# On the Formal Verification of Diffusion Phenomena in Open Dynamic Agent Networks

Francesco Belardinelli
Université d'Evry
France
belardinelli@ibisc.fr

Davide Grossi
University of Liverpool
United Kingdom
d.grossi@liverpool.ac.uk

## ABSTRACT

The paper is a contribution at the interface of social network theory and multi-agent systems. As realistic models of multi-agent systems, we assume agent networks to be open, that is, agents may join or leave the network at run-time, and dynamic, that is, the network structure may change as a result of agents actions. We provide a formal model of open dynamic agent networks (ODAN) in terms of interpreted systems, and define the problem of model checking properties of diffusion phenomena, such as the spread of information or diseases, expressed in a first-order version of computation-tree logic. We establish the decidability of the model checking problem by showing that, under specific conditions, the verification of infinite-state ODAN can be reduced to model checking finite bisimulations.

## Categories and Subject Descriptors

F.4.1 [**Mathematical Logic**]: Temporal Logic

## General Terms

Theory,Verification

## Keywords

Agent Networks; Diffusion Phenomena; First-order Temporal Logic

## 1. INTRODUCTION

Social network theory and analysis (SNA) is a thriving area of research (see [14, 11] for comprehensive introductions to the field) and its interaction with the field of multi-agent systems (MAS) has witnessed a steady growth over the last decade. On the one hand the MAS paradigm has established itself as a recognized tool within SNA, in particular for simulation purposes (e.g., [15]). On the other hand, concepts and methods from SNA are reaching the theory of MAS as a growing number of papers (e.g, [18, 19, 21], just to mention a few in the last edition of AAMAS) and events (e.g., the *Social Networks and Multi-Agent Systems Symposium* series) on the application of SNA to MAS testify.

**Appears in:** *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May, 4–8, 2015, Istanbul, Turkey.*


The present paper contributes further to the application of SNA concepts to MAS, bringing them closer to standard MAS concerns such as openness, and deploying established methods in MAS research such as formal verification. We focus specifically on (non-probabilistic) diffusion phenomena, that is, how information, ideas, behaviors spread in networks of agents similarly to epidemics. We model a MAS as a network, that is, a set of agents – the nodes – which are linked via edges constraining their possible interactions. As typical in SNA, such links are abstractions of concrete relationships such as proximity, or the availability of communication channels, or trust relationships, and the like. However, although building on established concepts from SNA, the paper focuses on the type of networks that are of specific relevance for MAS, namely networks that are *open*, as agents can enter and leave the MAS at run-time; and *dynamic*, as the network links can change as the direct result of agents' actions.[1]

**Aim of the paper and methodology.** The paper investigates the applicability in principle of formal verification techniques to the analysis of diffusion phenomena in agent networks which may evolve over time as the result of agents' actions. More concretely, the paper establishes the decidability of the model-checking problem for properties of open and dynamic networks specified in a first-order temporal logic (FO-CTL). This result is obtained by modelling networks as a special type of infinite-state data-aware systems [7, 9]. This allows us to capitalize on recent results on the formal verification of artifact-centric systems [2, 3], and to extend them to MAS where agents can join or leave at run-time.

**Related Work.** From the technical point of view the paper builds on two strands of research: the *application of logic to SNA*, and the *verification of data-aware systems*. The application of logic-based methods to SNA is a very recent area of research. Researchers have focused in particular on the formalization of information dynamics phenomena, mainly using dynamic epistemic logic (DEL, [22]), over networks, e.g., [5, 6]. In particular, we are aware of only one study attempting the application of formal verification techniques to SNA, and in particular to the study of epidemics [20], which however does not focus on open and dynamic networks.

The verification of data-aware systems, i.e., systems where data play a crucial role in directing the system's execution, is in itself a subject of growing interest. In [10, 9] the verification of data-driven web services and business processes

[1]It may be worth noting that dynamic networks are themselves an open research area in SNA with no established models (cf. [14, Ch. 7])

is tackled by assuming syntactic restrictions on the specification language, and [13] investigated the verification of dynamic relational databases. With a different focus, networks have been used also in concurrency theory as an abstraction for modeling and verifying communicating processes (e.g., [16, 8]). More directly related to our contribution, [2, 3] put forward abstraction techniques for the verification of artifact-centric systems. While we make use of ideas and notions appearing therein, the motivation and setting for our paper are markedly different. First, the inspiration comes from SNA and the formal analysis of agents' behaviours in networked contexts. Second, the agent networks here introduced are open and dynamic, in particular agents can join or leave the network at run time. None of these features is considered in [2, 3].

**Outline of the paper.** The paper is structured as follows. In Section 2 we introduce a multi-agent, data-aware model for open dynamic agent networks (ODAN), and a first-order temporal specification language. Then, we state the model checking problem for this setting. Section 3 is devoted to show that the proposed framework is rich enough to express established network models such as the SIR (susceptible-infected-recovered) model for epidemics. The main technical results are presented in Section 4 and 5, where we prove that under specific conditions the model checking problem for ODAN is decidable. We conclude in Section 6. Proofs are omitted in the interest of space.

# 2. OPEN DYNAMIC AGENT NETWORKS

In this section we introduce open dynamic agent networks (ODAN) as multi-agent systems whose main feature is that agents can join or depart at run time. Then, we present a first-order version of the branching-time logic CTL, and state the model checking problem for this setting. We first present the basic terminology on databases that will be used throughout the paper; we refer to [1, 3] for further details.

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 $P$ *with arity* $q \in \mathbb{N}$.

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

Given a database schema $\mathcal{D}$, $\mathcal{D}(X)$ is the set of all $\mathcal{D}$-instances on domain $X$; while 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.e., $adom(D) = \bigcup_{P \in \mathcal{D}} \{u_1, \ldots, u_q \in X \mid \langle u_1, \ldots, u_q \rangle \in D(P)\}$. Further, 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 such that *(i)* $D \oplus D'(P) = D(P)$, and *(ii)* $D \oplus D'(P') = D'(P)$. Intuitively, primed versions and disjoint unions will be used to describe the temporal evolution of a database schema from the previous state $D$ to the next state $D'$.

## 2.1 Agents in ODAN

In this paper we focus on agents manipulating data organized in relational structures, according to some database schema. Specifically, hereafter we assume a finite number of *agent types* $T_0, \ldots, T_k$. Each agent type $T$ is associated with *(i)* a *local database schema* $\mathcal{D}_T$, containing a reserved unary predicate symbol $N \in \mathcal{D}_T$ to represent the network structure, and *(ii)* a finite set $Act_T$ of *parametric actions* $\alpha(\vec{x})$ with parameters $\vec{x}$. Hence, agents of the same type share the database schema and available actions. For every agent type $T$, let $Ag_T$, $Ag'_T$, ... be possibly infinite sets of agent names. In the following we assume that the interpretation domain $X$ for database schemas contains a set $Ag_T$ of names for agents of type $T$, that is, $X = Ag \cup U$ for $Ag = \bigcup_{type\ T} Ag_T$ and some other set $U$ of elements. Also, the interpretation $D(N)$ of predicate $N \in \mathcal{D}_T$ is a subset of $Ag$, i.e., intuitively $D(N)$ stores the agents related to a specific agent. To account for the temporal evolution of ODAN, we introduce protocols for agent types. First of all, we consider a notion of isomorphism between database instances.

Definition 2 (Instance Isomorphism). *Two instances* $D \in \mathcal{D}(X)$ *and* $D' \in \mathcal{D}(X')$ *are* isomorphic*, or* $D \simeq D'$*, iff for some bijection* $\iota : adom(D) \mapsto adom(D')$*,*

(i) $\iota$ *preserves the type of agents, i.e., for every type* $T$*,* $\iota$ *is a bijection from* $adom(D) \cap Ag_T$ *into* $adom(D') \cap Ag'_T$*;*

(ii) *for every* $P \in \mathcal{D}$*,* $\vec{u} \in X^q$*,* $\vec{u} \in D(P)$ *iff* $\iota(\vec{u}) \in D'(P)$.

