform and Unbounded Artifact-Centric Systems: Decidability through Abstraction

Francesco Belardinelli
Laboratoire IBISC, Université d'Evry
belardinelli@ibisc.fr

ABSTRACT

The formal verification of Artifact-centric (AC) systems is a subject of growing interest in the Service Oriented Computing (SOC) community, which can benefit from techniques developed for Multi-agent systems and knowledge reasoning and representation. In the present contribution we consider the verification of AC systems that do not necessarily satisfy boundedness and uniformity, the typical assumptions used to prove decidability of the model checking problem in this setting. We provide a partial model checking procedure for agent-based AC systems against a first-order temporal logic that includes modal operators for agent knowledge. Interestingly, we obtain this result by introducing a counterpart semantics for first-order modal logic, and by defining notions of simulation and abstraction for this setting. This allows us to generate finite abstractions of infinite-state AC systems, even when these are not bounded nor uniform, thus enabling us to perform verification also in cases not covered by the current state-of-the-art.

Categories and Subject Descriptors

F.4 [**Theory of computation**]: Modal and temporal logics

General Terms

Languages, Theory, Verification.

Keywords

Temporal Epistemic Logic, Verification of Agent-based Systems.

1. INTRODUCTION

Artifact-centric (AC) systems have been recently put forward as a framework for the design, implementation and integration of business processes in Service Oriented Computing (SOC) [20, 21]. In the artifact paradigm the *service data model* and the *business processes* are seen as equally important components of the interface specification. This is in marked contrast with most of the traditional approaches to web-service architectures, which usually abstract data away to reduce the complexity of the system description [24]. In fact, artifacts allow for explicit dependence of state transitions on information contained in the data model. However, this enhanced expressiveness comes at a price. The presence of data means that the typical questions pertaining to system verification are much more difficult to answer. Indeed, the manipulation of data structures entails a possibly infinite state-space, which cannot be immediately handled by standard verification techniques.

In this paper we advance the state-of-the-art on artifact verification by developing a methodology to model check a class of AC systems that includes agents to account for the services operating on artifacts. Model checking is a success story on the application of formal methods in computer science [11, 2]. It has allowed for the rigorous verification of complex systems against rich specifications [10, 22, 23]. Hence, it is only natural that this technique has already been applied to the formal verification of AC systems. Since the model checking problem for AC systems in their most general setting is undecidable, the contributions in this area have focused on finding syntactic and semantics restrictions on either the system specifications or the class of relevant models, in order to ensure a decidable model checking problem, while still admitting most scenarios of interest [6, 19, 18]. Recently, a condition known as *uniformity* has been proved sufficient to ensure decidability of model checking when combined with a *boundedness* assumption on the number of active elements in each state [5, 6]. Also, uniformity was shown to be satisfied by a number of AC frameworks appearing in the literature [19, 6], thus highlighting the relevance of this condition. Therefore, a natural question is whether uniformity and boundedness are really necessary to model check AC systems. For practical purposes, the latter condition is usually difficult to check, thus more general techniques would be welcome.

In addition, most contributions on the verification of AC systems disregards the services operating on artifacts, hence modelling the system evolution in a monolithic manner. Hereafter we depart from this approach and formalize AC systems within an agent-based framework that accounts for processes operating on artifacts. We do so by relying on results on Multi-agent system (MAS) and knowledge reasoning and representation. Specifically, we model services as autonomous and proactive agents [26], and consider a specification language containing epistemic operators to express agent knowledge [17].

The main contribution of this paper can then be summarized as follows. We first introduce a formalisation of AC systems as a particular type of artifact-centric multi-agent systems (AC-MAS) [5, 6]. The latter can intuitively be seen as databases that evolve in time and are manipulated by agents. Differently from the references above, the AC-MAS presented hereafter are not necessarily uniform. This feature makes the verification techniques appearing in the state-of-the-art not applicable to the present framework. We state the model checking problem with respect to a first-order temporal epistemic logic including function symbols. We remark that the presence of functions increases the expressive power of our

Appears in: *Alessio Lomuscio, Paul Scerri, Ana Bazzan, and Michael Huhns (eds.), Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), May 5-9, 2014, Paris, France.*

specification language. However, it has not previously considered in the literature to our knowledge, as AC systems on such a language are not uniform in general. Most importantly, we develop a novel verification methodology for non-uniform and unbounded AC-MAS inspired to [3], and introduce a sound, albeit incomplete, model checking procedure with respect to formulas in the universal fragment of our first-order temporal epistemic logic. We deem the proposed technique of interest for the model theory of first-order modal logic (FOML) as well. Indeed, to obtain finite abstractions of infinite AC systems we make use of the counterpart semantics for FOML [9, 12], we define a notion of simulation for counterpart models, and then introduce quotient structures and similar finite abstractions. Thus, we believe that our contribution has a theoretical interest for modal logicians beyond the verification of AC systems.

Scheme of the paper. In Section **??** we fix the notation, give the syntax of the first-order temporal epistemic logic FO-CTLK, as well as an agent-based setting for AC systems. A counterpart semantics for AC systems and a notion of simulation are presented in Section 3. Section 4 contains the main result of the paper, i.e., the definition of finite abstractions for non-uniform and unbounded AC-MAS. Section **??** explores constructive methods to derive abstract AC-MAS. We conclude in Section 5 by discussing the results obtained and point to future work. For reasons of space and sake of presentation all proofs are omitted.

2. PRELIMINARIES

In this section we present the artifact-centric multi-agent systems (AC-MAS) as a multi-agent setting for AC systems [5, 6], we introduce a first-order version of the temporal epistemic logic CTLK, including function symbols for individuals, and state the corresponding model checking problem. We first present the basic terminology on databases that is used throughout the paper [1].

DEFINITION 1 (DATABASE SCHEMA AND INSTANCE). *A* database schema *is a finite set* $\mathcal{D} = \{P_1/q_1, \ldots, P_n/q_n\}$ *of* predicate symbols P_i *with arity* $q_i \in \mathbb{N}$.

Given a (possibly infinite) interpretation domain U*, a* $\mathcal{D}$-instance *over* U *is a mapping* D *associating each predicate symbol* P_i *to a* finite q_i*-ary relation on* U*, i.e.,* $D(P_i) \subseteq U^{q_i}$.

The set $\mathcal{D}(U)$ contains all the $\mathcal{D}$-instances on the domain U. The *active domain* $adom(D)$ of a $\mathcal{D}$-instance D is the *finite* set of all individuals occurring in some predicate interpretation $D(P_i)$. Also, the *primed version* of a database schema $\mathcal{D}$ as above is the schema $\mathcal{D}' = \{P_1'/q_1, \ldots, P_n'/q_n\}$. Then, the disjoint union $D \oplus D'$ of $\mathcal{D}$-instances D and D' is the $(\mathcal{D} \cup \mathcal{D}')$-instance s.t. *(i)* $D \oplus D'(P_i) = D(P_i)$, and *(ii)* $D \oplus D'(P_i') = D'(P_i)$.

We now introduce the notion of *service agent*, i.e., an agent operating a service in an artifact-centric system.

DEFINITION 2 (SERVICE AGENT). *Given an interpretation domain* U*, a* service agent *is a tuple* $A = \langle \mathcal{D}, Act, Pr \rangle$*, where*

- $\mathcal{D}$ *is the* local database schema*;*
- Act *is the finite set of* action types $\alpha(\vec{p})$*, where* $\vec{p}$ *is a tuple of* abstract parameters*;*
- $Pr : \mathcal{D}(U) \mapsto 2^{Act(U)}$ *is the* local protocol function*, where* $Act(U)$ *is the set of* ground actions $\alpha(\vec{u})$*, for* $\alpha(\vec{p}) \in Act$*, and* $\vec{u} \in U^{|\vec{p}|}$ *a tuple of* ground parameters.

Intuitively, in each moment the service agent A is in some local state $l \in \mathcal{D}(U)$ that represents all the information she has about the system. In this respect we follow the typical approach to MAS [17, 25], but here we require that this information is structured as a database. Also, as standard we assume that agents are autonomous and proactive, and perform the actions in Act according to the protocol function Pr. As we are interested in the interactions of service agents among themselves and with the external environment, we define their synchronous composition.

DEFINITION 3 (AC-MAS). *Given an interpretation domain* U *and a set* $Ag = \{A_0, \ldots, A_n\}$ *of service agents* $A_i = \langle \mathcal{D}_i, Act_i, Pr_i \rangle$ *defined on* U*, an* artifact-centric multi-agent system *is a tuple* $\mathcal{P} = \langle Ag, s_0, \tau \rangle$ *where*

- $s_0 \in \mathcal{D}_0(U) \times \ldots \times \mathcal{D}_n(U)$ *is the* initial global state*;*
- $\tau : \mathcal{D}_0(U) \times \ldots \times \mathcal{D}_n(U) \times Act(U) \mapsto 2^{\mathcal{D}_0(U) \times \ldots \times \mathcal{D}_n(U)}$ *is the* global transition function*, where* $Act(U) = Act_0(U) \times \cdots \ldots \times Act_n(U)$ *is the set of* global (ground) actions*, and* $\tau(\langle l_0, \ldots, l_n \rangle, \langle \alpha_0(\vec{u}_0), \ldots, \alpha_n(\vec{u}_n) \rangle)$ *is defined iff* $\alpha_i(\vec{u}_i) \in Pr_i(l_i)$ *for every* $i \leq n$.

