Abstraction-based Verification of Infinite-state Data-aware Systems

Francesco Belardinelli
Laboratoire IBISC, Université d'Evry

based on work with Alessio Lomuscio
Imperial College London, UK

and Fabio Patrizi
Sapienza Università di Roma & Fondazione Bruno Kessler, Bolzano

Institut de Recherche en Informatique Fondamentale – 19 September 2016

Overview

1. Motivation and Background:
   - **Data-aware Systems**: new paradigm in Service-oriented Computing [CH09]
   - GSM [HDM+11], KAB [BCM+13], Situation Calculus [DLP16], Reactive Modules [AH99].
   - English (ascending bid) auctions as Data-aware Systems

Overview

1. Motivation and Background:
   - **Data-aware Systems**: new paradigm in Service-oriented Computing [CH09]
   - GSM [HDM+11], KAB [BCM+13], Situation Calculus [DLP16], Reactive Modules [AH99].
   - English (ascending bid) auctions as Data-aware Systems

2. Main Task: **formal** verification of **infinite-state** Data-aware Systems
   - Given a model $\mathcal{M}_S$ of system S and a formula ϕ_P for property P,

$$\text{does } \mathcal{M}_S \models \phi_P?$$

     - ★ model checking is appropriate for control-intensive applications...
     - ★ ...but less suited for data-intensive applications (data range over infinite domains) [BK08]

Overview

1. Motivation and Background:
   - **Data-aware Systems**: new paradigm in Service-oriented Computing [CH09]
   - GSM [HDM+11], KAB [BCM+13], Situation Calculus [DLP16], Reactive Modules [AH99].
   - English (ascending bid) auctions as Data-aware Systems

2. Main Task: **formal** verification of **infinite-state** Data-aware Systems
   - Given a model $\mathcal{M}_S$ of system S and a formula ϕ_P for property P,

$$\text{does } \mathcal{M}_S \models \phi_P?$$

     - ★ model checking is appropriate for control-intensive applications...
     - ★ ...but less suited for data-intensive applications (data range over infinite domains) [BK08]

3. Key Result:
   - Under specific conditions, the verification of DaS is decidable

   ⇒ The verification of various types of auction is decidable

- Recent paradigm in Service-Oriented Computing [CH09, DSV07, DHPV09].
  - aka data-driven/data-centric systems
  - **motto**: let's give *data* and *processes* the same relevance!
  - key idea behind the UE STREP project ACSI (http://acsi-project.haifa.il.ibm.com/)

- Recent paradigm in Service-Oriented Computing [CH09, DSV07, DHPV09].
  - aka data-driven/data-centric systems
  - **motto**: let's give *data* and *processes* the same relevance!
  - key idea behind the UE STREP project ACSI (http://acsi-project.haifa.il.ibm.com/)

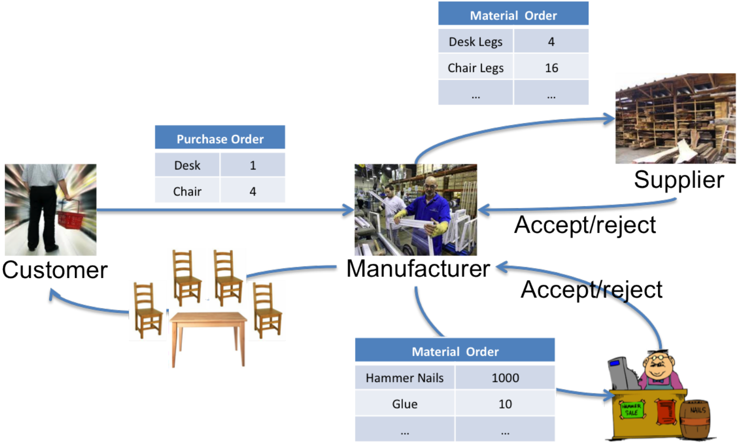
- ACSI: Artifact-Centric Service Interoperation
  - **Artifact**: data model + lifecycle
    - (nested) records equipped with actions
    - actions may affect several artifacts
    - evolution stemming from the interaction with other artifacts/external actors
  - **Artifact System**: interacting artifacts, representing services, manipulated by agents.
    - several frameworks to formalise Artifact Systems and DaS in general (GSM, KAB, ...).