*Then, we say that* $\iota$ *is a* witness *for* $D \simeq D'$ *and write* $D \overset{\iota}{\simeq} D'$ *to state this explicitly.*

Once we have a notion of isomorphism between states, we can introduce the *local protocol* $Pr_T$ for a type $T$ as follows.

Definition 3 (Protocol). *Given an interpretation domain* $X$*,* $Pr_T$ *is a function from* $\mathcal{D}_T(X)$ *to* $2^{Act_T(X)}$*, where* $Act_T(X)$ *is the set of* ground actions $\alpha(\vec{u})$*, for* $\alpha(\vec{x}) \in Act_T$ *and* $\vec{u} \in X^{|\vec{x}|}$.

Thus, the protocol $Pr_T$ returns a ground action in $Act_T(X)$ for every $\mathcal{D}_T$-instance of type $T$. Hereafter we assume the following constraint on protocol functions: *for every instance* $D \in \mathcal{D}_T(X)$ *and* $D' \in \mathcal{D}_T(X')$*, if* $D \overset{\iota}{\simeq} D'$ *then* $\alpha(\vec{u}) \in Pr_T(D)$ *iff* $\alpha(\iota(\vec{u})) \in Pr_T(D')$. This constraint ensures that isomorphic states allow the same actions whenever "isomorphic" values are substituted to parameters. As an example, sending an email is allowed in all states where a valid receiver, subject and email body are provided, independently from the actual data content of these fields.

We finally introduce the notion of *agent*.

Definition 4 (Agent). *Given an agent name* $a \in Ag_T$ *of type* $T$*, an* agent *is a tuple* $a = \langle \mathcal{D}_T, Act_T, Pr_T \rangle$ *where* $\mathcal{D}_T$*,* $Act_T$*, and* $Pr_T$ *are defined as above.*

Notice that above we assumed only a finite number of agent types. However, for each type we can have an infinite number of agents in priciple. This modelling choice is motivated by the use case scenario in Section 3. Also, in domains of interest typically it is possible to specify the relevant agent types at design time. However, it is much more difficult, viz. impossible, to know exactly how many agents of each type will appear during the system's execution. Accounting for incoming and outgoing agents is a major challenge we tackle in the proposed framework.

In what follows we often identify an agent with her name and write $a = \langle \mathcal{D}_a, Act_a, Pr_a \rangle$, omitting the type, whenever this is clear by the context. By Def. 4 at each moment agent $a$ is in some local state $l \in \mathcal{D}_a(U \cup Ag)$ that represents the information she has about the system as well as fellow agents.

In this respect we follow the *interpreted systems* approach to MAS [17, 12, 23], but a fundamental difference is that here we require that the agent's information is structured as a relational database. Also, agent $a$ is assumed to perform the actions in $Act_a$ according to protocol function $Pr_a$. Finally, the database schema $\mathcal{D}_a$ contains the unary predicate symbol $N$ to store the agents which $a$ is related to.

Since we are interested in the interactions amongst agents and with the external environment, we define their synchronous composition, beginning with the notions of global state and network.

Definition 5 (Global State). *Given a* finite *subset $A \subseteq Ag$ of agents $a_i = \langle \mathcal{D}_i, Act_i, Pr_i \rangle$ on domain $X = U \cup Ag$, for $i \leq n$, a* global state *is a tuple $s = \langle l_0, \ldots, l_n \rangle$ of instances $l_i \in \mathcal{D}_i(X)$ such that $\bigcup_{i \leq n} adom(l_i) \cap Ag \subseteq A$.*

By Def. 5 a global state accounts only for a finite set of agents, who are meant to be the *active* agents at a specific time in the system's execution. This assumption is consistent with the literature on MAS [17, 12, 23], where global states are tuples of fixed length. Also, by definition a global state $s$ accounts at least for all agents appearing in its active domain $adom(s) = \bigcup_{i \leq n} adom(l_i)$. That is, if some agent $a$ is mentioned in the local state of some other agent $b \in A$ (notably in the network relation $N$), and thus $a \in adom(s)$, then $a$ also belongs to $A$. In what follows we implicitly identify global states containing the same local states for the same agents, only in a different order. This can be done w.l.o.g. by assuming a fixed enumeration of agents. Further, let $ag$ be a function that for each global state $s = \langle l_0, \ldots, l_n \rangle$ returns the set $ag(s) = \{a_0, \ldots, a_n\}$ of agents such that $l_i \in \mathcal{D}_{a_i}(X)$ for $i \leq n$. By the constraint above on global states we have that for every state $s$, $adom(s) \cap Ag \subseteq ag(s)$. We write $\mathcal{S}$ to denote the set $\bigcup_{n \in \mathbb{N}} (\prod_{i \leq n} \mathcal{D}_{a_i}(X))$ of all global states. Notice that $\mathcal{S}$ is infinite whenever $Ag$ is.

We already remarked that the unary predicate symbol $N$ is used to encode the network structure of the global state. More precisely, given a state $s$ we define the network induced by $N$ as follows.

Definition 6 (Agent Network). *The* agent network *induced by state $s$ is the directed graph $\mathcal{N}_s = \langle ag(s), E \rangle$ where* (i) *$ag(s) \subseteq Ag$ is the set of vertices, and* (ii) *$E$ is the binary relation on $ag(s)$ such that $E(a, b)$ iff $b \in l_a(N)$.*

The agent networks in Def. 6 are digraphs in general. However, by assuming suitable conditions on global states, namely that $b \in l_a(N)$ iff $a \in l_b(N)$, we can also model undirected graphs. The latter might be more appropriate when modelling particular agent networks, as in Section 3.

Finally, we introduce open dynamic agent networks.

Definition 7 (ODAN). *Given a (possibly infinite) interpretation domain $X = Ag \cup U$ containing a (possibly infinite) set $Ag = \{a_0, a_1, \ldots\}$ of agents $a_i = \langle \mathcal{D}_i, Act_i, Pr_i \rangle$, an* open dynamic agent network *is a tuple $\mathcal{P} = \langle Ag, U, I, \tau \rangle$ where*

- *$I$ is the set of* initial states *$s_0$ for some finite $ag(s_0) \subseteq Ag$;*
- *$\tau : \mathcal{S} \times Act(X) \mapsto 2^{\mathcal{S}}$ is the* global transition function, *where $Act$ is the set of joint (parametric) 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 ODAN evolves from an initial state $s_0 \in I$ as specified by the global transition function $\tau$, which returns a set $\tau(s, \alpha(\vec{u})) \in 2^{\mathcal{S}}$ of successor states for each current state $s$ and joint ground action $\alpha(\vec{u})$ by all agents in $s$. Since the interpretation domain $X$ is infinite in general, ODAN are infinite-state systems normally. In this respect, ODAN can be thought of as a natural extension of interpreted systems to a first-order setting. Moreover, ODAN are *open* and *dynamic* as global states may be tuples of different length, comprising a variable number of agents. The transition function is defined only for joint actions providing an individual action for each agent in the current global state, but the resulting state may include fewer or more agents. This is in marked contrast with most of the current literature on MAS [17, 12, 23], which assumes that the set of agents is finite and fully specified at design time.

Hereafter we introduce a further constraint on joint actions in ODAN. To present it we need to extend the notion of isomorphism to global states.