An AC-MAS evolves from the initial state s_0 according to the global transition function τ, which returns a set of successor states for each possible joint action of service agents. We express the dependency of transitions on data by instantiating the formal parameters of actions with different ground individuals. Since the domain of interpretation U is infinite in general, this means that AC-MAS are infinite-state systems. In this respect, AC-MAS can be seen as a natural extension of interpreted systems [17] to the first order.

The framework of AC-MAS is rich enough to formalise AC systems as presented, for instance, in [21, 14]. Indeed, the data model of AC systems can be translated into the database schema of AC-MAS, while its lifecycle can be modelled by the transition function τ. Some of the more complex features of AC systems, such as tasks and messages, are abstracted in AC-MAS by the use of service agents. Nonetheless, in [4, 6] it is shown that AC-MAS are adequate to represent the Guard-Stage-Milestone (GSM) models for AC systems, as well as their small-step semantics.

Besides the formalisation of AC systems, we believe that AC-MAS have a more general, theoretical interest, as these represent relational models where each state is a finite first-order structure. In this respect, results available for AC-MAS transfer to first-order modal logic not dissimilarly from the way theorems on finite model theory find applications in database theory. Thus, a study of AC-MAS might benefit the model theory of FOML.

We now introduce some technical notions that will be used in the rest of the paper. We denote a global ground action as $\alpha(\vec{u})$, where $\alpha = \langle \alpha_0(p_0), \ldots, \alpha_n(p_n) \rangle$ and $\vec{u} = \langle \vec{u}_0, \ldots, \vec{u}_n \rangle$, and define the *transition relation* $\rightarrow$ on global states such that $s \rightarrow s'$ if there exists $\alpha(\vec{u}) \in Act(U)$ such that $s \xrightarrow{\alpha(\vec{u})} s'$, i.e., $s' \in \tau(s, \alpha(\vec{u}))$. A *run* r from a state s is an infinite sequence $s^0 \rightarrow s^1 \rightarrow \cdots$, with $s^0 = s$. For $n \in \mathbb{N}$, we define $r(n) = s^n$. A state s' is *reachable from* s if there exists a run r from $r(0) = s$ such that $r(i) = s'$ for some $i \geq 0$. Hereafter we assume that the relation $\rightarrow$ is serial. This can be ensured by using skip actions. Further, we define $\mathcal{S}$ as the set of states reachable from the initial state s_0. As in propositional interpreted systems [17], two global states $s = \langle l_0, \ldots, l_n \rangle$ and $s' = \langle l_0', \ldots, l_n' \rangle$ are *epistemically indistinguishable* for service agent A_i, written $s \sim_i s'$, if $l_i = l_i'$. Differently from propositional interpreted systems, the local equality is evaluated on database instances. Also, since we allow U to be infinite, the set $\mathcal{S}$ of reachable states is also infinite in principle. Indeed, in the general case our AC-MAS are infinite-state systems. Finally, for technical reasons we will refer to the *global* database schema $\mathcal{D} = \mathcal{D}_0 \cup \cdots \cup \mathcal{D}_n$ of an AC-MAS. Hence, every global state $s = \langle l_0, \ldots, l_n \rangle$ is associated with the $\mathcal{D}$-instance $D_s \in \mathcal{D}(U)$ such that $D_s(P_i) = \bigcup_{j \in Ag} l_j(P_i)$, for $P_i \in \mathcal{D}$, that is, we assume that each service agent has a truthful, yet limited, view of the

global database $\mathcal{D}$. This modelling choice is best suited to represent AC systems. We omit the subscript s whenever s is clear from the context and write $adom(s)$ for $adom(D_s)$. Notice that for every $s \in \mathcal{S}$, there is a unique D_s, while the converse is not true in general. Moreover, $s \oplus s'$ is tantamount to $\langle l_0 \oplus l'_0, \ldots, l_n \oplus l'_n \rangle$.

We now introduce the specification language for AC-MAS. We consider a set *Var* of *individual variables*, a set $Con \subseteq U$ of *individual constants*, and a set F of functions on U. The terms $t_1, t_2, \ldots$ in our language are inductively defined as either variables in *Var*, or constants in *Con*, or elements $f^k(t_1, \ldots, t_k)$, where f^k is a k-ary function in F and $t_1, \ldots, t_k$ are terms.

DEFINITION 4 (FO-CTLK). *The FO-CTLK formulas φ over a database schema $\mathcal{D}$ are defined as follows:*

$$\varphi ::= P_i(\vec{t}) \mid \neg\varphi \mid \varphi \rightarrow \varphi \mid \forall x\varphi \mid AX\varphi \mid A\varphi U\varphi \mid E\varphi U\varphi \mid K_j\varphi \mid C\varphi$$

where $P_i \in \mathcal{D}$, $j \in Ag$, and $\vec{t}$ is a q_i-tuple of terms*.*

The language FO-CTLK is a first-order extension of the propositional temporal epistemic logic CTLK including function symbols. While the use of function symbols is non-standard in database theory, these are crucial to describe the behaviour of AC-MAS, specifically w.r.t. the manipulation of data by service agents.

The temporal formulas $AX\varphi$ and $A\varphi U\varphi'$ (resp. $E\varphi U\varphi'$) are read as "for all runs, at the next step φ" and "for all runs (resp. some run), φ until φ'". The epistemic formulas $K_i\varphi$ and $C\varphi$ intuitively mean that "service agent A_i knows φ" and "it is common knowledge that φ" respectively. Free and bound variables are defined as standard, as well as the formulas $EX\varphi$, $AF\varphi$, $AG\varphi$, $EF\varphi$, and $EG\varphi$. We write $\phi(\vec{x})$ (resp. $t(\vec{x})$) to denote that the free variables of ϕ (resp. t) are among $x_1, \ldots, x_n$. The sublanguage FO-ACTLK, i.e., the restriction of FO-CTLK to the universal modalities AX and AU, is defined as follows:

$$\varphi ::= P_i(\vec{t}) \mid \neg P_i(\vec{t}) \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \forall x\varphi \mid \exists x\varphi \mid AX\varphi \mid A\varphi U\varphi \mid K_j\varphi \mid C\varphi$$

In what follows we conside also the non-modal fragment of FO-CTLK, i.e., the first-order logic FO on $\mathcal{D}$, and its extension $\text{FO}^{=}$ obtained by adding identity between terms.

An *assignment* is a function $\sigma : Var \mapsto U$. We denote by σ^x_u the assignment s.t. *(i)* $\sigma^x_u(x) = u$; and *(ii)* $\sigma^x_u(x') = \sigma(x')$ for $x' \neq x$. Also, we assume a Herbrand interpretation of constants and functions, that is, $\sigma(c) = c$ for all $c \in Con$, and $\sigma(f^k(t_1, \ldots, t_k)) = f^k(\sigma(t_1), \ldots, \sigma(t_k))$.

DEFINITION 5 (SEMANTICS OF FO-CTLK). *We define whether an AC-MAS $\mathcal{P}$* satisfies *a formula φ in a state s according to assignment σ, or $(\mathcal{P}, s, \sigma) \models \varphi$, as follows (clauses for propositional connectives are trivial and thus omitted):*

$(\mathcal{P}, s, \sigma) \models P_i(\vec{t})$ *iff* $\langle \sigma(t_1), \ldots, \sigma(t_{q_i}) \rangle \in D_s(P_i)$
$(\mathcal{P}, s, \sigma) \models \forall x\varphi$ *iff for all $u \in adom(s)$, $(\mathcal{P}, s, \sigma^x_u) \models \varphi$*
$(\mathcal{P}, s, \sigma) \models AX\varphi$ *iff for all r, if $r(0) = s$ then $(\mathcal{P}, r(1), \sigma) \models \varphi$*
$(\mathcal{P}, s, \sigma) \models A\varphi\mathcal{U}\varphi'$ *iff for all r, if $r(0) = s$ then there is $k \geq 0$ s.t. $(\mathcal{P}, r(k), \sigma) \models \varphi'$, and for all j, $0 \leq j < k$ implies $(\mathcal{P}, r(j), \sigma) \models \varphi$*
$(\mathcal{P}, s, \sigma) \models E\varphi\mathcal{U}\varphi'$ *iff for some r, $r(0) = s$ and there is $k \geq 0$ s.t. $(\mathcal{P}, r(k), \sigma) \models \varphi'$, and for all j, $0 \leq j < k$ implies $(\mathcal{P}, r(j), \sigma) \models \varphi$*
$(\mathcal{P}, s, \sigma) \models K_i\varphi$ *iff for all s', $s \sim_i s'$ implies $(\mathcal{P}, s', \sigma) \models \varphi$*
$(\mathcal{P}, s, \sigma) \models C\varphi$ *iff for all s', $s \sim s'$ implies $(\mathcal{P}, s', \sigma) \models \varphi$*

where $\sim$ is the transitive closure of $\bigcup_{A_i \in Ag} \sim_i$.

