Verification of Agent-based Artifact Systems: Abstraction Techniques and Decidability Results

Francesco Belardinelli
Laboratoire IBISC, Université d'Evry

Joint work with Alessio Lomuscio
Imperial College London, UK

and Fabio Patrizi
Sapienza Università di Roma, Italy

within the EU funded project ACSI (Artifact-Centric Service Interoperation)

Laboratoire LIP6 – 25 February 2013

Model Checking in one slide

Model checking: technique(s) to **automatically** verify that a system design S satisfies a property P **before** deployment.

More formally, given

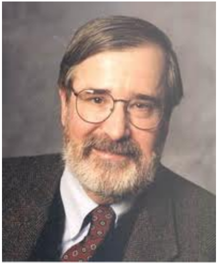
- a model $\mathcal{M}_S$ of a system S
- a formula ϕ_P representing a property P

we check that

$$\mathcal{M}_S \models \phi_P$$

Turing Award 2007

www.acm.org/press-room/news-releases-2008/turing-award-07



(a) E. Clarke (CMU, USA)

(b) A. Emerson (U. Texas, USA)

(c) J. Sifakis (IMAG, F)

- Jury justification

  For their roles in developing model checking into a highly effective verification technology, widely adopted in the hardware and software industries.

Overview

1. Motivation: Artifact Systems as *data-aware* systems

Overview

1. Motivation: Artifact Systems as *data-aware* systems
2. Main task: *formal* verification of infinite-state AS
   - model checking is appropriate for control-intensive applications...
   - ...but less suited for data-intensive applications (data typically ranges over infinite domains) [1].

Overview

1. Motivation: Artifact Systems as *data-aware* systems
2. Main task: *formal* verification of infinite-state AS
   - model checking is appropriate for control-intensive applications...
   - ...but less suited for data-intensive applications (data typically ranges over infinite domains) [1].
3. Key contribution: verification of *bounded* and *uniform* AS is decidable

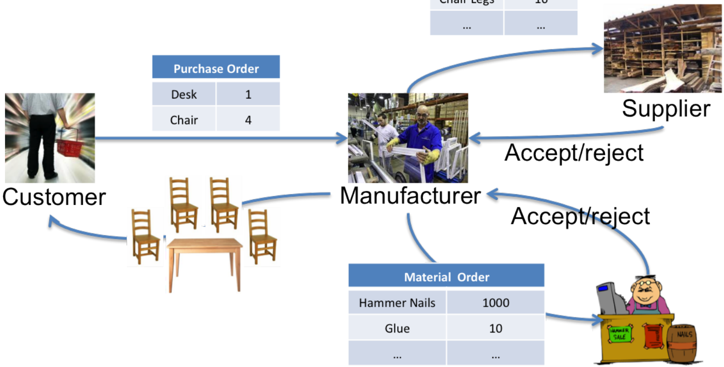
Artifact Systems

Outline

- Recent paradigm for Service-Oriented Computing [2].
- Motto: let's give *data* and *processes* the same relevance!
- *Artifact*: data model + lifecycle
  - (nested) records equipped with actions
  - actions may affect several artifacts
  - evolution stemming from the interaction with other artifacts/external actors
- *Artifact System*: set of interacting artifacts, representing services, manipulated by agents.

Artifact Systems

Order-to-Cash Scenario



Purchase Order	
Desk	1
Chair	4

Material Order	
Desk Legs	4
Chair Legs	16
...	...

Material Order	
Hammer Nails	1000
Glue	10
...	...

Artifact Systems

Data Model

PO			
id	*prod_code*	*offer*	*status*

- *createPO(prod_code, offer)*
- *deletePO(id)*
- *addItemPO(id,itm,qty)*
- ...

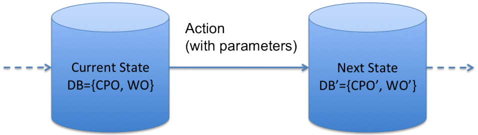
MO			
id	*prod_code*	*price*	*status*

- *createMO(id,price)*
- *deleteMO(id)*
- *addLineItemMO(id,mat,qty)*
- ...