Definition 8 (State Isomorphism). *The global states $s \in \mathcal{S}$ and $s' \in \mathcal{S}'$ are* isomorphic, *or $s \simeq s'$, iff for some bijection $\iota : adom(s) \cup ag(s) \mapsto adom(s') \cup ag(s')$,* (i) *$\iota$ preserves the type of agents; and* (ii) *for every $a_j \in ag(s)$, $l_j \overset{\iota}{\simeq} l'_j$.*

Any function $\iota$ as above is a *witness* for $s \simeq s'$, also indicated as $s \overset{\iota}{\simeq} s'$. Obviously, $\simeq$ is an equivalence relation. Further, given an injective function $f : X \mapsto X'$ such that its restriction $f_{|Ag}$ is a type-preserving injection from $Ag$ to $Ag'$, $f(s)$ denotes the instance in $\mathcal{D}(X')$ obtained from $s$ by renaming each $u \in adom(s) \cup ag(s)$ as $f(u)$. In particular, $f(s) \simeq s$. As a consequence of Def. 8, isomorphic states are tuples of the same length. Hereafter we consider the following constraint on the transition functions in ODAN: *for every state $s \in \mathcal{S}$ and $s' \in \mathcal{S}'$, if $s \overset{\iota}{\simeq} s'$ then $t \in \tau(s, \alpha(\vec{u}))$ iff $\iota(t) \in \tau(s', \alpha(\iota(\vec{u})))$*. Similarly to the condition on protocols, this constraint requires that actions performed with "isomorphic" values in isomorphic states, also return isomorphic states. In other words, "isomorphic" actions are invariant w.r.t. the relational structure of states. For instance, sending emails returns isomorphic states modulo the emails' actual data content.

It is important to stress that state isomorphism is an extremely natural condition. It amounts to the preservation of the interpretation of predicates in each local state up to renaming of corresponding agents and elements of the domain. In other words, it ensures that the behavior of the system does not depend on how agents or elements are named.

We now introduce some notation that will be used in the paper. We denote a joint (ground) action in $\prod_{i \leq n} Act_i(X)$, for $n \in \mathbb{N}$, as $\alpha(\vec{u})$, for $\alpha = \langle \alpha_0(\vec{x}_0), \ldots, \alpha_n(\vec{x}_n) \rangle$ and $\vec{u} = \langle \vec{u}_0, \ldots, \vec{u}_n \rangle$, and define the *transition relation* $s \to s'$ on global states iff $s \xrightarrow{\alpha(\vec{u})} s'$ for some joint action $\alpha(\vec{u})$, i.e., $s' \in \tau(s, \alpha(\vec{u}))$. When no risk of confusion arises, we will use the same symbols to denote parameters and ground values for actions. An *$s$-run* $r$ is an infinite sequence $s^0 \to s^1 \to \cdots$, with $s^0 = s$. For $n \in \mathbb{N}$, we set $r(n) = s^n$. A state $s'$ is *reachable* from $s$ iff $s' = r(i)$ for some $s$-run $r$ and $i \geq 0$. In what follows we assume that the transition relation $\to$ is serial. This can be ensured w.l.o.g. by assuming that each agent has a $\mathsf{skip}$ action enabled at each local state. Further, we define $\mathcal{R}$ as the set of states reachable from any initial state $s_0 \in I$,

i.e., $\mathcal{R} = \{s \in \mathcal{S} \mid s \text{ is reachable from } s_0, \text{ for some } s_0 \in I\}$. Since the domain $X$ may be infinite, the set $\mathcal{R}$ of reachable states is also infinite in principle. Indeed, in the general case our ODAN are infinite-state systems. Finally, for technical reasons we will refer to the *global* database schema $\mathcal{D}_s = \mathcal{D}_0 \cup \cdots \cup \mathcal{D}_n$ of a state $s = \langle l_0, \ldots, l_n \rangle$. Hence, every state $s$ is associated with the $\mathcal{D}_s$-instance $D_s \in \mathcal{D}_s(X)$ such that $D_s(P) = \bigcup_{i \leq n} l_i(P)$, for $P \in \mathcal{D}_s$, that is, we assume that each agent has a truthful, yet limited, view of the global database $\mathcal{D}_s$. In particular, $adom(D_s)$ is equal to $adom(s)$. Notice that for every $s \in \mathcal{S}$, there is a unique instance $D_s$, while the converse is not true in general. Also, the disjoint union $s \oplus s'$ is defined as $\langle l_0 \oplus l'_0, \ldots, l_n \oplus l'_n \rangle$.

## 2.2 The Specification Language FO-CTL

We now introduce a formal language to specify properties of interest of open dynamic agent networks. The presence of data in ODAN calls for the use of first-order logic, whereas temporal operators are needed to account for the system's evolution. Hereafter we consider a set *Var* of *individual variables* and the database schema $\mathcal{D} = \bigcup_{type\ T} \mathcal{D}_T$.

Definition 9 (FO-CTL). *The FO-CTL formulas over the database schema $\mathcal{D}$ are defined in BNF as follows:*

$$\varphi \quad ::= \quad P(\vec{x}) \mid x = y \mid \neg\varphi \mid \varphi \to \varphi \mid \forall x\varphi \mid AX\varphi \mid A\varphi U\varphi \mid E\varphi U\varphi$$

*where $x, y \in Var$, $P \in \mathcal{D}$, and $\vec{x}$ is a q-tuple of variables.*

The language FO-CTL is a first-order extension of the propositional temporal logic CTL. The temporal formulas $AX\varphi$ and $A\varphi U\varphi'$ (resp. $E\varphi U\varphi'$) are read as "for all runs, next $\varphi$" and "for every (resp. some) run, $\varphi$ until $\varphi'$". Free and bound variables are defined as standard, as well as formulas $EX\varphi$, $AF\varphi$, $AG\varphi$, $EF\varphi$, and $EG\varphi$. We write $\phi(\vec{x})$ to denote that the free variables of $\phi$ are among $\vec{x} = x_1, \ldots, x_n$. Notice that we use the same symbols to refer to individual variables and action parameters, the context will disambiguate. The present language can be enriched with constants for individuals. Since such an enhanced framework does not require significant new formal results, while making the notation more cumbersome, we consider only variables as individual terms. In the following we consider also first-order non-modal logic, as defined by the following syntax:

$$\varphi \quad ::= \quad P(\vec{x}) \mid x = y \mid \neg\varphi \mid \varphi \to \varphi \mid \forall x\varphi$$

To define the satisfaction of an FO-CTL formula on an ODAN, we introduce the notion of an *assignment* $\sigma : Var \mapsto X$. We denote by $\sigma^x_u$ the assignment such that *(i)* $\sigma^x_u(x) = u$; and *(ii)* $\sigma^x_u(x') = \sigma(x')$ for every $x'$ different from $x$.

Definition 10 (Semantics of FO-CTL). *We define whether an ODAN $\mathcal{P}$* satisfies *a formula $\varphi$ in a state $s$ according to assignment $\sigma$, or $(\mathcal{P}, s, \sigma) \models \varphi$, as follows:*

$(\mathcal{P}, s, \sigma) \models P(\vec{x})$ *iff* $\langle \sigma(x_1), \ldots, \sigma(x_q) \rangle \in D_s(P)$
$(\mathcal{P}, s, \sigma) \models x = y$ *iff* $\sigma(x) = \sigma(y)$
$(\mathcal{P}, s, \sigma) \models N(x, y)$ *iff* $\sigma(y) \in l_{\sigma(x)}(N)$
$(\mathcal{P}, s, \sigma) \models \neg\varphi$ *iff* $(\mathcal{P}, s, \sigma) \not\models \varphi$
$(\mathcal{P}, s, \sigma) \models \varphi \to \varphi'$ *iff* $(\mathcal{P}, s, \sigma) \not\models \varphi$ *or* $(\mathcal{P}, s, \sigma) \models \varphi'$
$(\mathcal{P}, s, \sigma) \models \forall x\varphi$ *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 for some $k \geq 0$,*
*$(\mathcal{P}, r(k), \sigma) \models \varphi'$, 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 for some $k \geq 0$,*
*$(\mathcal{P}, r(k), \sigma) \models \varphi'$, and for all $j$,*
*$0 \leq j < k$ implies $(\mathcal{P}, r(j), \sigma) \models \varphi$*

A formula $\varphi$ is *true* at $s$, or $(\mathcal{P}, s) \models \varphi$, if $(\mathcal{P}, s, \sigma) \models \varphi$ for all assignments $\sigma$; $\varphi$ is *true* in $\mathcal{P}$, or $\mathcal{P} \models \varphi$, if $(\mathcal{P}, s_0) \models \varphi$ for all $s_0 \in I$. Notice that in Def. 10 we adopt an *active domain* semantics, where quantifiers range over the active domain $adom(s)$ of a state $s$. This is a standard assumption in database theory that has been lifted to data-aware systems [3, 13]. Also, observe the particular clause for formulas of the form $N(x, y)$, according to the intution that $N(x, y)$ describes an edge from $\sigma(x)$ to $\sigma(y)$.