A formula φ is *true* at s, written $(\mathcal{P}, s) \models \varphi$, if $(\mathcal{P}, s, \sigma) \models \varphi$ for all σ; φ is *true* in $\mathcal{P}$, written $\mathcal{P} \models \varphi$, if $(\mathcal{P}, s_0) \models \varphi$.

Notice that we adopt an *active domain* semantics, where quantifiers range over the active domain $adom(s)$ of s. This is a standard assumption in database theory.

Finally, we present the model checking problem for AC-MAS with respect to the specification language FO-CTLK.

DEFINITION 6 (MODEL CHECKING PROBLEM). *Given an AC-MAS $\mathcal{P}$ and an FO-CTLK formula φ, determine whether there is an assignment σ_0 such that $(\mathcal{P}, s_0, \sigma_0) \models \varphi$.*

Model checking general AC-MAS is known to be undecidable. In [5] this problem is proved to be decidable for *bounded* and *uniform* systems. We now introduce both notions in relation to AC-MAS on a language with only variables and constants as individual terms, which is the original setting of [5]. We first define a notion of isomorphism between states: two states s and s' are *isomorphic*, or $s \simeq s'$, iff there exists a bijection $\iota : adom(s) \cup Con \mapsto adom(s') \cup Con$ s.t. *(i)* ι is the identity on *Con*; and *(ii)* for every $i \in Ag$, $P_j \in \mathcal{D}$, $\vec{u} \in U^{q_j}$, $\vec{u} \in l_i(P_j)$ iff $\iota(\vec{u}) \in l'_i(P_j)$.

DEFINITION 7 (BOUNDEDNESS AND UNIFORMITY). *An AC-MAS $\mathcal{P}$ is* bounded *iff there exists $b \in \mathbb{N}$ such that for all $s \in \mathcal{S}$, $|adom(s)| \leq b$.*

An AC-MAS $\mathcal{P}$ is uniform *iff for $s, t, s' \in \mathcal{S}$, $t' \in \mathcal{D}(U)$, if $s \xrightarrow{\alpha(\vec{u})} t$ and $s \oplus t \simeq s' \oplus t'$ for some bijection ι, then $s' \xrightarrow{\alpha(\iota'(\vec{u}))} t'$ for every constant-preserving bijection ι' extending ι to $\vec{u}$.*

Notice that the boundedness condition restricts the number of elements appearing in the active domain of each state, not the total number of states in $\mathcal{S}$, which is infinite in general. In [5] the combination of both features is proved sufficient to obtain a decidable model checking problem. Yet, in [3] boundedness is shown not strong enough to ensure decidability alone. To conclude, we illustrate the formal machinery introduced thus far with a running example.

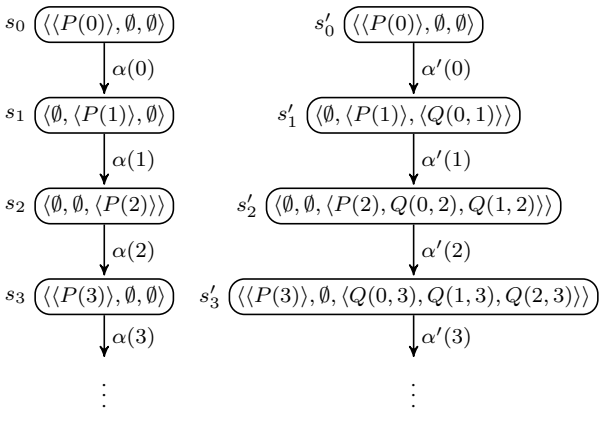
EXAMPLE 1. While the following example is relatively small and designed on purpose, it is nonetheless instructive to describe the relevant features of AC-MAS introduced above. It elaborates on a similar example appeared in [3]. Assume $\mathbb{N}$ as the interpretation domain and consider a set $Ag = \{A_0, A_1, A_2\}$ of service agents $A_i = \langle \mathcal{D}_i, Act_i, Pr_i \rangle$ defined as follows: for $i \leq 2$, *(i)* $\mathcal{D}_i = \{P/1\}$; *(ii)* $Act_i = \{\alpha_i(n)\}$; and *(iii)* for every $l_i \in \mathcal{D}_i(\mathbb{N})$, $Pr_i(l_i) = \{\alpha_i(n)\}$. Now let $\mathcal{P}_1 = \langle Ag, s_0, \tau \rangle$ be the AC-MAS depicted in Fig. 1(a), where

- the initial state s_0 is equal to $\langle \langle P(0) \rangle, \emptyset, \emptyset \rangle$;
- $s' \in \tau(s, \alpha(n))$ whenever exactly one of the l_i contains $P(n)$, while $l'_{(i+1)\%3}$ contains exactly $P(n+1)$. Any other local state is empty.

Notice that for every state s' and $n \in \mathbb{N}$ there exists at most one s' such that $s \xrightarrow{\alpha(n)} s'$. Moreover, $\mathcal{P}_1$ is non-uniform. Indeed, $s_0 \oplus s_1 \simeq s_0 \oplus s_4$ with bijection $\iota(0) = 0$ and $\iota(1) = 4$. Also, $s_0 \xrightarrow{\alpha(0)} s_1$, but it is not the case that $s_0 \xrightarrow{\alpha(0)} s_4$.

Furthermore, as an example of an unbounded AC-MAS we consider the following modification of $\mathcal{P}_1$. Define $A'_2 = \langle \mathcal{D}'_2, Act'_2, Pr'_2 \rangle$ s.t. *(i)* $\mathcal{D}'_2 = \{P/1, Q/2\}$; *(ii)* $Act_2 = \{\alpha'_2(n)\}$; and *(iii)* for every $l_2 \in \mathcal{D}_2(\mathbb{N})$, $Pr'_2(l_2) = \{\alpha'_2(n)\}$. For $Ag' = \{A_0, A_1, A'_2\}$ let $\mathcal{P}_2 = \langle Ag', s'_0, \tau' \rangle$ be the AC-MAS depicted in Fig. 1(b), where

- the initial state s'_0 is equal to $s_0 = \langle \langle P(0) \rangle, \emptyset, \emptyset \rangle$;
- $s' \in \tau'(s, \alpha(n))$ whenever exactly one of the l_i contains $P(n)$, while $l'_{(i+1)\%3}$ contains exactly $P(n+1)$. Also, l'_2 contains $Q(k, n+1)$ for all $k < n+1$. Any other local state is empty.



(a) the AC-MAS $\mathcal{P}_1$

(b) the AC-MAS $\mathcal{P}_2$

Figure 1: the AC-MAS $\mathcal{P}_1$ and $\mathcal{P}_2$ (epistemic components are omitted)

Clearly, $\mathcal{P}_2$ is unbounded. Now consider also the following specifications in FO-ACTLK:

$$\chi = AG\,\forall x K_1(P(x) \rightarrow AX\neg P(x))$$
$$\theta = AG\,\forall x K_2(P(x) \rightarrow AXAG\neg P(x))$$

As an example, θ states that service agent 2 always knows that if something is P, then it will never be P again. Since $\mathcal{P}_1$ and $\mathcal{P}_2$ are infinite-state systems, χ and θ cannot be model checked directly. Moreover, we remarked that $\mathcal{P}_1$ is not uniform, while $\mathcal{P}_2$ is unbounded. Thus, the techniques developed in [5, 19] cannot be of help in the present context. ■

Hereafter we explore the boundary between decidability and undecidability of model checking general non-uniform and unbounded AC-MAS. These investigations require the counterpart semantics for first-order modal logic.

3. COUNTERPART SEMANTICS

In this section we introduce a semantics for FO-CTLK based on counterparts [9]. Then, we recast the notion of simulation for counterpart models in [3] to a multi-agent system setting. Finally, we prove that the simulation relation preserves FO-ACTLK, the universal fragment of FO-CTLK.

DEFINITION 8 (MA C-MODEL). *Given a set Ag of service agents, a* multi-agent counterpart model *is a tuple $\mathcal{M} = \langle S, s_0, U, \rightarrow, C^t, \{\sim_i\}_{i\in Ag}, \{C^i\}_{i\in Ag}, I\rangle$ such that* (i) *S is a non-empty* set of states*;* (ii) *s_0 is the* initial state*;* (iii) *$\rightarrow$ is a serial binary* transition relation *on S;* (iv) *each $\sim_i$ is an* equivalence relation *on S; and* (iv) *for $s, s' \in S$,*

- *$U(s)$ is a non-empty* set of individuals*;*
- *$C^t_{s,s'} \subseteq U(s) \times U(s')$ is a serial* temporal counterpart relation*;*
- *each $C^i_{s,s'}$ is a relation on $U(s) \times U(s')$ s.t.* (i) *$C^i_{s,s} \subseteq U(s)^2$,* (ii) *$C^i_{s,s'} = \overline{C}^i_{s',s}$, and* (iii) *$C^i_{s,s'} \cdot C^i_{s',s''} \subseteq C^i_{s,s''}$ (where $\overline{R}$ is the converse of R and $\cdot$ is relation composition);*
- *I is a first-order interpretation, i.e.,* (i) *if P^n is an n-ary predicate symbol and $s \in S$, then $I(P^n, s)$ is an n-ary relation on $U(s)$;* (ii) *if $c \in Con$, then $I(c, s) \in U(s)$; and* (iii) *if $f^k \in F$, then $I(f^k, s)$ is a function from $U(s)^k$ to $U(s)$.*

Counterpart semantics was originally conceived as a semantics for first-order modal logic in which we do not need to assume that the same individual appears in more than one system state. In fact, the task of identifying "related" individuals across states is performed by the counterpart relations.