- Recent paradigm in Service-Oriented Computing [CH09, DSV07, DHPV09].
  - aka data-driven/data-centric systems
  - **motto**: let's give *data* and *processes* the same relevance!
  - key idea behind the UE STREP project ACSI (http://acsi-project.haifa.il.ibm.com/)
- ACSI: Artifact-Centric Service Interoperation
  - **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**: interacting artifacts, representing services, manipulated by agents.
    - several frameworks to formalise Artifact Systems and DaS in general (GSM, KAB, . . .).
- **Auctions** as Data-aware Systems
  - the auctioneer and bidders compare bids
  - the bidders' behaviour depends on the value of bids

- Recent paradigm in Service-Oriented Computing [CH09, DSV07, DHPV09].
  - aka data-driven/data-centric systems
  - **motto**: let's give *data* and *processes* the same relevance!
  - key idea behind the UE STREP project ACSI (http://acsi-project.haifa.il.ibm.com/)
- ACSI: Artifact-Centric Service Interoperation
  - **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**: interacting artifacts, representing services, manipulated by agents.
    - several frameworks to formalise Artifact Systems and DaS in general (GSM, KAB, ...).
- **Auctions** as Data-aware Systems
  - the auctioneer and bidders compare bids
  - the bidders' behaviour depends on the value of bids
- **Logical Perspective**: first-order modal (temporal) Kripke models

Data-aware 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
...	...

1. a single **auctioneer** a and a finite number of **bidders** $b_1, \ldots, b_\ell$

1. a single **auctioneer** a and a finite number of **bidders** $b_1, \ldots, b_\ell$
2. the auctioneer puts on sale an item with a **base price** (**public** to all bidders)

1. a single **auctioneer** a and a finite number of **bidders** $b_1, \ldots, b_\ell$
2. the auctioneer puts on sale an item with a **base price** (**public** to all bidders)
3. the bidding process is structured in discrete rounds

1. a single **auctioneer** a and a finite number of **bidders** $b_1, \ldots, b_\ell$
2. the auctioneer puts on sale an item with a **base price** (**public** to all bidders)
3. the bidding process is structured in discrete rounds
4. at each round every bidder can either bid or skip

1. a single **auctioneer** a and a finite number of **bidders** $b_1, \ldots, b_\ell$
2. the auctioneer puts on sale an item with a **base price** (**public** to all bidders)
3. the bidding process is structured in discrete rounds
4. at each round every bidder can either bid or skip
5. at time out the item is assigned to the bidder with the highest bid.

1. a single **auctioneer** a and a finite number of **bidders** $b_1, \ldots, b_\ell$
2. the auctioneer puts on sale an item with a **base price** (**public** to all bidders)
3. the bidding process is structured in discrete rounds
4. at each round every bidder can either bid or skip
5. at time out the item is assigned to the bidder with the highest bid.
6. the auctioneer puts another item on sale ...

1. a single **auctioneer** a and a finite number of **bidders** $b_1, \ldots, b_\ell$
2. the auctioneer puts on sale an item with a **base price** (**public** to all bidders)
3. the bidding process is structured in discrete rounds
4. at each round every bidder can either bid or skip
5. at time out the item is assigned to the bidder with the highest bid.
6. the auctioneer puts another item on sale . . .

Assumptions:

1. a single **auctioneer** a and a finite number of **bidders** $b_1, \ldots, b_\ell$
2. the auctioneer puts on sale an item with a **base price** (**public** to all bidders)
3. the bidding process is structured in discrete rounds
4. at each round every bidder can either bid or skip
5. at time out the item is assigned to the bidder with the highest bid.
6. the auctioneer puts another item on sale ...

Assumptions:

- each bidder is rational

1. a single **auctioneer** a and a finite number of **bidders** $b_1, \ldots, b_\ell$
2. the auctioneer puts on sale an item with a **base price** (**public** to all bidders)
3. the bidding process is structured in discrete rounds
4. at each round every bidder can either bid or skip
5. at time out the item is assigned to the bidder with the highest bid.
6. the auctioneer puts another item on sale ...

Assumptions:

- each bidder is rational
- she has an **intrinsic value** for each item being auctioned

1. a single **auctioneer** a and a finite number of **bidders** $b_1, \ldots, b_\ell$
2. the auctioneer puts on sale an item with a **base price** (**public** to all bidders)
3. the bidding process is structured in discrete rounds
4. at each round every bidder can either bid or skip
5. at time out the item is assigned to the bidder with the highest bid.
6. the auctioneer puts another item on sale . . .