Finally, we state the model checking problem for ODAN with respect to the specification language FO-CTL.

Definition 11 (Model Checking Problem). *Given an ODAN $\mathcal{P}$ and an FO-CTL formula $\varphi$, determine whether for every initial state $s_0 \in I$, $(\mathcal{P}, s_0, \sigma_0) \models \varphi$ for some assignment $\sigma_0$.*

In the statement of the model checking problem we suppose that the transition function $\tau$ is given as some sort of computable function. Also, we assume finitary descriptions for the set $I$ of initial states and the domain of interpretation. As it will be apparent in Section 3, these requirements are normally fulfilled in cases of interest. Also, for most of relevant applications the specification $\varphi$ is an FO-CTL sentence, with no free variables. Hence, the model checking problem reduces to determine whether $\mathcal{P} \models \varphi$, as the satisfaction of FO-CTL sentences does not depend on bound variables.

Model checking general data-aware systems is known to be undecidable [10]. In [2, 3] the same problem is proved to be decidable for *bounded* and *uniform* systems. However, the set of agents is assumed to be fixed at design time. In the next section we illustrate and motivate the formal machinery introduced so far by means of a diffusion phenomenon.

# 3. THE SIR MODEL

In this section we show how an influential network diffusion model can be handled in our framework. The model at issue is the SIR (susceptible-infected-recovered) model for epidemics, i.e., the diffusion of a diseases in a population (see [11, Ch. 21] or [14, Ch. 7] for textbook introductions). In the SIR model a population of individuals is liable to go through three different stages during an epidemic. First, each agent is *susceptible* to be infected; she may actually get *infected* at a certain point; and finally she will eventually *recover*.[2] The SIR model typically assumes a finite population and a static network structure. In the following we show how the SIR model can be captured and generalized to an open and dynamic system within the framework of ODAN and how interesting properties of SIR can be expressed in FO-CTL. In Section 5 we will briefly discuss how the techniques developed in this paper can be deployed to verify the SIR model against FO-CTL specifications. We start with the standard case in which the topology of the network is assumed to be fixed and it does not change over time.

## 3.1 Static SIR Model

In the static SIR model the network topology is assumed to be fixed. This case is handy to introduce basic notation and terminology. More formally, we consider a unique

[2]Many variants of such model are of course possible, and many are indeed studied. For instance, the SIS model assumes a cycle from susceptibility to infection and back, without any recovered state.

type of agents with names in $Ag$. Also, the interpretation domain $X$ is equal to $Ag$, i.e., for the time being we are only interested in facts concerning our agents. An *agent* in the static SIR model is a tuple $a = \langle \mathcal{D}_a, Act_a, Pr_a \rangle$ such that

- $\mathcal{D}_a = \{Sus/1, Inf/1, Rec/1, N/1\}$, where the intuitive meaning of *Sus* (resp. *Inf*, *Rec*) is that an agent is susceptible (resp. infected, recovered), while $N$ is the network predicate expressing the proximity/contact relation between agents;
- $Act_a = \{\mathsf{skip}\}$, i.e., agents can only perform a skip;
- the protocol $Pr_a$ is such that $Pr_a(l) = Act_a(X)$ for all $l \in \mathcal{D}_a(X)$, i.e., the skip action is enabled in any state.

Given the set $Ag$ of agents as defined above, the *static SIR ODAN* is defined as the tuple $\mathcal{P} = \langle Ag, I, \tau \rangle$ such that

- $I$ is the set of states where no agent is in the recovered state, and it is assumed that at least one agent is infected to rule out trivial models. Also, $Sus(b)$ (resp. $Inf(b)$, $Rec(b)$) belongs to $l_a$ only if $b = a$, or $b \in l_a(N)$ and $Sus(b)$ (resp. $Inf(b)$, $Rec(b)$) belongs to $l_b$. Basically, we assume that each agent is exactly in one of the three possible states, and she knows only her local state and at most those of the agents she is directly related to.
- $s' \in \tau(s, \mathsf{skip})$ iff
  1. $Sus(a) \in l_a$, for some $b \in l_a(N)$, $Inf(b) \in l_b$, and either $Sus(a) \in l'_a$ or $Inf(a) \in l'_a$; or
  2. $Inf(a) \in l_a$ and either $Inf(a) \in l'_a$ or $Rec(a) \in l'_a$; or
  3. $Rec(a) \in l_a$ and $Rec(a) \in l'_a$;
  4. the consistency between an agent's information about the agents she is related to and said agents' local states is preserved.

Intuitively, by definition of the static SIR ODAN $\mathcal{P}$, in each moment a necessary but not sufficient condition for getting infected is that at least one related agent is such. Also, each infected agent non-deterministically recovers and remains so everafter. We observe that $\mathcal{P}$ is an infinite-state system as the set $Ag$ of agents is infinite, thus allowing infinitely many initial states in $I$, according to the network topology. Nonetheless, notice that each $s_0 \in I$ defines a finite agent network. This observation is key for the developments in Section 5.

Further, we remark that the definitions of protocol $Pr_a$ and transition function $\tau$ satisfy the constraints on both outlined in Section 2. Specifically, it is trivially true that in isomorphic local states the protocol $Pr_a$ prescribes isomorphic actions (as it always prescribes $\mathsf{skip}$); while $\tau$ returns isomorphic states provided isomorphic states and actions.

The formalisation of the SIR model here provided is extremely simple. More elaborate rules for infection spreading could be considered, especially if probabilities of infection are taken into account, which we abstract from here. Nonetheless, even this basic setting raises verificational issues. Indeed, the static SIR ODAN is an infinite-state system in general, and thus not amenable by standard model checking techniques for finite-state systems.

## 3.2 Dynamic SIR Model

To account for the agents' actions, in the dynamic SIR model agents are allowed to change the structure of the network given their current information on the other agents. Specifically, we suppose that agents disconnect from other agents in the network whenever the latter get infected. This can be seen for instance as a standard quarantine procedure during epidemics. Also in the present case we consider a unique type of agent. Then, an *agent* in the dynamic SIR model is introduced as a tuple $a = \langle \mathcal{D}_a, Act_a, Pr_a \rangle$ such that $\mathcal{D}_a$ is defined as in the static model and

- $Act_a = \{\mathsf{skip}, con(ag), disc(ag)\}$, where the intuitive meaning of $con(ag)$ (resp. $disc(ag)$) is that agent $a$ connects with (resp. disconnects from) agent $ag$;
- the protocol $Pr_a$ is such that $disc(b) \in Pr_a(l_a)$ whenever $b \in l_a(N)$ and $Inf(b) \in l_a$. Moreover, $\{\mathsf{skip}, con(b)\} \subseteq Pr_a(l)$ for all $l_a \in \mathcal{D}_a(X)$.

Notice that an agent can disconnect from another agent whenever she knows that the latter is infected. However, she can always connect to any other agent.