Similarly to AC-MAS, the active domain *adom*(s) of a state s is defined as the set of all individuals occurring in some predicate interpretation $I(P^n, s)$. We remark without proof that any AC-MAS can be seen as an MA c-model on a finite language, where the counterpart relation is the identity, and constants and functions are interpreted as themselves.

REMARK 1. *Let $\mathcal{P} = \langle Ag, s_0, \tau\rangle$ be an AC-MAS on the domain U. Consider $\mathcal{M}_\mathcal{P} = \langle \mathcal{S}, s_0, U', \rightarrow, C^t, \{\sim_i\}_{i\in Ag}, \{C^i\}_{i\in Ag}, I\rangle$, where* (i) *$\mathcal{S}$ is the set of reachable states in $\mathcal{P}$,* (ii) *s_0 is the initial state in $\mathcal{P}$,* (iii) *$\rightarrow$ and each $\sim_i$ are defined as in Section* **??***; and for all $s, s' \in \mathcal{S}$,* (iv) *$U'(s) = U$;* (v) *$C^t_{s,s'}$ and each $C^i_{s,s'}$ are the identity relation;* (vi) *$I(c, s) = c$, $I(f^k, s) = f^k$, and $I(P^n, s) = D_s(P^n)$. Then $\mathcal{M}_\mathcal{P}$ is an MA c-model.*

The notion of *run* is defined as for AC-MAS. The relation C^{t+} is the transitive closure of C^t, i.e., $C^{t+}_{s,s'}(a, a')$ iff there is a sequence $s^0 \rightarrow \ldots \rightarrow s^k$ s.t. $s^0 = s$, $s^k = s'$, and there are $a^0, \ldots, a^k$ s.t. $a^0 = a$, $a^k = a'$, and $C^t_{s^i,s^{i+1}}(a^i, a^{i+1})$ for $i < k$.

As in Section **??**, $\sim$ is the transitive closure of $\bigcup_{A_i\in Ag} \sim_i$, while for $s, s' \in S$ s.t. $s \sim s'$, $C_{s,s'}(a, a')$ iff there is a sequence $s^0 \sim_{i_1} \ldots \sim_{i_k} s^k$ s.t. $s^0 = s$, $s^k = s'$, and there are $a^0, \ldots, a^k$ s.t. $a^0 = a$, $a^k = a'$, and $C^{i_{h+1}}_{s^h,s^{h+1}}(a^h, a^{h+1})$ for $h < k$.

To define the satisfaction of FO-CTLK formulas in MA c-models, we consider typed languages and finitary assignments. This is standard when working with counterpart semantics [9, 12]. Specifically, every variable $x_j \in Var$ is a term of type n, or n-term, for $n \geq j$; every constant $c \in Con$ is an n-term; and if f^k is a function symbol and $\vec{t}$ is a k-tuple of n-terms, then $f^k(\vec{t})$ is an n-term.

DEFINITION 9 (FO-CTLK$_T$). *The typed language FO-CTLK$_T$ contains all n-formulas $\phi : n$, for $n \in \mathbb{N}$, defined as follows:*

- *if P^m is an m-ary predicate symbol and $\vec{t}$ is an m-tuple of n-terms, then $P^m(\vec{t})$ is an (atomic) n-formula;*
- *if ψ, ψ' are n-formulas, then $\neg\psi$, $\psi \rightarrow \psi'$, $AX\psi$, $A\psi U\psi'$, $E\psi U\psi'$, $K_i\psi$ and $C\psi$ are n-formulas;*
- *if ψ is an $(n+1)$-formula, then $\forall x_{n+1}\psi$ is an n-formula.*

The other logical operators are defined as standard. In what follows we consider also the sublanguage FO-ACTLK$_T$, which is standardly obtained by restricting FO-CTLK$_T$ to the universal modalities AX, AU, K_i and C. Also, the typed first-order logic FO$_T$ is the non-modal fragment of FO-CTLK$_T$.

The meaning of a typed formula $\phi : n$ at a state s can intuitively be understood as a subset of $U(s)^n$, i.e., the set of n-tuples satisfying $\phi : n$ at s. Therefore, the definition of satisfaction is given by means of finitary assignments, where an n-assignment in s is an n-tuple $\vec{a}$ of elements in $U(s)$. Let t be an n-term, the valuation $\vec{a}(t)$ for the n-assignment $\vec{a}$ is equal to a_j if $t = x_j$. Also, $\vec{a}(t) = I(c, s)$ whenever $t = c$, and $\vec{a}(f^k(\vec{t})) = I(f^k, s)(\vec{a}(t_1), \ldots, \vec{a}(t_k))$.

DEFINITION 10 (SEMANTICS OF FO-CTLK$_T$). *The satisfaction relation $\models$ for a state $s \in \mathcal{M}$, a typed formula $\phi : n$ and an n-assignment $\vec{a}$ is inductively defined as follows (clauses for propositional connectives are trivial and thus omitted):*

$(\mathcal{M}, s, \vec{a}) \models P^m(\vec{t})$ *iff* $\langle \vec{a}(t_1), \ldots, \vec{a}(t_m)\rangle \in I(P^m, s)$
$(\mathcal{M}, s, \vec{a}) \models AX\psi$ *iff for all r, $\vec{b} \in U(r(1))$, if $r(0) = s$ and*

$C^t_{s,r(1)}(\vec{a},\vec{b})$ *then* $(\mathcal{M}, r(1), \vec{b}) \models \psi$

$(\mathcal{M}, s, \vec{a}) \models A\varphi U\varphi'$ *iff for all* r, *if* $r(0) = s$ *then there are* $k \geq 0$, $\vec{b} \in U(r(k))$ *s.t.* $C^+_{s,r(k)}(\vec{a},\vec{b})$ *and* $(\mathcal{M}^{\vec{b}}, r(k)) \models \varphi'$, *and for all* j, $\vec{c} \in U(r(j))$, *if* $0 \leq j < k$ *and* $C^+_{s,r(j)}(\vec{a},\vec{c})$ *then* $(\mathcal{M}, r(j), \vec{b}) \models \varphi$

$(\mathcal{M}, s, \vec{a}) \models E\varphi U\varphi'$ *iff there is* r *s.t.* $r(0) = s$, *and* $k \geq 0$, $\vec{b} \in U(r(k))$ *s.t.* $C^+_{s,r(k)}(\vec{a},\vec{b})$ *and* $(\mathcal{M}, r(k), \vec{b}) \models \varphi'$, *and for all* j, $\vec{c} \in U(r(j))$, *if* $0 \leq j < k$ *and* $C^+_{s,r(j)}(\vec{a},\vec{c})$ *then* $(\mathcal{M}, r(j), \vec{c}) \models \varphi$

$(\mathcal{M}, s, \vec{a}) \models K_i\psi$ *iff for all* s', $\vec{b} \in U(s')$, *if* $s \sim_i s'$ *and* $C^i_{s,s'}(\vec{a},\vec{b})$ *then* $(\mathcal{M}, s', \vec{b}) \models \psi$

$(\mathcal{M}, s, \vec{a}) \models C\psi$ *iff for all* s', $\vec{b} \in U(s')$, *if* $s \sim s'$ *and* $C_{s,s'}(\vec{a},\vec{b})$ *then* $(\mathcal{M}, s', \vec{b}) \models \psi$

$(\mathcal{M}, s, \vec{a}) \models \forall x_{n+1}\psi$ *iff for every* $a^* \in adom(s)$, $(\mathcal{M}, s, \vec{a} \cdot a^*) \models \psi$

where $\vec{a} \cdot a^*$ *is the* $(n+1)$*-assignment* $\langle a_1, \ldots, a_n, a^* \rangle$.

An n*-formula* ϕ *is* true at a state s, *or* $(\mathcal{M}, s) \models \phi$, *iff it is satisfied by every* n*-assignment;* ϕ *is* true on an MA c-model $\mathcal{M}$, *or* $\mathcal{M} \models \phi$, *iff* $(\mathcal{M}, s_0) \models \phi$.