Assumptions:

- each bidder is rational
- she has an **intrinsic value** for each item being auctioned
- and she keeps this information **private** from other bidders and the auctioneer

Data-aware Systems

Auction Data Model

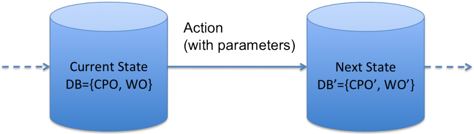
Bidding					
item	*base_price*	bid_1	...	bid_ℓ	*status*

- $init_A(item, base\_price)$
- $bid_i(item, bid)$
- $time\_out(item)$
- $skip_A$
- $skip_i$
- ...

$trueValue_i$	
item	*true_value*

- $init_i(item, true\_value)$
- ...

- Agents operate on the data model
  - e.g., the bidder sends a new bid to the auctioneer
- Actions add/remove artifacts or change artifact attributes
  - e.g., the auctioneer puts a new item on auction
- The whole system can be seen as a dynamic *data-aware* system
  - at every step, an action yields a change in the current state



Research questions

1. Which syntax and semantics to specify Data-aware Systems?

Research questions

1. Which syntax and semantics to specify Data-aware Systems?
2. Is verification of DaS decidable?

Research questions

1. Which syntax and semantics to specify Data-aware Systems?
2. Is verification of DaS decidable?
3. If not, can we identify **interesting** fragments that are reasonably well-behaved?

Challenges

Distributed (multi-agent) systems, but . . .

Challenges

Distributed (multi-agent) systems, but . . .

- . . . states have a relational structure,

Challenges

Distributed (multi-agent) systems, but ...

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

Challenges

Distributed (multi-agent) systems, but ...

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

Challenges

Distributed (multi-agent) systems, but ...

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

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

1. **Artifact-centric Multi-agent Systems** (AC-MAS) as a formal model for DaS.
   Intuition: databases that evolve over time and are manipulated by agents.

1. **Artifact-centric Multi-agent Systems** (AC-MAS) as a formal model for DaS.
   Intuition: databases that evolve over time and are manipulated by agents.

2. Specification language: first-order extensions of temporal (strategy) logics

$$AG\ \forall it, \vec{bd}, s(\exists! bp\ Bidding(it, \vec{bd}, bp, s) \wedge \exists^{\leq 1} tv\ trueValue_i(it, tv))$$

   each item has exactly one base price, while bidders associate at most one true value to each item (possibly none).

1. **Artifact-centric Multi-agent Systems** (AC-MAS) as a formal model for DaS.
   Intuition: databases that evolve over time and are manipulated by agents.

2. Specification language: first-order extensions of temporal (strategy) logics

$$AG\ \forall it, \vec{bd}, s(\exists! bp\ Bidding(it, \vec{bd}, bp, s) \wedge \exists^{\leq 1} tv\ trueValue_i(it, tv))$$

   each item has exactly one base price, while bidders associate at most one true value to each item (possibly none).

3. Model theory of FO modal logic: bisimulations and abstraction to tackle model checking.
   Main result: under specific conditions MC can be reduced to the finite case.

1. **Artifact-centric Multi-agent Systems** (AC-MAS) as a formal model for DaS.
   Intuition: databases that evolve over time and are manipulated by agents.

2. Specification language: first-order extensions of temporal (strategy) logics

$$AG\ \forall it, \vec{bd}, s(\exists! bp\ Bidding(it, \vec{bd}, bp, s) \land \exists^{\leq 1} tv\ trueValue_i(it, tv))$$

   each item has exactly one base price, while bidders associate at most one true value to each item (possibly none).

3. Model theory of FO modal logic: bisimulations and abstraction to tackle model checking.
   Main result: under specific conditions MC can be reduced to the finite case.

4. Case study: modelling and veryfing auctions as AC-MAS.

Semantics: Databases

The data model of DaS is given as a database.