Given a set $Ag$ of agents as defined above, the *dynamic SIR ODAN* is a tuple $\mathcal{P} = \langle Ag, I, \tau \rangle$ where $I$ is defined as in the static SIR ODAN, and

- $s' \in \tau(s, \alpha)$ iff
  1. conditions (1)-(4) in the definition of $\tau$ for the static SIR ODAN hold;
  2. $\alpha_a = con(b)$, $l'_a(N) = l_a(N) \cup \{b\}$, and $l'_a = l_a \cup \{P(b) \mid P \in \mathcal{D}_b, P(b) \in l'_b\}$;
  3. $\alpha_a = dis(b)$, $l'_a(N) = l_a(N) \setminus \{b\}$, and $l'_a = l_a \setminus \{P(b) \mid P \in \mathcal{D}_b, P(b) \in l_b\}$.

We remark that when agent $a$ connects to agent $b$, the health status of the latter become part of the local state of the former, as assumed to be the case already in the definition of the static model. Moreover, $b$ may be a new agent, not appearing in $adom(s)$ nor $ag(s)$. If this is the case, the health status of $b$ is initialised to *susceptible* for definiteness. Since agents can "be born" and "die" at run time, the lengths of tuples $s$ and $s'$ in the definition of $\tau$ will be different in general. This formal feature, which reflects the open and dynamic nature of ODAN, is in marked contrast with the standard literature on interpreted systems [17, 12, 23].

In the formalisation provided each agent disconnects or connects to a single agent at a time. We can generalize the model by allowing multiple connections/disconnections. Moreover, we may suppose that an agent not only sees the health status of directly reachable partners, but possibly of agents within a distance of $k$ steps, for some $k \in \mathbb{N}$. These extra features, that can have an impact on the properties of the agent network, can be seamlessly modelled on ODAN.

## 3.3 Dynamic SIR Model with Health Workers

The third version of the SIR model is intended to account for the contribution of different *types* of agents to the diffusion process. For the case in hand, besides the standard agents of the static and dynamic SIR models, we introduce a new type of agent to mimick health workers. Containement policies are key to prevent the spreading of diseases. However, it is extremely challenging to assess the impact of such policies in real-life scenarios. In this endeavour formal verification techniques are surely of help (cf. [20]).

A *health agent* is defined as a tuple $h = \langle \mathcal{D}_h, Act_h, Pr_h \rangle$ where $\mathcal{D}_h$ and $Act_h$ are the same as for standard agents; while the protocol $Pr_h$ is such that

- $disc(b) \notin Pr_h(l_h)$ whenever $Inf(b) \in l_h$, but $disc(b) \in Pr_h(l_h)$ whenever $Inf(h) \in l_h$ and $b \in l_h(N)$.

Thus, differently from standard agents, health agents are not allowed to disconnect from infected agents. Nonetheless, they disconnect once they also become infected. As a result, health workers behave differently from standard agents as long as they are not infected. This modelling choice reflects the idea that health agent are also susceptible to infection.

Given sets $Ag$ of agents and $Ag_h$ of health agents, the *dynamic SIR ODAN with recovery threshold* $k \in \mathbb{N}$ is defined as a tuple $\mathcal{P} = \langle Ag \cup Ag_h, I, \tau \rangle$ such that the set $I$ of initial states and the transition function $\tau$ are defined as for the dynamic model, but for the following clause:

- $s' \in \tau(s, \alpha)$ iff $Inf(a) \in l_a$, $Rec(a) \in l'_a$, and $|l_a(N) \cap Ag_h| > k$.

The ODAN $\mathcal{P}$ formalises, even though naively, the contribution provided by health workers to patient recovery. That is, we suppose that when an infected agent is in contact with more than $k$ health workers, for some threshold $k \in \mathbb{N}$, then she is guaranteed to recover. Notice that, by definition of the procol $Pr_h$, these health workers are not infected and indeed capable of doing their job. By tweaking the threshold $k$ we can simulate stricter or milder policies. Also, ODAN are expressive enough to accommodate communication amongst health agents in order to treat infected patients.

To summarize, the framework of ODAN is rich enough to model various assumptions on the network agents, including available actions, behaviour, and internal state. For systems such as the above SIR ODAN we will show in Section 5 that it is possible to develop verification techniques, even if we are dealing with infinite-state systems.

## 3.4 Specifications

We now consider specifications in FO-CTL that express interesting properties of the SIR models illustrated above. Firstly, a property that it is natural to consider is whether each agent goes through the susceptible-infected-recovered cycle. This can be easily expressed as an FO-CTL formula:

$$AG\ \forall x A(Sus(x) U A(Inf(x) U Rec(x))) \tag{1}$$

The same property can be recast by using the *weak until* operator defined as $A\phi W \phi' \equiv \neg E(\neg\phi' U(\neg\phi \wedge \neg\phi'))$:

$$AG\ \forall x A(Sus(x) W A(Inf(x) W Rec(x))) \tag{2}$$

We anticipate that, while (1) does not hold for some executions of the SIR model, its *weak until* version (2) is indeed satisfied.

Secondly, we might want to verify topological properties of the agent network, such as whether every agent either remains susceptible or will eventually become infected if she is continuously in contact with someone infected. This can be expressed in FO-CTL as follows:

$$AG\ \forall x(AG Sus(x) \vee E(\exists y(Inf(y) \wedge N(x,y)) U Inf(x))) \tag{3}$$

Another property of interest is whether each agent eventually disconnects from an infected neighbour:

$$AG\ \forall x, y(Inf(y) \wedge N(x,y) \rightarrow AF\neg N(x,y)) \tag{4}$$

Finally, we might want to assess the impact of health agents in the third version of the SIR model, for instance by evaluating whether an infected agent will always eventually recover if she is in contact with at least $k$ health agents:

$$AG\ \forall x(\exists^{>k} y(Inf(x) \wedge Ag_h(y) \wedge N(x,y)) \rightarrow AF Rec(x)) \tag{5}$$

In (5) the bounded quantifier $\exists^{>k}$ can be defined by using equality $=$, while $Ag_h$ is introduced as a new predicate in the language, true only of health agents.

Our aim in the present paper is to develop techniques to model check FO-CTL specifications as these on open dynamic agent networks. We remarked above that the SIR ODAN here provided are infinite-state systems in general, so their verification cannot be straightforwardly tackled by using techniques developed for finite-state systems. In the next section we prove novel results that make ODAN verification feasible.

# 4. BISIMULATION

In Section 2 we stated that model checking ODAN against FO-CTL specifications is undecidable in general. Clearly, for the verification of open dynamic agent networks it is crucial to isolate syntactic and semantical fragments with a decidable model checking problem. Hereafter we identify a rather natural subclass of ODAN that we call *bounded*. Bounded ODAN admit finite bisimilar abstractions, so that the verification of FO-CTL properties can be conducted on the latter, rather than on the original infinite-state system. We will discuss this in some detail in Section 5. To introduce finite abstractions we first present bisimulations and show that bisimilar ODAN satisfy the same FO-CTL formulas. The results presented in this section build on [2, 3]. However, the setting here is fundamentally different, as we consider networks where agents can join and leave at run time.

In the rest of the section we let $\mathcal{P} = \langle Ag, U, I, \tau \rangle$ and $\mathcal{P}' = \langle Ag', U', I', \tau' \rangle$ be two ODAN and assume that $s = \langle l_0, \ldots, l_n \rangle \in \mathcal{R}$ and $s' = \langle l'_0, \ldots, l'_n \rangle \in \mathcal{R}'$. First of all, according to Def. 8 isomorphic states have the same relational structure. However, they do not necessarily satisfy the same first-order formulas, as satisfaction depends also on the values assigned to free variables. To account for this, we introduce the following notion.