In counterpart semantics the meaning of modal operators is defined by using accessibility relations both on states and individuals. Notice that by considering MA c-models where the counterpart relation is the identity and constants and functions are interpreted as themselves, we have that the relation of satisfaction in Def. 10 reduces to the notion in Def. 5. Thus, MA c-models can really be seen as a generalisation of AC-MAS. Hereafter we state this result formally. First, we define a translation π_n, for $n \in \mathbb{N}$, from FO-CTLK to FO-CTLK$_T$. Given an FO-CTLK formula ϕ and n greater than or equal to the maximum k such that x_k occurs in ϕ, the n-formula $\pi_n(\phi)$ in FO-CTLK$_T$ is inductively defined as follows:

$$\begin{array}{lcl} \pi_n(P^m(t_1,\ldots,t_k)) & := & P^m(t_1,\ldots,t_k) \\ \pi_n(\flat\ \psi) & := & \flat\ \pi_n(\psi) \\ \pi_n(\psi\ \sharp\ \psi') & := & \pi_n(\psi)\ \sharp\ \pi_n(\psi') \\ \pi_n(\forall x_i \psi) & := & \forall x_{n+1}(\pi_n(\psi)[x_i/x_{n+1}]) \end{array}$$

where $\flat$ (resp. $\sharp$) is any unary (resp. binary) connective. Hence, π_n simply renames bound variables in ϕ. We can now state the following result on the relation between an AC-MAS $\mathcal{P}$ and the corresponding MA c-models $\mathcal{M}_\mathcal{P}$ as defined in Remark 1.

LEMMA 1. *Let* $\mathcal{P}$ *be an AC-MAS,* $t(\vec{x})$ *a term,* $\phi(\vec{x})$ *an FO-CTLK formula, and* $\sigma(\vec{x}) = \vec{a}$. *We have that*

$$\begin{array}{lcl} \sigma(t) & = & \vec{a}(t) \\ (\mathcal{P}, s, \sigma) \models \phi & \textit{iff} & (\mathcal{M}_\mathcal{P}, s, \vec{a}) \models \pi_n(\phi) \end{array}$$

This result, proved by induction on the length of t and ϕ, allows us to model check an AC-MAS by verifying the corresponding MA c-model. We tackle the latter task in the next section.

3.1 Simulation

We now introduce a notion of simulation for MA c-models and show that it preserves the satisfaction of formulas in FO-ACTLK$_T$. We start with some preliminaries. In the following we consider the MA c-models $\mathcal{M} = \langle S, s_0, U, \rightarrow, C^t, \{\sim_i\}_{i\in Ag}, \{C^i\}_{i\in Ag}, I\rangle$ and $\mathcal{M}' = \langle S', s'_0, U', \rightarrow', C^{t'}, \{\sim'_i\}_{i\in Ag}, \{C^{i'}\}_{i\in Ag}, I'\rangle$, with $s \in S$ and $s' \in S'$.

DEFINITION 11 (STATE SIMULATION). *A state* s' simulates s, *or* $s \preceq s'$, *iff there exists a surjective function* $\iota : U(s) \rightarrow U'(s')$ *s.t.* (i) *for every constant* c, $\iota(I(c,s)) = I'(c,s')$; (ii) *for every function* f^k, $\vec{u} \in U(s)^k$, $\iota(I(f^k,s)(\vec{u})) = I'(f^k,s')(\iota(u_1),\ldots,\iota(u_k))$; (iii) *and for every* $P_j \in \mathcal{D}$, *and* $\vec{u} \in U(s)^{q_j}$, $\vec{u} \in I(P_j,s)$ *iff* $\iota(\vec{u}) \in I'(P_j,s')$.

Any function ι as above is a *witness* for $s \preceq s'$. We write $s \overset{\iota}{\preceq} s'$ to state this explicitly. Witnesses preserve the interpretation of terms and predicates, but not necessarily the multiplicity of individuals. Notice that by definition $u \in adom(s)$ iff $\iota(u) \in adom(s')$.

DEFINITION 12 (ASSIGNMENT SIMULATION). *Let* $\vec{a} \in U(s)^n$ *and* $\vec{a}' \in U'(s')^n$ *be* n*-assignments,* $(s', \vec{a}')$ simulates $(s, \vec{a})$, *or* $(s, \vec{a}) \preceq (s', \vec{a}')$, *iff for some witness* ι, $s \overset{\iota}{\preceq} s'$ *and* $\iota(\vec{a}) = \vec{a}'$.

We overload the symbol $\preceq$ to represent state and assignment simulations; the difference will be clear from the context. Notice that $\preceq$ is a transitive relation on S and $\bigcup_{s\in S} U(s)$ respectively. Also, assignment simulation preserves the interpretation of FO$_T$-formulas.

LEMMA 2. *If* $(s, \vec{a}) \preceq (s', \vec{a}')$, *then for every* n*-term* t *and* n*-formula* ϕ *in FO*$_T$,

$$\begin{array}{lcl} \iota(\vec{a}(t)) & = & \vec{a}'(t) \\ (\mathcal{M}, s, \vec{a}) \models \phi & \textit{iff} & (\mathcal{M}', s', \vec{a}') \models \phi \end{array}$$

The proof is by induction on the length of t and the length and type of ϕ. We now introduce the notion of simulation on MA c-models, which will be used to extend Lem. 2 to FO-ACTLK$_T$.

DEFINITION 13 (MODEL SIMULATION). *The MA c-model* $\mathcal{M}'$ simulates $\mathcal{M}$, *or* $\mathcal{M} \preceq \mathcal{M}'$, *iff* (i) $s_0 \preceq s'_0$; (ii) *if* $(s, \vec{a}) \preceq (s', \vec{a}')$ *then for every* $t \in S$, $\vec{b} \in U(t)^n$, *if* $s \rightarrow t$ *and* $C^t_{s,t}(\vec{a},\vec{b})$, *then there are* $t' \in S'$, $\vec{b}' \in U'(t')^n$ *s.t.* $s' \rightarrow' t'$, $C^{t'}_{s',t'}(\vec{a}',\vec{b}')$, *and* $(t, \vec{b}) \preceq (t', \vec{b}')$; *and* (iii) *if* $(s, \vec{a}) \preceq (s', \vec{a}')$ *then for every* $t \in S$, $\vec{b} \in U(t)^n$, *if* $s \sim_i t$ *and* $C^i_{s,t}(\vec{a},\vec{b})$, *then there are* $t' \in S'$, $\vec{b}' \in U'(t')^n$ *s.t.* $s' \sim'_i t'$, $C^{i'}_{s',t'}(\vec{a}',\vec{b}')$ *and* $(t, \vec{b}) \preceq (t', \vec{b}')$.

Again, we use the symbol $\preceq$ to express a simulation between MA c-models; the difference will be clear from the context. The simulation relation $\preceq$ extends to MA c-model the commutativity conditions of standard model simulations [8]. Most importantly, since by Remark 1 AC-MAS can be seen as a specific class of MA c-models, Def. 13 applies also to the former. Finally, we can state the main result of this section, namely, the simulation relation on MA c-models preserves the satisfaction of FO-ACTLK$_T$ formulas.

THEOREM 3. *Suppose that* $\mathcal{M} \preceq \mathcal{M}'$ *and* $(s, \vec{a}) \preceq (s', \vec{a}')$. *Then for every* n*-formula* ϕ *in FO-ACTLK*$_T$,

$$(\mathcal{M}', s', \vec{a}') \models \phi \quad \Rightarrow \quad (\mathcal{M}, s, \vec{a}) \models \phi$$

From Thm. 3 we immediately obtain the following result.

COROLLARY 4. *If* $\mathcal{M} \preceq \mathcal{M}'$, *then for every* n*-formula* $\phi \in$ *FO-ACTLK*$_T$, $\vec{a}'_0 \in U'(s'_0)^n$, *there exists* $\vec{a}_0 \in U(s_0)^n$ *such that*

$$(\mathcal{M}', s'_0, \vec{a}'_0) \models \phi \quad \Rightarrow \quad (\mathcal{M}, s_0, \vec{a}_0) \models \phi$$

By Lem. 1 and Cor. 4 we can tackle the model checking problem for an AC-MAS $\mathcal{P}$ and an FO-ACTLK formula ϕ by considering an MA c-model $\mathcal{M}'$ that is similar to $\mathcal{M}_\mathcal{P}$. By Cor. 4, if $(\mathcal{M}', s'_0, \vec{a}'_0) \models \pi_n(\phi)$ for some n-assignment $\vec{a}'_0$, then there exists $\vec{a}_0 \in U(s_0)$ s.t. $(\mathcal{M}_\mathcal{P}, s_0, \vec{a}_0) \models \pi_n(\phi)$. Moreover, by Lem. 1, if $(\mathcal{M}_\mathcal{P}, s_0, \vec{a}_0) \models \pi_n(\phi)$ then $(\mathcal{P}, s_0, \sigma) \models \phi$ for some assignment σ that agrees with $\vec{a}_0$ on the free variables in ϕ. Thus, a positive solution to the model checking problem for the abstract MA c-model $\mathcal{M}'$ implies a positive solution also for the concrete AC-MAS $\mathcal{P}$. In the following sections we analyse the conditions under which the abstract MA c-model $\mathcal{M}'$ is finite and can be constructively given.