Artifact Systems

Lifecycle

- Agents operate on artifacts.
  - e.g., the Customer sends the Purchase Order to the Manufacturer.
- Actions add/remove artifacts or change artifact attributes.
  - e.g., the PO status changes from *created* to *submitted*.
- The whole system can be seen as a *data-aware* dynamic system.
  - at every step, an action yields a change in the current state.



Research questions

Research questions

1. Which syntax and semantics should we use to specify AS?

Research questions

1. Which syntax and semantics should we use to specify AS?
2. Is verification of AS decidable?

Research questions

1. Which syntax and semantics should we use to specify AS?
2. Is verification of AS decidable?
3. If not, can we identify *relevant* fragments that are reasonably well-behaved?

Research questions

1. Which syntax and semantics should we use to specify AS?
2. Is verification of AS decidable?
3. If not, can we identify *relevant* fragments that are reasonably well-behaved?
4. How can we implement this?

Challenges

Multi-agent systems, but . . .

Challenges

Multi-agent systems, but ...

- ... states have a relational structure,

Challenges

Multi-agent systems, but ...

- ... states have a relational structure,
- data are potentially infinite,

Challenges

Multi-agent systems, but . . .

- . . . states have a relational structure,
- data are potentially infinite,
- state space is infinite in general.

Challenges

Multi-agent systems, but . . .

- . . . states have a relational structure,
- data are potentially infinite,
- state space is infinite in general.

$\Rightarrow$ The model checking problem cannot be tackled by standard techniques.

Artifact Systems

Results

1. *Artifact-centric multi-agent systems* (AC-MAS): formal model for AS.
   Intuition: databases that evolve in time and are manipulated by agents.
2. FO-CTLK as a specification language:

   $$AG \; \forall id, pc \; (\exists \vec{x} \; MO(id, pc, \vec{x}) \rightarrow K_M \; \exists \vec{y} \; PO(id, pc, \vec{y}))$$

   the manufacturer M knows that each MO has to match a corresponding PO.
3. Abstraction techniques and finite interpretation to tackle model checking.
   Main result: under specific conditions MC can be reduced to the finite case.
4. Modelling of declarative GSM systems, developed by IBM, as AC-MAS.

Semantics: Databases

The data model of Artifact Systems is given as a database.

- a *database schema* is a *finite* set $\mathcal{D} = \{P_1/a_1, \dots, P_n/a_n\}$ of predicate symbols P_i with arity $a_i \in \mathbb{N}$.
- a *$\mathcal{D}$-interpretation* on a domain U is a mapping D associating each predicate symbol P_i with a *finite* a_i-ary relation on U.
- the *active domain* $adom(D)$ is the set of all $u \in U$ appearing in D
- the *primed* version of the db schema $\mathcal{D}$ as above is the db schema $\mathcal{D}' = \{P'_1/a_1, \dots, P'_n/a_n\}$.
- *Composition*: $D \oplus D'$ is the $(\mathcal{D} \cup \mathcal{D}')$-interpretation s.t.
  (i) $D \oplus D'(P_i) = D(P_i)$, and
  (ii) $D \oplus D'(P'_i) = D'(P_i)$.

Artifact-centric Multi-agent Systems

Agents

Agents have partial access (views) to the artifact system.

- an *agent* is a tuple $i = \langle \mathcal{D}_i, L_i, Act_i, Pr_i \rangle$ where
  - $\mathcal{D}_i$ is the *local database schema*
  - $L_i \subseteq \mathcal{D}_i(U)$ is the set of *local states*
  - Act_i is the set of *local actions* $\alpha(\vec{x})$ with parameters $\vec{x}$
  - $Pr_i : L_i \mapsto 2^{Act_i}$ is the *local protocol function*
- the *global* database schema is defined as $\mathcal{D} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_n$.
- the setting is reminiscent of the *interpreted systems semantics* for MAS [3],...
- ...but here the local state of each agent is relational.

Intuitively, agents manipulate artifacts and have (partial) access to the information contained in the global db schema $\mathcal{D}$.