Definition 12 (Equivalent assignments). *Given states $s \in \mathcal{R}$ and $s' \in \mathcal{R}'$, and a set $V \subseteq Var$ of variables, assignments $\sigma : Var \mapsto X$ and $\sigma' : Var \mapsto X'$ are* equivalent for $V$ w.r.t. $s$ and $s'$ *iff for some bijection $\gamma : adom(s) \cup ag(s) \cup \sigma(V) \mapsto adom(s') \cup ag(s') \cup \sigma'(V)$,* (i) *the restriction $\gamma|_{adom(s) \cup ag(s)}$ is a witness for $s \simeq s'$; and* (ii) *$\sigma'|_V = \gamma \circ \sigma|_V$.*

Intuitively, equivalent assignments preserve agent types, the (in)equalities of the variables in $V$, as well as the active elements in $s, s'$ up to renaming. Moreover, two assignments are *equivalent for an FO-CTL formula $\varphi$* (omitting states $s$ and $s'$ whenever clear by the context) if these are equivalent for its free variables $fr(\varphi)$.

We now state the following standard result on the preservation of first-order (non-modal) formulas.

Lemma 1. *Given isomorphic states $s \in \mathcal{R}$ and $s' \in \mathcal{R}'$, an FO formula $\varphi$, and assignments $\sigma$ and $\sigma'$ equivalent for $\varphi$, we have that $(D_s, \sigma) \models \varphi$ iff $(D_{s'}, \sigma') \models \varphi$.*

As a result, isomorphic states cannot be distinguished by FO formulas (whenever equivalent assignments are considered). We make use of this observation to define bisimulations on ODAN. In particular, plain bisimulations are known to be satisfaction preserving in a modal propositional setting [4]. Hereafter we explore under which conditions this applies to ODAN as well.

Definition 13 (Simulation). *A relation $R$ on $\mathcal{R} \times \mathcal{R}'$ is a* simulation *if $\langle s, s' \rangle \in R$ implies* (i) *$s \simeq s'$; and* (ii) *for every $t \in \mathcal{R}$, if $s \rightarrow t$ then for some $t' \in \mathcal{R}'$, $s' \rightarrow t'$, $s \oplus t \simeq s' \oplus t'$, and $\langle t, t' \rangle \in R$;*

A state $s' \in \mathcal{R}'$ *simulates* $s \in \mathcal{R}$ iff $\langle s, s' \rangle \in R$ for some simulation $R$. Notice that similar states are isomorphic by

condition 13.*(i)* above. Simulations can naturally be extended to bisimulations as follows.

DEFINITION 14 (BISIMULATION). *A relation $B$ on $\mathcal{R} \times \mathcal{R}'$ is a* bisimulation *iff both $B$ and $B^{-1} = \{\langle s', s\rangle \mid \langle s, s'\rangle \in B\}$ are simulations.*

Two states $s \in \mathcal{R}$ and $s' \in \mathcal{R}'$ are *bisimilar*, or $s \approx s'$, iff $\langle s, s'\rangle \in B$ for some bisimulation $B$. It can be shown that $\approx$ is the largest bisimulation, and an equivalence relation, on $\mathcal{R} \cup \mathcal{R}'$. Finally, $\mathcal{P}$ and $\mathcal{P}'$ are *bisimilar*, or $\mathcal{P} \approx \mathcal{P}'$, iff for every $s_0 \in I$, $s_0 \approx s_0'$ for some $s_0' \in I'$, and for every $s_0' \in I'$, $s_0 \approx s_0'$ for some $s_0 \in I$.

We remark that for general data-aware systems, bisimilarity is not sufficient to preserve FO-CTL formulas (please refer to [3] for a proof). This is in marked constrast with the modal propositional case. However, we show that ODAN, being *uniform*, admit bisimulations that preserve FO-CTL.

LEMMA 2 (UNIFORMITY). *Every ODAN $\mathcal{P}$ is* uniform*, that is, for every $s, t, s' \in \mathcal{R}$, $t' \in \mathcal{S}$, if $s \to t$ and $s \oplus t \simeq s' \oplus t'$, then $s' \to t'$.*

Intuitively, uniformity expresses that if $t$ can be reached by executing the ground action $\alpha(\vec{u})$ in $s$, and we uniformly replace the element $v$ with $v'$ in $s$, $\vec{u}$ and $t$, obtaining $s'$, $\vec{u}'$ and $t'$ respectively, then $t'$ can be reached by executing $\alpha(\vec{u}')$ in $s'$. This feature is a consequence of the invariance of the protocol and transition functions w.r.t. the actual data content of states and actions, as formalised in the requirements on ODAN in Section 2.

Now we state some lemmas, which are needed to prove the main preservation result Theorem 6. A further distinctive feature of ODAN is that isomorphic states are bisimilar.

LEMMA 3. *For every ODAN $\mathcal{P}$, for every $s, s' \in \mathcal{R}$, $s \simeq s'$ implies $s \approx s'$.*

By Lemma 3 the submodels generated by isomorphic states are bisimilar.

The next two results guarantee that, under appropriate constraints, bisimulations preserve assignments equivalence.

LEMMA 4. *Consider bisimilar ODAN $\mathcal{P}$ and $\mathcal{P}'$, bisimilar states $s \in \mathcal{R}$ and $s' \in \mathcal{R}'$, an FO-CTL formula $\varphi$, and assignments $\sigma$ and $\sigma'$ equivalent for $\varphi$ w.r.t. $s$ and $s'$.*

*For every $t \in \mathcal{R}$, if* (i) *$s \to t$,* (ii) *$|X'| \geq |adom(s) \cup ag(s) \cup adom(t) \cup ag(t) \cup \sigma(fr(\varphi))|$, and* (iii) *for every agent type $T$, $|Ag'_T| \geq |ag_T(s) \cup ag_T(t) \cup \sigma(fr(\varphi))|$, then for some $t' \in \mathcal{R}'$,* (i) *$s' \to t'$,* (ii) *$t \approx t'$, and* (iii) *$\sigma$ and $\sigma'$ are equivalent for $\varphi$ w.r.t. $t$ and $t'$.*

The proof of Lemma 4 makes essential use of Lemma 3 and uniformity. Thus, every time the cardinality constraints in Lemma 4 are satisfied, it is possible to extend equivalent assignments between pairs of bisimilar global states to their successors.

The next result relies on and generalises Lemma 4 to runs.

LEMMA 5. *Consider bisimilar ODAN $\mathcal{P}$ and $\mathcal{P}'$, bisimilar states $s \in \mathcal{R}$ and $s' \in \mathcal{R}'$, an FO-CTL formula $\varphi$, and assignments $\sigma$ and $\sigma'$ equivalent for $\varphi$ w.r.t. $s$ and $s'$.*

*For every $s$-run $r$ in $\mathcal{P}$, if for all $i \geq 0$,* (i) *$|X'| \geq |adom(r(i)) \cup ag(r(i)) \cup adom(r(i+1)) \cup ag(r(i+1)) \cup \sigma(fr(\varphi))|$, and* (ii) *$|Ag'_T| \geq |ag_T(r(i)) \cup ag_T(r(i+1)) \cup \sigma(fr(\varphi))|$ for every agent type $T$, then for some $s'$-run $r'$ in $\mathcal{P}'$, for all $i \geq 0$,* (i) *$r(i) \approx r'(i)$;* (ii) *$\sigma$ and $\sigma'$ are equivalent for $\varphi$ w.r.t. $r(i)$ and $r'(i)$; and* (iii) *if $r(i) \to r(i+1)$ then $r'(i) \to r'(i+1)$.*

By Lemma 5, if we have a sufficient number of elements in $X'$ and of agents in each $Ag'_T$, we can simulate the execution of an $s$-run by constructing a corresponding $s'$-run.

We finally prove that FO-CTL formulas cannot distinguish between bisimilar ODAN ($var(\varphi)$ is the set of variables in $\varphi$).