- a **database schema** is a *finite* set $\mathcal{D} = \{P_1/a_1, \ldots, P_n/a_n\}$ of (typed) relation symbols P_i with arity $a_i \in \mathbb{N}$
- Consider a (possibly infinite) interpretation domain U.
  A **db instance** on U is a mapping D associating each symbol P_i with a *finite* a_i-ary relation on U
- the domain U may be ordered (e.g. reals and rationals with $\leq$)
- the **active domain** $adom(D)$ is the set of all $u \in U$ appearing in some $D(P_i)$.
  The active domain is always finite
- the **disjoint union** $D \oplus D'$ is the $(\mathcal{D} \cup \mathcal{D}')$-interpretation s.t.
  (i) $D \oplus D'(P_i) = D(P_i)$
  (ii) $D \oplus D'(P_i') = D'(P_i)$

Agents have partial observability (*imperfect information*) of the system.

- An **agent** $i = \langle \mathcal{D}_i, Act_i, Pr_i \rangle$ is such that
  - she registers her information in the **local database schema** $\mathcal{D}_i$, and
  - performs the **parametric actions** $\alpha(\vec{x})$ in Act_i
  - according to the **local protocol** $Pr_i : \mathcal{D}_i(U) \mapsto 2^{Act_i(U)}$
- the setting is inspired by the *interpreted systems semantics* for MAS [FHMV95],...
- ...but here the local state of each agent is relational.

Agents manipulate data and have (partial) observability of the information contained in the global db schema $\mathcal{D} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_\ell$.

Example 1: English Auction

- agents: auctioneer, bidder$_1$, ..., bidder$_\ell$
- local db schema $\mathcal{D}_a$ for auctioneer
  - *Bidding(item, base_price, bid$_1$, ..., bid$_\ell$, status)*
- local db schema $\mathcal{D}_i$ for bidders
  - *Bidding(item, base_price, bid$_1$, ..., bid$_\ell$, status)*
  - *TValue$_i$(item, true_value)*
- then, $\mathcal{D} = \{Bidding, TValue_1, \ldots, TValue_\ell\}$
- actions introduce values from an infinite domain $U = Items \cup \mathbb{Q} \cup \{active, term\}$:
  - *init$_a$(item, base_price), time out(item), skip$_a$* belong to Act_a
  - *init$_i$(item, true_value), bid$_i$(item, bid), skip$_i$* belong to each Act_i
- the protocol function specifies the preconditions for actions:
  - e.g., $bid_i(item, bid) \in Pr_i(D)$ whenever
    - ★ *item* appears in $D(TValue_i)$
    - ★ for all $j \neq i$, $bid_j < bid \leq true\_value_i$
    - ★ $D(status) = active$ for *item*
  - the *skip* actions are always enabled.

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

- a **global state** $s = \langle D_0, \ldots, D_\ell \rangle$ registers information about all agents.
- an **AC-MAS** $\mathcal{P} = \langle Ag, s_0, \rightarrow \rangle$ describes the interactions of ...
  - a **finite set** $Ag = \{a_0, \ldots, a_\ell\}$ of agents
  - from some **initial global state** s_0
  - according to the **transition relation** $s \xrightarrow{\alpha(\vec{u})} s'$
- AC-MAS are infinite-state systems in general

AC-MAS are first-order temporal structures.

$\Rightarrow$ FO temporal logics can be used as specification languages.

Example 2: the Auction AC-MAS

The **Auction AC-MAS** $\mathcal{A} = \langle Ag, s_0, \rightarrow \rangle$ is given as

- $Ag = \{a, b_1, \ldots, b_\ell\}$
- s_0 is the **empty interpretation** of $\mathcal{D} = \{Bidding, TValue_1, \ldots, TValue_\ell\}$
- $\rightarrow$ is the **transition relation** s.t. $s \xrightarrow{\alpha(\vec{u})} s'$ whenever
  - $\alpha_i = bid_i(item, bid')$ and s' modifies s by replacing any tuple $(item, \ldots, bid_i, \ldots, status)$ in $D_s(Bidding)$ with $(item, \ldots, bid'_i, \ldots, status)$
  - $\alpha_A = timeout(item)$ and the value of $status$ in $D_{s'}(Bidding)$ for $item$ is $term$
  - $\ldots$

Syntax: First-order CTL

- Data call for First-order Logic
- Evolution calls for Temporal Logic

The specification language **FO-CTL**:

$$\varphi \quad ::= \quad P(t_1, \ldots, t_a) \mid t = t' \mid t \leq t' \mid \neg\varphi \mid \varphi \rightarrow \varphi \mid \forall x \varphi \mid AX\varphi \mid A\varphi U\varphi \mid E\varphi U\varphi$$

where P is any relation symbol in $\mathcal{D}$.