Example 1: the Order-to-Cash Scenario

- Agents: Customer, Manifacturer, Supplier.
- Local db schema $\mathcal{D}_C$
  - *Products(prod_code, budget)*
  - *PO(id, prod_code, offer, status)*
- Local db schema $\mathcal{D}_M$
  - *PO(id, prod_code, offer, status)*
  - *MO(id, prod_code, price, status)*
- Local db schema $\mathcal{D}_S$
  - *Materials(mat_code, cost)*
  - *MO(id, prod_code, price, status)*
- Then, $\mathcal{D} = \{Materials, Products, PO, MO\}$.
- Parametric actions can introduce values from an infinite domain U.
  - *createPO(prod_code, offer)* belongs to Act_C.
  - *createMO(prod_code, price)* belongs to Act_M.

Artifact-centric Multi-agent Systems

AC-MAS

Agents are modules that can be composed together to obtain AC-MAS.

- An *AC-MAS* is a tuple $\mathcal{P} = \langle \mathcal{S}, U, D_0, \tau \rangle$ where:
  - $\mathcal{S} \subseteq L_1 \times \cdots \times L_n$ is the set of *reachable global states*
  - U is the *interpretation domain*
  - $D_0 \in \mathcal{S}$ is the *initial global state*
  - $\tau : \mathcal{S} \times Act(U) \mapsto 2^{\mathcal{S}}$ is the *transition function*
- *Temporal transition*: $D \rightarrow D'$ iff there is $\alpha(\vec{u})$ s.t. $D' \in \tau(D, \alpha(\vec{u}))$.
- *Epistemic relation*: $D \sim_i D'$ iff $D_i = D'_i$.
- AC-MAS are infinite-state systems in general.

AC-MAS are first-order temporal epistemic structures.
Hence, FO-CTLK can be used as a specification language.

Syntax: FO-CTLK

- Data call for First-order Logic.
- Evolution calls for Temporal Logic.
- Agents (operating on artifacts) call for Epistemic Logic.

The specification language FO-CTLK:

$$\varphi ::= P(\vec{t}) \mid t = t' \mid \neg\varphi \mid \varphi \rightarrow \varphi \mid \forall x \varphi \mid AX\varphi \mid A\varphi U \varphi \mid E\varphi U \varphi \mid K_i \varphi$$

Alternation of variables and path quantifiers is enabled.

Semantics of FO-CTLK

Formal definition

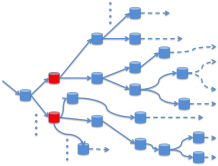
An AC-MAS $\mathcal{P}$ satisfies an FO-CTLK-formula φ in a state D for an assignment σ, or $(\mathcal{P}, D, \sigma) \models \varphi$, iff

$(\mathcal{P}, D, \sigma) \models P_i(\vec{t})$	iff	$\langle \sigma(t_1), \ldots, \sigma(t_\ell) \rangle \in D(P_i)$
$(\mathcal{P}, D, \sigma) \models t = t'$	iff	$\sigma(t) = \sigma(t')$
$(\mathcal{P}, D, \sigma) \models \neg\varphi$	iff	$(\mathcal{P}, D, \sigma) \not\models \varphi$
$(\mathcal{P}, D, \sigma) \models \varphi \rightarrow \psi$	iff	$(\mathcal{P}, D, \sigma) \not\models \varphi$ or $(\mathcal{P}, D, \sigma) \models \psi$
$(\mathcal{P}, D, \sigma) \models \forall x \varphi$	iff	for all $u \in adom(D)$, $(\mathcal{P}, D, \sigma^x_u) \models \varphi$
$(\mathcal{P}, D, \sigma) \models AX\varphi$	iff	for all runs r, $r^0 = D$ implies $(\mathcal{P}, r^1, \sigma) \models \varphi$
$(\mathcal{P}, D, \sigma) \models A\varphi U\varphi'$	iff	for all runs r, $r^0 = D$ implies $(\mathcal{P}, r^k, \sigma) \models \varphi'$ for some $k \geq 0$, and $(\mathcal{P}, r^{k'}, \sigma) \models \varphi$ for all $0 \leq k' < k$
$(\mathcal{P}, D, \sigma) \models E\varphi U\varphi'$	iff	there exists r s.t. $r^0 = D$, $(\mathcal{P}, r^k, \sigma) \models \varphi'$ for some $k \geq 0$, and $(\mathcal{P}, r^{k'}, \sigma) \models \varphi$ for all $0 \leq k' < k$
$(\mathcal{P}, D, \sigma) \models K_i\varphi$	iff	for all runs r, $n \in \mathbb{N}$, $D \sim_i r^n$ implies $(\mathcal{P}, r^n, \sigma) \models \varphi$

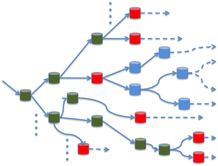
- Active-domain semantics for quantifiers.