THEOREM 6. *Consider bisimilar ODAN $\mathcal{P}$ and $\mathcal{P}'$, bisimilar states $s \in \mathcal{R}$ and $s' \in \mathcal{R}'$, an FO-CTL formula $\varphi$, and assignments $\sigma$ and $\sigma'$ equivalent for $\varphi$ w.r.t. $s$ and $s'$. If*

1. *for every $s$-run $r$, for every $k \geq 0$,* (i) *$|X'| \geq |adom(r(k)) \cup ag(r(k)) \cup adom(r(k+1)) \cup ag(r(k+1)) \cup \sigma(fr(\varphi))| + |var(\varphi) \setminus fr(\varphi)|$, and* (ii) *$|Ag'_T| \geq |ag_T(r(k)) \cup ag_T(r(k+1)) \cup \sigma(fr(\varphi))| + |var(\varphi) \setminus fr(\varphi)|$ for every type $T$;*
2. *for every $s'$-run $r'$, for every $k \geq 0$,* (i) *$|X| \geq |adom(r'(k)) \cup ag(r'(k)) \cup adom(r'(k+1)) ag(r'(k+1)) \cup \sigma'(fr(\varphi))| + |var(\varphi) \setminus fr(\varphi)|$, and* (ii) *$|Ag_T| \geq |ag_T(r'(k)) \cup ag_T(r'(k+1)) \cup \sigma'(fr(\varphi))| + |var(\varphi) \setminus fr(\varphi)|$ for every type $T$;*

*then $(\mathcal{P}, s, \sigma) \models \varphi$ iff $(\mathcal{P}', s', \sigma') \models \varphi$.*

As a consequence of Theorem 6, bisimilar states satisfy the same FO-CTL formulas for equivalent assignments, whenever cardinality constraints (1) and (2) are satisfied. The proof of this result makes essential use of Lemma 5.

We now apply Theorem 6 to the model checking problem for ODAN. First of all, we introduce bounded ODAN.

DEFINITION 15 (BOUNDED ODAN). *An ODAN $\mathcal{P}$ is $b$-*bounded*, for $b \in \mathbb{N}$, iff for all $s \in \mathcal{R}$, $|adom(s) \cup ag(s)| \leq b$.*

An ODAN $\mathcal{P}$ is *bounded* iff it is $b$-bounded for some $b \in \mathbb{N}$. Notice that bounded ODAN are still infinite-state systems in general. Hereafter let $\sup_{s \in \mathcal{R}}\{|adom(s) \cup ag(s)|\}$ be equal to $\infty$ whenever the ODAN $\mathcal{P}$ is unbounded. Similarly for $\sup_{s \in \mathcal{R}}\{|ag_T(s)|\}$.

COROLLARY 7. *Consider bisimilar ODAN $\mathcal{P}$ and $\mathcal{P}'$, and an FO-CTL formula $\varphi$. If*

1. *$|X'| \geq 2 \sup_{s \in \mathcal{R}}\{|adom(s) \cup ag(s)|\} + |var(\varphi)|$ and $|Ag'_T| \geq 2 \sup_{s \in \mathcal{R}}\{|ag_T(s)|\} + |var(\varphi)|$*
2. *$|X| \geq 2 \sup_{s' \in \mathcal{R}'}\{|adom(s') \cup ag(s')|\} + |var(\varphi)|$ and $|Ag_T| \geq 2 \sup_{s' \in \mathcal{R}'}\{|ag_T(s')|\} + |var(\varphi)|$*

*then $\mathcal{P} \models \varphi$ iff $\mathcal{P}' \models \varphi$.*

Corollary 7 shows that ODAN can in principle be verified by model checking a bisimilar system. Most importantly, this applies to any infinite ODAN $\mathcal{P}$ as well. Hence, by this result we can model check the corresponding, possibly finite $\mathcal{P}'$, as long as $X'$ is sufficiently large for $\mathcal{P}'$ to bisimulate $\mathcal{P}$.

In the next section we show that finite abstractions can indeed be constructed for bounded ODAN, thus allowing for the verification of systems such as those of Section 3.

# 5. FINITE ABSTRACTION

In this section we state sufficient conditions to reduce the model checking problem for an infinite ODAN to the verification of a finite system. The main result is Theorem 8, which guarantees that boundedness is sufficient to obtain finite bisimilar abstractions that preserve FO-CTL formulas.

## 5.1 Reduction to finite-system verification

In Section 4 we specified that an ODAN is $b$-bounded if no active domain in its reachable state space contains more than

$b$ distinct elements. Moreover, no more than $b$ agents can be active at the same time. Observe that bounded ODAN may still contain infinitely many states, all bounded by some $b$. So, bounded ODAN are infinite-state systems in general, with a non-trivial model checking problem.

In order to verify ODAN we introduce abstractions in a modular manner by first defining abstract agents.

DEFINITION 16 (ABSTRACT AGENT). *Let $a = \langle \mathcal{D}, Act, Pr\rangle \in Ag_T$ be an agent of type $T$ defined on a countable interpretation domain $X = U \cup Ag$. Given a countable set $X' = U' \cup Ag'$ of elements, the* abstract agent *$a' \in Ag'_T$ is a tuple $\langle \mathcal{D}', Act', Pr'\rangle$ on $X'$ such that* (i) *$\mathcal{D}' = \mathcal{D}$;* (ii) *$Act' = Act$; and* (iii) *$Pr'$ is the smallest function defined as*

- *if $\alpha(\vec{u}) \in Pr(l)$, $l' \in \mathcal{D}'(X')$ and $l' \simeq l$ for some witness $\iota$, then $\alpha(\iota(\vec{u})) \in Pr'(l')$.*

*Given a set $Ag_T$ of agents, let $Ag'_T$ be the set of the corresponding abstract agents.*

We remark that $a'$, as defined in Def. 16, is indeed an agent of type $T$ according to Def. 4. In particular, the protocol $Pr'$ is well-defined provided $Pr$ is, and it satisfies the assumption on protocols by definition. We now present abstractions.

DEFINITION 17 (ABSTRACTION). *Let $\mathcal{P} = \langle Ag, U, I, \tau\rangle$ be an ODAN, and $Ag'$ the set of abstract agents given as in Def. 16. The ODAN $\mathcal{P}' = \langle Ag', U', I', \tau'\rangle$ is an* abstraction *of $\mathcal{P}$ iff* (i) *$I' = \{s'_0 \in \mathcal{D}'(X') \mid s'_0 \simeq s_0$ for some $s_0 \in I\}$, and* (ii) *$\tau'$ is the smallest function defined as follows*

- *if $s \xrightarrow{\alpha(\vec{u})} t$ in $\mathcal{P}$, $s', t' \in \mathcal{D}'(X')$, and $s \oplus t \simeq s' \oplus t'$ for some witness $\iota$, then $s' \xrightarrow{\alpha(\iota(\vec{u}))} t'$.*

Notice that $\mathcal{P}'$ is indeed an ODAN as it satisfies the relevant conditions on protocols and transitions in Def. 7. Also, by varying $X'$ we can obtain abstractions of different cardinalities. In particular, we are interested in finite ones.

The following result guarantees that for every bounded ODAN there exists a bisimilar abstraction, provided that the latter is built over a sufficiently large interpretation domain. In the following we assume that, for a bound $b \in \mathbb{N}$, $N_b$ is the maximum numbers of parameters contained in any parametric joint actions, i.e., $N = b \cdot \max_{\{\alpha(\vec{x}) \in Act_T, type\ T\}}\{|\vec{x}|\}$.