4. FINITE ABSTRACTION

In this section we introduce the *abstraction* of a MA c-model $\mathcal{M}$ and show that it is similar to $\mathcal{M}$. Further, we identify the conditions under which such abstraction is finite, thus allowing the verification of infinite-state AC-MAS by the results in Section 3. Hereafter we assume that the sets *Con* of constants and F of functions are finite. This can be done without loss of generality, as we can take *Con* and F as the sets of constants and functions appearing in the formula to be verified. All the results in previous sections still hold for FO-ACTLK formulas on this finite language.

First, we define $[s]$ as the equivalence class of $s \in S$ according to the symmetric closure $\approx$ of the state simulation relation $\preceq$. Further, for $s \in [t]$, $[s,a]_{[t]}$ is the equivalence class of (s,a) according to the symmetric closure $\approx$ of the assignment simulation relation $\preceq$. Notice that we overload the symbol $\approx$ as we did with $\preceq$; the distinction will be clear from the context. Also, $\approx$ is not to be confused with the isomorphism relation $\simeq$ in Section **??**.

DEFINITION 14 (ABSTRACTION). *Given an MA c-model* $\mathcal{M} = \langle S, s_0, U, \rightarrow, C^t, \{\sim_i\}_{i\in Ag}, \{C^i\}_{i\in Ag}, I\rangle$*, the* abstraction *of* $\mathcal{M}$ *is a tuple* $\mathcal{M}' = \langle S', s_0', U', \rightarrow', C^{t'}, \{\sim_i'\}_{i\in Ag}, \{C^{i'}\}_{i\in Ag}, I'\rangle$ *s.t.* (i) $S' = \{[s] \mid s \in S\}$*;* (ii) $s_0' = [s_0]$*; and for all* $[s],[s'] \in S'$*,*

- $U'([s]) = \{[s,a]_{[s]} \mid a \in U(s)\}$
- $[s] \rightarrow' [s']$ *iff there are* $u \in [s]$, $v \in [s']$ *s.t.* $u \rightarrow v$
- $C^{t'}_{[s],[s']} = \{([s,a]_{[s]}, [s',b]_{[s']}) \mid$ *there are* $(u,a') \in [s,a]_{[s]}$, $(v,b') \in [s',b]_{[s']}$ *and* $C^t_{u,v}(a',b')\}$
- $\sim_i'$ *is the transitive closure of relation* $\{([s],[s']) \mid$ *there exist* $u \in [s], v \in [s']$ *s.t.* $u \sim_i v\}$
- $C^{i'}_{[s],[s']}$ *is the transitive closure of relation* $\{([s,a]_{[s]}, [s',b]_{[s']}) \mid$ *there are* $(u,a') \in [s,a]_{[s]}, (v,b') \in [s',b]_{[s']}$ *and* $C^i_{u,v}(a',b')\}$
- *the interpretation* I' *is s.t.* (i) $I'(c,[s]) = [s, I(c,s)]_{[s]}$*;*
(ii) $I'(f^k,[s])([s,a_1]_{[s]},\ldots,[s,a_k]_{[s]}) = [s, I(f^k,s)(\vec{a})]_{[s]}$*;*
(iii) $\langle [s,a_1]_{[s]},\ldots,[s,a_k]_{[s]}\rangle \in I'(P,[s])$ *iff* $\vec{a} \in I(P,s)$.

It can be shown that the abstraction of an MA c-model $\mathcal{M}$ is also an MA c-model. In particular, the interpretation I' is well-defined as it is independent from the specific representative: if $(s',a') \in [s,a]_{[s]}$, then $(s',a') \approx (s,a)$. Thus, $\vec{a} \in I(P,s)$ iff $\vec{a}' \in I(P,s')$.

Given an MA c-model $\mathcal{M}$ and its abstraction $\mathcal{M}'$, we define a mapping $g : \mathcal{M} \rightarrow \mathcal{M}'$ such that for $a \in U(s)$, $g(a) = [s,a]_{[s]}$. We now prove that the mapping g defines simulation relations.

THEOREM 5. *Let* $\mathcal{M}$ *be a MA c-model with abstraction* $\mathcal{M}'$*,*

1. for all $s \in S$*,* g *witnesses a state simulation, i.e.,* $s \overset{g}{\preceq} [s]$*;*
2. for all $s \in S$*,* $\vec{a} \in U(s)^n$*,* g *witnesses an assignment simulation, i.e.,* $(s,\vec{a}) \preceq ([s], g(\vec{a}))$*;*
3. $\mathcal{M}'$ *simulates* $\mathcal{M}$*.*

As a consequence, by Cor. 4 and Thm. 5(3) we obtain the following result.

COROLLARY 6. *Let* ϕ *be an* n*-formula in FO-ACTLK*$_T$*, and* $\mathcal{M}$ *an MA c-model with abstraction* $\mathcal{M}'$*. For every* $\vec{a}_0 \in U(s_0)$*,*

$$(\mathcal{M}', [s_0], g(\vec{a_0})) \models \phi \quad \Rightarrow \quad (\mathcal{M}, s_0, \vec{a}_0) \models \phi$$

Now we investigate the problem of determining sufficient conditions for the abstraction of an MA c-model $\mathcal{M}$ to be finite. This will allow the verification of an infinite-state AC-MAS $\mathcal{P}$ by model checking the finite abstraction of $\mathcal{M}_\mathcal{P}$. It turns out that, since AC-MAS are defined on a finite database schema and we consider simulations on finite sets of constants and functions, abstractions of AC-MAS are always finite.

THEOREM 7. *For every AC-MAS* $\mathcal{P}$*, the abstraction* $\mathcal{M}'$ *of the MA c-model* $\mathcal{M}_\mathcal{P}$ *is finite.*

As a consequence of Cor. 6 and Thm. 7, to verify an FO-ACTLK formula ϕ on an infinite-state AC-MAS $\mathcal{P}$, we can model check $\pi_n(\phi) \in$ FO-ACTLK$_T$ on the finite abstraction of $\mathcal{M}_\mathcal{P}$. We illustrate this methodology by elaborating on Example 1.

EXAMPLE 2. We define the abstraction $\mathcal{M}_2'$ of the AC-MAS $\mathcal{P}_2$ in Example 1. The abstraction $\mathcal{M}_1'$ of $\mathcal{P}_1$ can be obtained similarly and it is depicted in Fig. 2(c), a detailed description is omitted for reasons of space.

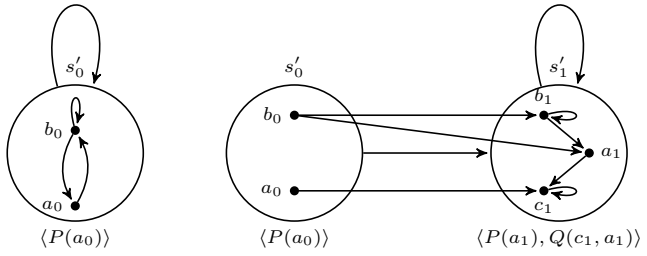
We observe that $\mathcal{P}_2$ contains a unique run $s_0 \rightarrow s_1 \rightarrow \ldots$, as depicted in Fig. 1(b). Further, for every $i > 0$, there is a surjective function $\iota : U \mapsto U$ from s_{i+1} to s_i that satisfies the conditions in Def. 11. Specifically, the witness ι maps $i+1$ to i, any $k < i+1$ to some $k' < i$, and any $k > i+1$ to some $k' > i$. As a result, for every $i > 0$, $s_{i+1} \preceq s_1$. Thus, we define the set of states in the abstraction $\mathcal{M}_2'$ as $S' = \{s_0', s_1'\}$, where $s_0' = [s_0] = \{s_0\}$ and $s_1' = [s_1] = \{s_i \mid i > 0\}$, according to the equivalence relation $\approx$. In particular, $s_0' \rightarrow' s_1'$ and $s_1' \rightarrow' s_1'$ by definition of $\rightarrow'$. Further, for every $n, m > 0$, $(s_m, m) \preceq (s_n, n)$ by the witnesses ι defined as above. Moreover, if $n' < n, m' < m$ (resp. $n' > n, m' > m$), we have that $(s_m, m') \preceq (s_n, n')$. Hence, we obtain two equivalence classes on $\mathbb{N}$ for $n = 0$: $a_0 = \{(s_0, 0)\}$, $b_0 = \{(s_0, n') \mid n' \neq 0\}$; while for $n > 0$ we obtain three equivalence classes on $\mathbb{N}$: $a_n = \{(s_n, n)\}$, $b_n = \{(s_n, n') \mid n < n'\}$ and $c_n = \{(s_n, n') \mid n > n'\}$. Further, the counterpart relation is defined as $C^t_{s_0',s_1'} = \{(a_0,c_1),(b_0,b_1),(b_0,a_1)\}$, corresponding intuitively to the transitions $(s_0,0) \rightarrow (s_1,0)$, $(s_0,n') \rightarrow (s_1,n')$ for $n' > 1$, and $(s_0,1) \rightarrow (s_1,1)$. Moreover, $C^t_{s_1',s_1'}$ is equal to $\{(a_1,c_1),(b_1,b_1),(b_1,a_1),(c_1,c_1)\}$, which corresponds to the transitions $(s_n,n) \rightarrow (s_{n+1},n)$, $(s_n,n') \rightarrow (s_{n+1},n')$ for $n' > n+1$, $(s_n,n+1) \rightarrow (s_{n+1},n+1)$, and $(s_n,n') \rightarrow (s_{n+1},n')$ for $n' < n$. As regards the epistemic components, by Def. 14 we have that $\sim_0$ and $\sim_2$ are the identity relation, while $\sim_1$ is equal to $S' \times S'$. Moreover, $C^i_{s_j',s_j'}$ is the identity for every $A_i \in Ag'$ and $j = 0,1$; while $C^1_{s_0',s_1'} = C^1_{s_1',s_0'} = \{(a_0,c_1),(c_1,a_0),(b_0,b_1),(b_1,b_0),(b_0,a_1),(a_1,b_0)\}$. Finally, we remark that the FO-ACTLK formulas χ and θ do not contain neither constants nor functions. Hence, the interpretation I' is s.t. $I'(P,s_0') = \{a_0\}$, $I'(Q,s_0') = \emptyset$, $I'(P,s_1') = \{a_1\}$, and $I'(Q,s_1') = \{(c_1,a_1)\}$. The abstract MA c-model $\mathcal{M}_2'$ is illustrated in Fig.2 (d).