Semantics of FO-CTLK

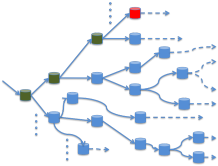
Intuition



(d) $AX\varphi$



(e) $A\varphi U\psi$



(f) $E\varphi U\psi$

Verification of AC-MAS

How do we verify FO-CTLK specifications on AC-MAS?

- the manufacturer M knows that each MO has to match a corresponding PO:

$$AG\ \forall id, pc\ (\exists pr, s\ MO(id, pc, pr, s) \rightarrow K_M\ \exists o, s'\ PO(id, pc, o, s'))$$

- the client C knows that every PO will eventually be discharged (by the manufacturer M):

$$AG\ \forall id, pc\ (\exists pr, s\ MO(id, pc, pr, s) \rightarrow EF\ K_C\ \exists o\ PO(id, ps, o, \text{shipped}))$$

Problem: the infinite domain U can determine infinitely many states!

Investigated solution: can we *simulate* the concrete values from U with a finite set of *abstract* symbols?

Abstraction: Isomorphism and Bisimulation

- Two states D, D' are *isomorphic*, or $D \simeq D'$, if there is a bijection $\iota : adom(D) \cup C \mapsto adom(D') \cup C$ s.t.
  - ι is the identity on C
  - for every $\vec{u} \in adom(D)^{a_i}$, $i \in Ag$, $\vec{u} \in D_i(P_j) \Leftrightarrow \iota(\vec{u}) \in D'_i(P_j)$

D	
a	b
b	c
d	e

$\simeq$

D'	
1	2
2	c
4	5

  - $\iota : a \mapsto 1$
    $b \mapsto 2$
    $c \mapsto$ c
    $d \mapsto 4$
    $e \mapsto 5$

Abstraction: Isomorphism and Bisimulation

- Two states D, D' are *bisimilar*, or $D \approx D'$, if
  - $D \simeq D'$
  - if $D \to E$ then there is E' s.t. $D' \to E'$, $D \oplus E \simeq D' \oplus E'$, and $E \approx E'$

$$
\begin{tikzcd}
\boxed{D} \arrow[rr] \arrow[d, phantom, "\approx"] & & \boxed{E} \\
\boxed{D'} & &
\end{tikzcd}
$$

Abstraction: Isomorphism and Bisimulation

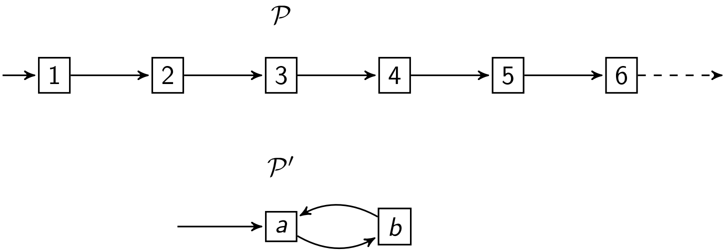
- Two states D, D' are *bisimilar*, or $D \approx D'$, if
  - $D \simeq D'$
  - if $D \rightarrow E$ then there is E' s.t. $D' \rightarrow E'$, $D \oplus E \simeq D' \oplus E'$, and $E \approx E'$

$$\begin{tikzcd} \boxed{D} \arrow[r] \arrow[d, phantom, "\approx"] & \boxed{E} \arrow[d, phantom, "\approx"] \\ \boxed{D'} \arrow[r] & \boxed{E'} \end{tikzcd}$$

  - similarly for the epistemic relation $\sim_i$
  - the other direction holds as well

Abstraction: Isomorphism and Bisimulation

However, bisimulation is not sufficient to preserve FO-CTLK formulas:



$$\phi \quad = \quad AG\ \forall x\ (P(x) \rightarrow AX\ AG\ \neg P(x))$$

Uniformity

- An AC-MAS $\mathcal{P}$ is *uniform* iff for $D, E, D' \in \mathcal{S}$ and $E' \in \mathcal{D}(U)$:
  - $D \rightarrow E$ and $D \oplus E \simeq D' \oplus E'$ imply $D' \rightarrow E'$