THEOREM 8. *Consider a bounded ODAN $\mathcal{P}$ over an infinite interpretation domain $X$, an FO-CTL formula $\varphi$, and an interpretation domain $X'$. If* (i) *$|X'| \geq 2b + \max\{|var(\varphi)|, N_b\}$, and* (ii) *for every type $T$, $|Ag'_T| \geq 2b + \max\{|var(\varphi)|, N_b\}$, then there exists a bisimilar abstraction $\mathcal{P}'$ of $\mathcal{P}$ over $X'$ such that $\mathcal{P} \models \varphi$ iff $\mathcal{P}' \models \varphi$*

We remark that each $Ag'_T$ and $X'$ in Theorem 8 might as well be finite. So, by using a sufficient number of abstract agents and values, we can in principle reduce the model checking problem for infinite-state ODAN to the verification of a finite abstraction. Specifically, we obtain the following corollary to Theorem 8.

COROLLARY 9. *Given a bounded ODAN $\mathcal{P}$ over an infinite interpretation domain $X$, and an FO-CTL formula $\varphi$, there exists an abstract ODAN $\mathcal{P}'$ over a finite interpretation domain $X'$ such that $\varphi$ holds in $\mathcal{P}$ iff $\mathcal{P}'$ satisfies $\varphi$.*

As a consequence of Corollary 9, we can verify an infinite-state, bounded ODAN, by model checking its finite, bisimilar abstraction.

For the time being we do not discuss efficient model checking procedure for finite ODAN, as it is beyond the scope of the paper. We only remark that the state space of ODAN is usually exponential in the number of agents and data. Thus, a major challenge for these systems is to develop efficient model checking algorithms. We leave this topic for future research and remind here that the main motivation and contribution of this paper is to show that under specific conditions (namely, boundedness) we can reduce the model checking problem for an infinite-state ODAN to the verification of its finite abstraction.

## 5.2 Model Checking SIR Models

In this section we outline how the properties listed in Section 3.4 can be verified on the SIR models provided in Section 3. We start by remarking the following.

Firstly, when studying the evolution of a diffusion phenomenon, such as epidemics, we can safely assume that the population will not exceed a certain bound at any given time, possibly determined by the resources of the environment. Hence, even though individual agents can join or leave the ODAN according to the cycle of births and deaths, their number is supposed to never exceed a given bound $b \in \mathbb{N}$.

Secondly, the computational capabilities of any real-life system are limited at each time-point. Specifically, system memories are capable of storing only a finite amount of data at each moment, even though these data can constantly change at run time.

Given the above, we can safely assume that the number of agents in any single state of the SIR ODAN in Section 3 is bounded by a value $b \in \mathbb{N}$. We can enforce this by modifying the definition of the SIR ODAN, by requiring that the initial states contain at most $b$ agents, i.e., for every $s_0 \in I$, $|ag(s_0)| \leq b$. Further, we modify the definition of the transition function $\tau$ by specifying that recovered agents are discarded from the network and that existing agents can connect to new agents as long as the bound $b$ is not met. Indeed, recovered agents no longer play an active role in the evolution of the SIR model, so they can be safely discarded.

Therefore, according to Theorem 8, to model check specification (1) in Section 3.4 on the dynamic SIR ODAN it is sufficient to consider a domain $Ag'$ of agents of cardinality $|Ag'| = 2b + \max\{|var(\varphi)|, N_b\} = 3b$. The agents in $Ag'$ generate a dynamic SIR ODAN $\mathcal{P}'$ that is bisimilar to the original concrete system $\mathcal{P}$ defined on some infinite set $Ag$ of agents of the same type. In particular, $\mathcal{P}'$ is finite, so we can effectively verify specification (1) on $\mathcal{P}'$ and then transfer the result to $\mathcal{P}$.

# 6. CONCLUSIONS

In this paper we introduced a data-aware model for open dynamic agent networks (ODAN) and investigated the verification of first-order temporal specifications on such structures. The main technical result consists in proving that, under the boundedness assumption, the model checking problem for ODAN against FO-CTL is decidable. We also showed that the framework of ODAN is expressive enough to capture non-probabilistic diffusion phenomena such as epidemics in open and dynamic variants of the SIR model.

Future work should focus on exploring the complexity of the model-checking problem for ODAN possibly restricting the first-order expressivity of the underlying logic to a suitable modal fragment.

# 7. REFERENCES


[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] F. Belardinelli, A. Lomuscio, and F. Patrizi. An Abstraction Technique for the Verification of Artifact-Centric Systems. In *Proc. of the 13th International Conference on Principles of Knowledge Representation and Reasoning (KR'12)*, pages 319 – 328, 2012.

[3] F. Belardinelli, F. Patrizi, and A. Lomuscio. Verification of agent-based artifact systems. *Journal of Artificial Intelligence Research*, 51:333–77, 2014.

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

[5] Z. Christoff and J. U. Hansen. A two-tiered formalization of social influence. In *Proceedings of LORI'13*, number 8196 in LNCS, pages 68–81. Springer, 2013.

[6] Z. Christoff and J. U. Hansen. A logic for diffusion in social networks. *Journal of Applied Logic*, 13(1):48–77, 2015.

[7] D. Cohn and R. Hull. Business Artifacts: A Data-Centric Approach to Modeling Business Operations and Processes. *IEEE Data Eng. Bull.*, 32(3):3–9, 2009.

[8] M. Dam. Model checking mobile processes. *Information and Computation*, 129:35–51, 1996.

[9] A. Deutsch, R. Hull, F. Patrizi, and V. Vianu. Automatic Verification of Data-Centric Business Processes. In *Proc. of ICDT*, 2009.

[10] Alin Deutsch, Liying Sui, and Victor Vianu. Specification and Verification of Data-Driven Web Applications. *J. Comput. Syst. Sci.*, 73(3):442–474, 2007.

[11] D. Easley and J. Kleinberg. *Networks, Crowds, and Markets*. Cambridge University Press, 2010.

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

[13] B. Bagheri Hariri, D. Calvanese, G. De Giacomo, A. Deutsch, and M. Montali. Verification of relational data-centric dynamic systems with external services. In R. Hull and W. Fan, editors, *PODS*, pages 163–174. ACM, 2013.

[14] M. O. Jackson. *Social and Economic Networks*. Princeton University Press, 2008.

[15] S. Khan, R. Makkena, F. McGeary, K. Decker, W. Gillis, and C. Schmidt. A multi-agent system for the quantitative simulation of biological networks. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems (AAMAS'03)*, pages 385–392. ACM, 2003.

[16] R. Milner. The polyadic $\pi$-calculus: A tutorial. Technical Report ECS-LFCS-91-180, Laboratory for the Foundations of Computer Science, Department of Computer Science, University of Edinburgh, 1991.

[17] R. Parikh and R. Ramanujam. Distributed processes and the logic of knowledge. In *Logic of Programs*, pages 256–268, 1985.

[18] M. Rovatsos. Multiagent systems for social computation. In A. Lomuscio, P. Scerri, A. Bazzan, and M. Huhns, editors, *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS14)*. IFAAMAS, 2014.

[19] A. Salehi-Abari and C. Boutilier. Empathetic social choice on social networks. In A. Lomuscio, P. Scerri, A. Bazzan, and M. Huhns, editors, *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS14)*. IFAAMAS, 2014.

[20] G. Santhanam, Y. Suvorov, S. Basu, and V. Honavar. Verifying intervention policies to counter infection propagation over networks: A model checking approach. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*. AAAI Press, 2011.

[21] L. Sless, N. Hazon, M. Wooldridge, and S. Kraus. Forming coalitions and facilitating relationships for completing tasks in social networks. In A. Lomuscio, P. Scerri, A. Bazzan, and M. Huhns, editors, *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS14)*. IFAAMAS, 2014.

[22] H. van Ditmarsch, B. Kooi, and W. van der Hoek. *Dynamic Epistemic Logic*, volume 337 of *Synthese Library Series*. Springer, 2007.

[23] M. Wooldridge. Computationally Grounded Theories of Agency. In *Proc. of ICMAS*, pages 13–22. IEEE Press, 2000.