We can now model check on $\mathcal{M}_1'$ and $\mathcal{M}_2'$ the typed versions of χ and θ, namely

$$\begin{aligned} \chi_T &= AG\,\forall x_1 K_0(P(x_1) \rightarrow AX\neg P(x_1)) \\ \theta_T &= AG\,\forall x_1 K_1(P(x_1) \rightarrow AXAG\neg P(x_1)) \end{aligned}$$

It can be checked that $\mathcal{M}_2' \models \chi_T \wedge \theta_T$, and $\mathcal{M}_1' \models \chi_T$, while $\mathcal{M}_1' \not\models \theta_T$. Hence, by Lem. 1 and Cor. 6 we can derive that $\mathcal{P}_2 \models \chi \wedge \theta$, and $\mathcal{P}_1 \models \chi$; while nothing can be said about θ in $\mathcal{P}_1$. As a result, whenever the abstraction satifies the FO-ACTLK specification, we are able to model check AC-MAS even though these are neither uniform nor bounded. ■

We conclude by remarking that, since we consider surjective mappings between states, multiple individuals can be "compressed" into one single abstract individual. Hence, the abstraction of an AC-MAS as defined in Def. 14 is not an AC-MAS in general, rather an MA c-model that includes temporal and epistemic counterpart relations on its elements. This motivates the introduction of MA c-models in first place.



(a) the MA c-model $\mathcal{M}'_1$

(b) the MA c-model $\mathcal{M}'_2$

Figure 2: the MA c-models $\mathcal{M}'_1$ and $\mathcal{M}'_2$ (epistemic components are omitted)

4.1 Towards Constructive Abstraction

In the previous section we defined the abstraction of an MA c-model $\mathcal{M}$ and proved that it is similar to $\mathcal{M}$. Also, in Thm. 7 we remarked that this abstraction is finite, thus allowing for model checking a specification on a concrete, infinite-state AC-MAS $\mathcal{P}$ by verifying the finite abstraction of $\mathcal{M}_{\mathcal{P}}$. However, the definition of abstraction provided in Section **??** is not constructive in general, as it relies on the quotient structure $S_{/\approx}$ of the infinite state space S of $\mathcal{M}$. In this section we explore a mechanism to build finite abstractions constructively, starting from the way actions and transitions are specified. The following definition of actions has first appeared in a different form in [5].

DEFINITION 15 (ACTIONS). *For each service agent A_i, Act_i is the set of* action types $\alpha(\vec{x}) = \langle \pi(\vec{y}), \psi(\vec{z})\rangle$, *where*

- $\alpha(\vec{x})$ *is the* action signature *with* $\vec{x} = \vec{y} \cup \vec{z}$ *as* parameters*;*
- $\pi(\vec{y})$ *is the* action precondition*, i.e., an FO$^{=}$-formula over $\mathcal{D}_i$;*
- $\psi(\vec{z})$ *is the* action postcondition*, i.e., an FO$^{=}$-formula over $\mathcal{D} \cup \mathcal{D}'_i$.*

Observe that we admit the use of identities in the action specifications and, differently from [5], of functions as well. Given actions as in Def. 15, we can define the transition function as follows, where F_ϕ is the *finite* set of function symbols appearing in a formula ϕ, and if k is the maximum nesting of function symbols in ϕ, $F^k_\phi(\vec{u})$ is the *finite* set of elements obtained by applying the functions $f \in F_\phi$ to individuals in $\vec{u}$ at most k times.

DEFINITION 16 (PROTOCOL AND TRANSITION FUNCTION). *Let $\alpha_i(\vec{x}_i) = \langle \pi_i(\vec{y}_i), \psi_i(\vec{z}_i)\rangle$ be an action for service agent $A_i \in Ag$, and $\vec{u}_i = \vec{v}_i \cup \vec{w}_i$ individuals in the domain U. Then, $\alpha_i(\vec{u}_i) \in Pr_i(l_i)$ iff $(l_i, \sigma) \models \pi_i(\vec{y}_i)$ for $\sigma(\vec{y}_i) = \vec{v}_i$.*

Further, for an AC-MAS $\mathcal{P}$ on Ag the transition function τ is defined so that, for $s = \langle l_0, \ldots, l_n\rangle$ and $s' = \langle l'_0, \ldots, l'_n\rangle$, $s' \in \tau(s, \langle \alpha(\vec{u})\rangle)$ iff for every $A_i \in Ag$,

1. $(D_s \oplus l'_i, \sigma') \models \psi_i(\vec{z}_i)$ *for* $\sigma'(\vec{z}_i) = \vec{w}_i$
2. $adom(l'_i) \subseteq adom(l_i) \cup F^k_{\psi_i}(\vec{w}_i) \cup con(\psi_i)$

In Def. 16 we consider a satisfaction relation $\models$ between database instances and FO$^{=}$-formulas, which can be derived from Def. 5. Also, $\oplus$ is the disjoint union of db instances introduced in Section **??**. Notice that by condition (2) the number of candidates for each l'_i is finite. Also, condition (1) can be effectively computed. Hence, the transition function is decidable. Notice that in general we have multiple successors s' for given s and $\alpha(\vec{u})$, but always in finite number.

EXAMPLE 3. We show that the agent services in Example 1 and 2 can be described by means of actions as specified in Def. 15. For each agent A_i, $i \leq 2$, we assumed that $Act_i = \{\alpha_i(n)\}$, where each $\alpha_i(n) = \langle \pi_i, \psi_i(n)\rangle$ is specified as

$$\pi_i = true$$
$$\psi_i = (((n-i)\%3 = 2) \wedge P(n) \wedge P'(n+1) \wedge \forall x((x \neq n+1) \rightarrow \neg P'(x))) \vee (((n-i)\%3 \neq 2) \wedge P(n) \wedge \forall x \neg P'(x))$$

Further, for A'_2 we have that $\alpha'_2(n) = \langle \pi'_2, \psi'_2(n)\rangle$ is specified as

$$\pi'_2 = true$$
$$\psi'_2 = \psi_2 \wedge Q'(n, n+1) \wedge \forall x, y(Q(x,y) \rightarrow Q'(x, n+1) \wedge Q'(y, n+1)) \wedge (Q'(x,y) \rightarrow (y = n+1) \wedge \exists z(Q(x,z) \vee Q(z,x)))$$

It can be checked that the protocols and transition function as defined in Def. 16 correspond indeed to those for the AC-MAS P_1 and $\mathcal{P}_2$ in Example 1. Notice the use of function symbols '+1' (successor) and '%3' (remainder of division by 3), which entails that the AC-MAS thus obtained are not uniform in general. ■

Actions specified as in Def. 15 generate AC-MAS that are not necessarily uniform. This is in contrast with the situation for AC-MAS described by actions on a language without function symbols. For that restricted language it was shown that the corresponding AC-MAS are always uniform [5, 6]. Thus, the increased expressiveness of the specification language for actions requires the novel verification technique put forward in this paper.

Now we explore methods to obtain finite abstractions constructively and consider the following condition on AC-MAS.

DEFINITION 17 (WEAK UNIFORMITY). *An AC-MAS $\mathcal{P}$ is* weakly uniform *iff for every $s, s' \in \mathcal{S}$, if $s \approx s'$, $s \xrightarrow{\alpha(\vec{u})} t$ and $s' \xrightarrow{\alpha(\vec{u}')} t'$, then $t \approx t'$.*

If an AC-MAS is weakly uniform, then its temporal evolution modulo the relation $\approx$ is captured by the actions independently from ground parameters. Weak uniformity still allows for great expressiveness. In particular, the AC-MAS in Examples 1-3 are weakly uniform. Notice that weak uniformity is strictly weaker than fully-fledge uniformity; for instance the AC-MAS $\mathcal{P}_1$ is weakly uniform but not uniform. Also, weakly uniformity admits unboudedness as it is the case for the AC-MAS $\mathcal{P}_2$.