D	
a	b
b	c
d	e

E	
a	f
f	c

D'	
1	2
2	3
4	5

E'	
1	6
6	3

Uniformity

- An AC-MAS $\mathcal{P}$ is *uniform* iff for $D, E, D' \in \mathcal{S}$ and $E' \in \mathcal{D}(U)$:
  - $D \to E$ and $D \oplus E \simeq D' \oplus E'$ imply $D' \to E'$

D	
a	b
b	c
d	e

E	
a	f
f	c

D'	
1	2
2	3
4	5

E'	
1	6
6	3

- Intuitively, the behaviour of uniform AC-MAS is independent from data not explicitly named in the system description.
- Uniform AC-MAS cover a vast number of interesting cases [2, 4].

Bisimulation and Equivalence w.r.t. FO-CTLK

Theorem

Consider

- bisimilar *and* uniform *AC-MAS* $\mathcal{P}_1$ *and* $\mathcal{P}_2$
- *an FO-CTLK formula* φ

If

1. $|U_2| \geq 2 \cdot \sup_{D \in \mathcal{P}_1} |adom(D)| + |C| + |vars(\varphi)|$
2. $|U_1| \geq 2 \cdot \sup_{D \in \mathcal{P}_2} |adom(D)| + |C| + |vars(\varphi)|$

then

$$\mathcal{P}_1 \models \varphi \quad \textit{iff} \quad \mathcal{P}_2 \models \varphi$$

Can we apply this result to finite abstraction?

Abstractions

- Let $A = \langle \mathcal{D}, L, Act, Pr \rangle$ be an agent defined on the domain U.
  Given a domain U', the *abstract agent* $A' = \langle \mathcal{D}', L', Act', Pr' \rangle$ on U' is s. t.
  - $\mathcal{D}'_i = \mathcal{D}_i$
  - $L'_i = \mathcal{D}'_i(U')$
  - $Act'_i = Act_i$
  - $\alpha(\vec{u}') \in Pr'_i(l'_i)$ iff there exist $l_i \in L_i$ and $\alpha(\vec{u}) \in Pr_i(l_i)$ s.t. $l'_i \simeq l_i$, for some witness ι, and $\vec{u}' = \iota'(\vec{u})$, for some bijection ι' extending ι to $\vec{u}$.
- Given a set Ag of agents on U, let Ag' be the set of abstract agents on U'.
- Let $\mathcal{P} = \langle \mathcal{S}, U, D_0, \tau \rangle$ be an AC-MAS on the set Ag of agents.
  The AC-MAS $\mathcal{P}' = \langle \mathcal{S}', U', D'_0, \tau' \rangle$ on the set Ag' of abstract agents is an *$\oplus$-abstraction* of $\mathcal{P}$ iff:
  - $D'_0 = D_0$;
  - $t' \in \tau'(s', \vec{\alpha}(\vec{u}'))$ iff there exist $s, t \in \mathcal{S}$ and $\vec{\alpha}(\vec{u}) \in Act(U)$, such that $s \oplus t \simeq s' \oplus t'$, for some witness ι, $t \in \tau(s, \vec{\alpha}(\vec{u}))$, and $\vec{u}' = \iota'(\vec{u})$ for some bijection ι' extending ι to $\vec{u}$.

Bounded Models and Finite Abstractions

- An AC-MAS $\mathcal{P}$ is *b-bounded* iff for all $D \in \mathcal{P}$, $|adom(D)| \leq b$.
- Bounded systems can still be infinite.

Theorem

Consider

- *a* b-bounded *and* uniform *AC-MAS* $\mathcal{P}$ *on an infinite domain* U
- *an FO-CTLK formula* φ.

Given $U' \supseteq C$ *s.t.*

$$|U'| \geq 2b + |C| + \max\{|vars(\varphi)|, N_{Ag}\}$$

there exists a finite abstraction $\mathcal{P}'$ *of* $\mathcal{P}$ *s.t.*

- $\mathcal{P}'$ *is uniform and bisimilar to* $\mathcal{P}$

In particular,

$$\mathcal{P} \models \varphi \quad \textit{iff} \quad \mathcal{P}' \models \varphi$$

How can we define finite abstractions constructively?