Alternation of free variables and modal operators is enabled.

- We can also deal with FO extensions of ATL, as well as epistemic modalities [BLP14, BL16].

Semantics of FO-CTL

Formal definition

An **assignment** is a function $\sigma : Var \to U$.

An AC-MAS $\mathcal{P}$ **satisfies** an FO-CTL formula φ in a state s for an assignment σ, iff

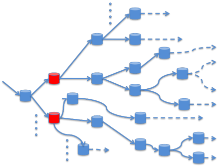
$$
\begin{array}{lll}
(\mathcal{P}, s, \sigma) \models P(\vec{t}) & \text{iff} & \langle \sigma(t_1), \ldots, \sigma(t_a) \rangle \in D_s(P) \\
(\mathcal{P}, s, \sigma) \models t = t' & \text{iff} & \sigma(t) = \sigma(t') \\
(\mathcal{P}, s, \sigma) \models t \leq t' & \text{iff} & \sigma(t) \leq \sigma(t') \\
(\mathcal{P}, s, \sigma) \models \neg\varphi & \text{iff} & (\mathcal{P}, s, \sigma) \not\models \varphi \\
(\mathcal{P}, s, \sigma) \models \varphi \to \psi & \text{iff} & (\mathcal{P}, s, \sigma) \not\models \varphi \text{ or } (\mathcal{P}, s, \sigma) \models \psi \\
(\mathcal{P}, s, \sigma) \models \forall x \varphi & \text{iff} & \text{for every } u \in adom(s),\ (\mathcal{P}, s, \sigma^x_u) \models \varphi \\
(\mathcal{P}, s, \sigma) \models AX\varphi & \text{iff} & \text{for every run } r,\ r(0) = s \text{ implies } (\mathcal{P}, r(1), \sigma) \models \varphi \\
(\mathcal{P}, s, \sigma) \models A\varphi U \varphi' & \text{iff} & \text{for every run } r,\ r(0) = s \text{ implies } (\mathcal{P}, r(k), \sigma) \models \varphi' \text{ for some } k \geq 0, \\
 & & \text{and } (\mathcal{P}, r(k'), \sigma) \models \varphi \text{ for every } 0 \leq k' < k \\
(\mathcal{P}, s, \sigma) \models E\varphi U \varphi' & \text{iff} & \text{for some run } r,\ r(0) = s,\ (\mathcal{P}, r(k), \sigma) \models \varphi' \text{ for some } k \geq 0, \\
 & & \text{and } (\mathcal{P}, r(k'), \sigma) \models \varphi \text{ for all } 0 \leq k' < k
\end{array}
$$

Active-domain semantics, but...

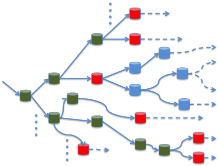
- ...we can refer to individuals that no longer exist
- the number of states is infinite in general

Semantics of FO-CTL

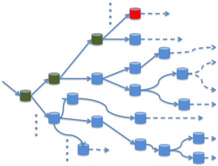
Intuition



(a) $AX\varphi$



(b) $A\varphi U\psi$



(c) $E\varphi U\psi$

Verification of AC-MAS

How do we check FO-CTL specifications on auctions?

- for each bidder, each bid is less than or equal to her true value:

$$AG\ \forall it, \vec{x}, bd_i, \vec{y}, tv(Bidding(it, \vec{x}, bd_i, \vec{y}) \wedge TValue_i(it, tv) \rightarrow bd_i \leq tv)$$

- each bidder can raise her bid unless she has already hit her true value:

$$\begin{aligned} &AG\ \forall it, \vec{x}, bd_i, \vec{y}(Bidding(it, \vec{x}, bd_i, \vec{y}) \rightarrow \\ &\rightarrow (TValue_i(it, bd_i) \vee EF\ \exists \vec{x}', bd_i', \vec{y}'(bd_i' > bd_i \wedge Bidding(it, \vec{x}', bd_i', \vec{y}')))) \end{aligned}$$

- define

$$\begin{aligned} Win_i(it) \quad = \quad & Status(it, \text{term}) \wedge \exists \vec{x}, bd_i, \vec{y}(Bidding(it, \vec{x}, bd_i, \vec{y}) \wedge \\ & \wedge \bigwedge_{j \neq i} \forall \vec{x'}, bd_j, \vec{y'}(Bidding(it, \vec{x'}, bd_j, \vec{y'}) \rightarrow bd_j < bd_i)) \end{aligned}$$

Manipulability: bidder b_i will necessarily win the auction for item *it* eventually

$$AF\,Win_i(it)$$

Problem: the infinite domain U may generate infinitely many states!