Now we briefly describe the construction of the abstract MA c-model $\mathcal{M}'_{\mathcal{P}}$ for a weakly uniform AC-MAS $\mathcal{P}$. We keep this discussion informal for reasons of space. Starting from the initial state s_0, for each $s \in \mathcal{S}$ and each joint action $\alpha(\vec{x})$ we consider all tuples $\vec{u}$ containing elements in $adom(s)$ together with at most $|\vec{x}|$ elements not in $adom(s)$ (notice that the elements in $U \setminus adom(s)$ behave in the same way w.r.t. the satisfaction of FO$^{=}$-formulas). Obviously the number of such tuples is finite. Further, if $s \xrightarrow{\alpha(\vec{u})} t$ for some $\vec{u}$, then we add the representative t to the set of reachable states in the abstraction, unless $t \approx t'$ for some t' already appearing in the set of reachable states. Since the number of equivalence classes according to $\approx$ is finite, this construction terminates after a finite number of steps, returning the abstraction $\mathcal{M}'_{\mathcal{P}}$.

EXAMPLE 4. To give a hint of the methodology proposed, we build part of the abstract MA c-model $\mathcal{M}'_2$ for the unbounded and

weakly uniform AC-MAS $\mathcal{P}_2$, starting from the initial state s_0. We first observe that, since the active domain of s_0 is finite and all non-active individuals can be mapped to any $u \in \mathbb{N} \setminus adom(s_0)$, we obtain two equivalence classes in $s'_0 = [s_0]$, that is, $a_0 = [s_0, 0]_{[s_0]} = \{(s_0, 0)\}$ and $b_0 = [s_0, n]_{[s_0]} = \{(s_0, n) \mid n \neq 0\}$. Then, we consider the only joint action $\alpha(n)$ in $\mathcal{P}_2$ with $n \in adom(s_0) = \{0\}$ or n equal to any $u \in \mathbb{N} \setminus adom(s_0)$. The transition triggers only in the former case and we have $s_0 \xrightarrow{\alpha(0)} s_1$. We add s_1 to the set of reachable states in $\mathcal{M}'_2$ as $s_1 \not\approx s_0$. Again, we consider the joint action $\alpha(n)$ for $n \in adom(s_1) = \{0, 1\}$ or n equal to any $u \in \mathbb{N} \setminus adom(s_1)$. Now the transition is triggered only for $n = 1$ with $s_1 \xrightarrow{\alpha(1)} s_2$. Moreover, by considering the finitely many subjective functions from $adom(s_2)$ to $adom(s_1)$ we can check that $s_2 \preceq s_1$, that is, $s_2 \in [s_1]$. Therefore, in the abstract $\mathcal{M}'_2$ we only add a reflexive transition on $[s_1]$. This concludes the construction of the abstraction $\mathcal{M}'_2$ for $\mathcal{P}_2$, as by weak uniformity we know that if $s_2 \xrightarrow{\alpha(u)} t$ for some $u \in \mathbb{N}$, then in particular, since $s_1 \xrightarrow{\alpha(1)} s_2$ and $s_1 \approx s_2$, it is the case that $t \approx s_2$, i.e., $t \in [s_2] = [s_1]$. ■

As a result, if an AC-MAS is weakly uniform, even though it is neither uniform nor bounded, we know that a finite similar abstraction can be constructively obtained.

5. CONCLUSIONS AND FURTHER WORK

In this paper we introduced an abstraction methodology for a class of artifact-centric systems that was not yet tractable by the current state-of-the-art, namely non-uniform and unbounded AC-MAS. We remarked that non-uniform AC-MAS rise naturally when considering specification languages for agent actions that contain functions. This feature is often crucial to express functional dependency between data in our underlying databases, as well as their temporal evolution. Further, we identified the class of weakly uniform AC-MAS, which admits a constructive procedure for building abstractions, while being strictly weaker than fully-fledge uniformity, as it allows for unbounded systems. Finally, we see the simulation and abstraction notions put forward as interesting theoretical contributions *per se* to the semantics of first-order modal logic.

Related Work. To the best of our knowledge, [7, 18] are among the first contributions to consider the verification of artifact-centric business processes. This line of research was pursued further in [16, 13], where the decidability of the model checking problem is obtained by syntactically restricting the system description and the specification language. Differently from [16, 13], we do not constraint the alternation of quantifiers and modal operators. This constitutes a major challenge for the development of abstraction techniques. Closely related to the present contribution is [19], where conditions for decidable model checking of data-centric dynamic systems are given. However, also [19] restricts quantification. Moreover, this paper differs from similar results in [4, 5, 6, 15] as we do not assume uniformity nor boundedness as conditions to obtain finite abstractions. This approach is in common with [3], but here we consider a multi-agent system setting and a specification language including epistemic operators. Most importantly, our language contains function symbols. This features entails that our AC-MAS are not uniform in general, thus calling for more sophisticated abstraction techniques. Finally, we also presented some preliminary investigations into constructive abstraction building.

There is a number of directions to extend the results here presented. In particular, we aim at developing the methodology for building abstractions of weakly uniform systems outlined in Section **??** into an effective verification technique. In this respect, results on the complexity of the procedure, that has not been addressed here, are of utmost importance.

Acknowledgements: this paper has benefited from the author's collaboration with Prof A. Lomuscio and Dr F. Patrizi. The author would also like to thank the anonymous referees.

6. REFERENCES

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.
[2] C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT, 2008.
[3] F. Belardinelli and A. Lomuscio. Decidability of model checking non-uniform artifact-centric quantified interpreted systems. In *Proc. of IJCAI'13*. IJCAI/AAAI, 2013.
[4] F. Belardinelli, A. Lomuscio, and F. Patrizi. Verification of Deployed Artifact Systems via Data Abstraction. In *Proc. of ICSOC'11*, pp. 142–156, 2011.
[5] F. Belardinelli, A. Lomuscio, and F. Patrizi. An Abstraction Technique for the Verification of Artifact-Centric Systems. In *Proc. of KR'12*, pp. 319 – 328, 2012.
[6] F. Belardinelli, A. Lomuscio, and F. Patrizi. Verification of GSM-based artifact-centric systems through finite abstraction. In *Proc. of ICSOC'12*, pp. 17–31, 2012.
[7] K. Bhattacharya et al. Towards Formal Analysis of Artifact Centric Business Process Models. In *Proc. of BPM*, 2007.
[8] P. Blackburn et al. *Modal Logic*, Cambridge UP, 2001.
[9] T. Brauner and S. Ghilardi. First-order modal logic. In *Handbook of Modal Logic*, pp. 549–620. Elsevier, 2007.
[10] J. Burch et al. Symbolic Model Checking. *Information and Computation*, 98(2):142–170, 1992.
[11] E. Clarke et al. *Model Checking*. MIT, 1999.
[12] G. Corsi. *Counterparts and possible worlds*. CLUEB, 2001.
[13] E. Damaggio, A. Deutsch, and V. Vianu. Artifact Systems with Data Dependencies and Arithmetic. *ACM Transactions on Database Systems*, 37(3):22:1–22:36, 2012.
[14] E. Damaggio et al. On the Equivalence of Incremental and Fixpoint Semantics for Business Artifacts with Guard-Stage-Milestone Lifecycles. In *Proc. of BPM*, 2011.
[15] G. De Giacomo, Y. Lespérance, and F. Patrizi. Bounded Situation Calculus Action Theories and Decidable Verification. In *Proc. of KR'12*, pp. 467–477, 2012.
[16] A. Deutsch et al. Automatic Verification of Data-Centric Business Processes. In *Proc. of ICDT*, 2009.
[17] R. Fagin et al. *Reasoning About Knowledge*. MIT, 1995.
[18] C. Gerede and J. Su. Specification and Verification of Artifact Behaviors in Business Process Models. In *Proc. of ICSOC'07*, pp. 181–192, 2007.
[19] B. B. Hariri et al. Verification of relational data-centric dynamic systems with external services. In *Proc. of PODS'13*, pp. 163–174. ACM, 2013.
[20] R. Hull. Artifact-centric business process models. In *Proc. of OTM Conferences (2)*, pp. 1152–1163, Springer, 2008.
[21] R. Hull et al. Business artifacts with guard-stage-milestone lifecycles. In *Proc. of DEBS'11*, pp. 51–62, ACM, 2011.
[22] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS. In *Proc. of CAV'09*, pp. 682–688, 2009.
[23] C. Pasareanu and G. Salaün, editors. *Formal Aspects of Component Software, FACS'12*. Springer, 2013.
[24] M. Singh et al. *Service-Oriented Computing*. Wiley, 2005.
[25] M. Wooldridge. Computationally Grounded Theories of Agency. In *Proc. of ICMAS*, pp. 13–22. IEEE Press, 2000.
[26] M. Wooldridge. *Introduction to Multiagent Systems*. John Wiley & Sons, Inc., 2001.