Compact descriptions: AS Programs

Example of uniform AC-MAS written in a FO language.

- for each agent i, Act_i is the set of of *local (parametric) actions* of the form $\omega(\vec{x}) = \langle \pi(\vec{y}), \psi(\vec{z}) \rangle$ s.t.
  - $\omega(\vec{x})$ is the *operation signature* and $\vec{x} = \vec{y} \cup \vec{z}$ is the set of *operation parameters*
  - $\pi(\vec{y})$ is the *operation precondition*, i.e., an FO-formula over $\mathcal{D}_i$
  - $\psi(\vec{z})$ is the *operation postcondition*, i.e., an FO-formula over $\mathcal{D} \cup \mathcal{D}'$

We call the AC-MAS specified in this way *Artifact System Programs*.

Example 2: the Order-to-Cash Scenario

Specification of actions affecting the MO in the order-to-cash scenario:

- $createMO(po\_id, price) = \langle \pi(po\_id, price), \psi(po\_id, price) \rangle$, where:
- $\pi(po\_id, price) \equiv$
  $\exists p, o\ (PO(po\_id, p, o, \text{prepared}) \wedge \exists cost\ Materials(p, cost) \wedge \phi_{b-1}$
- $\psi(po\_id, price) \equiv$
  $\exists id\ (MO'(id, po\_id, price, \text{preparation}) \wedge$

  $\forall id', c, p, s\ (MO(id', c, p, s) \rightarrow id \neq id')) \wedge \phi_b$

where ϕ_k is the FO-formula saying that there are at most k objects in the active domain.

The specification of *createMO* guarantees that the bound b is not violated by action execution.

Verification of Artifact System Programs

Lemma

AS programs generate uniform AC-MAS.

Theorem

Consider

- *a* b-bounded *AS program* $\mathcal{P}_{Act,U}$ *on an infinite domain* U
- *an FO-CTLK formula* φ.

Given $U' \supseteq C$ *s.t.*

$$|U_2| \geq 2b + |C| + \max\{N_{AS}, |vars(\varphi)|\}$$

then $\mathcal{P}_{Act,U'}$ *is a finite abstraction of* $\mathcal{P}_{Act,U}$ *s.t.*

- $\mathcal{P}_{Act,U'}$ *is uniform and bisimilar to* $\mathcal{P}_{Act,U}$

In particular,

$$\mathcal{P}_{Act,U} \models \varphi \quad \textit{iff} \quad \mathcal{P}_{Act,U'} \models \varphi$$

- The abstraction is finite and the procedure is *constructive*.
- Thus, we can apply standard techniques in model checking.

Extensions

1. Non-uniform AC-MAS: for the *sentence-atomic* fragment of FO-CTL, the results above still hold.

$$AG\ \forall c\ (shippedPO(c) \rightarrow \forall m(related(c, m) \rightarrow shippedMO(m))) \qquad ✔$$

2. Non-uniform AC-MAS: one-way preservation result for FO-ACTL.

Theorem

If an AC-MAS $\mathcal{P}$ is bounded, *and $\varphi \in$ FO-ACTL, then there exists a finite abstraction $\mathcal{P}'$ such that if $\mathcal{P}' \models \varphi$ then $\mathcal{P} \models \varphi$.*

3. *Model checking bounded* AC-MAS w.r.t. FO-CTL is undecidable.
4. Complexity result:

Theorem

The model checking problem for finite AC-MAS w.r.t. FO-CTLK is EXPSPACE-complete in the size of the formula and data.

Results

and main limitations

- We are able to model check AC-MAS w.r.t. full FO-CTLK...
- ...however, our results hold only for *uniform* and *bounded* systems.
- This class includes many interesting systems (AS programs, [2, 4]).
- The model checking problem is EXPSPACE-complete.

Next Steps

- Techniques for finite abstraction.
- Abstraction techniques for finite-state systems are effective on the abstract system?
- How to perfom the boundedness check.

Merci!

Christel Baier and Joost-Pieter Katoen.
Principles of Model Checking.
MIT Press, 2008.

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.

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

B. Bagheri Hariri, D. Calvanese, G. De Giacomo, R. De Masellis, and P. Felli.
Foundations of Relational Artifacts Verification.
In *Proc. of BPM*, 2011.