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

Bisimulation: Isomorphism

- two states s, s' are **isomorphic**, or $s \simeq s'$, if there is a bijection

$$\iota : adom(s) \mapsto adom(s')$$

such that for every $\vec{u}$ in $adom(s)$, $i \in Ag$, $\vec{u} \in D_i(P) \Leftrightarrow \iota(\vec{u}) \in D'_i(P)$

	$D(P_i)$	
P_1	a	b
P_2	b	c
P_3	d	e

$\simeq$

	$D'(P_i)$	
P_1	1	2
P_2	2	3
P_3	4	5

- $\iota : a \mapsto 1$
  $b \mapsto 2$
  $c \mapsto 3$
  $d \mapsto 4$
  $e \mapsto 5$

Bisimulation

- two states s, s' are **bisimilar**, or $s \approx s'$, if
  1. $s \simeq s'$
  2. if $s \to t$ then for some t', $s' \to t'$, $s \oplus t \simeq s' \oplus t'$, and $t \approx t'$

$$\begin{tikzcd} \boxed{s} \arrow[r] \arrow[d, phantom, "\approx"] & \boxed{t} \\ \boxed{s'} & \end{tikzcd}$$

Bisimulation

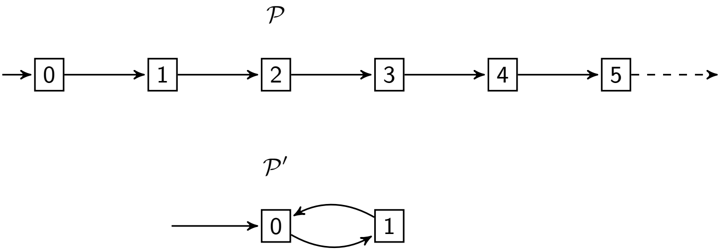
- two states s, s' are **bisimilar**, or $s \approx s'$, if
  1. $s \simeq s'$
  2. if $s \to t$ then for some t', $s' \to t'$, $s \oplus t \simeq s' \oplus t'$, and $t \approx t'$

$$\begin{tikzcd} \boxed{s} \arrow[r] \arrow[d, phantom, "\approx"] & \boxed{t} \arrow[d, phantom, "\approx"] \\ \boxed{s'} \arrow[r] & \boxed{t'} \end{tikzcd}$$

  3. the other direction holds as well

Bisimulation

However, bisimulations are not sufficient to preserve FO-CTL formulas:



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

Uniformity

- The behaviour of uniform AC-MAS is **independent** from data not explicitly mentioned in the system description.
- related to the notion of **genericity** in databases.

Uniformity

- The behaviour of uniform AC-MAS is **independent** from data not explicitly mentioned in the system description.
- related to the notion of **genericity** in databases.
- more formally, an AC-MAS $\mathcal{P}$ is **uniform** iff for states $s, t, s' \in \mathcal{S}$ and $t' \in \mathcal{D}(U)$,
  - $s \rightarrow t$ and $s \oplus t \simeq s' \oplus t'$ imply $s' \rightarrow t'$

s	
a	b
b	c
d	e

t	
a	f
f	c

s'	
1	2
2	3
4	5

t'	
1	6
6	3

Uniformity

- The behaviour of uniform AC-MAS is **independent** from data not explicitly mentioned in the system description.
- related to the notion of **genericity** in databases.
- more formally, an AC-MAS $\mathcal{P}$ is **uniform** iff for states $s, t, s' \in \mathcal{S}$ and $t' \in \mathcal{D}(U)$,
  - $s \rightarrow t$ and $s \oplus t \simeq s' \oplus t'$ imply $s' \rightarrow t'$

s	
a	b
b	c
d	e

t	
a	f
f	c

s'	
1	2
2	3
4	5

t'	
1	6
6	3

Uniformity

- The behaviour of uniform AC-MAS is **independent** from data not explicitly mentioned in the system description.
- related to the notion of **genericity** in databases.
- more formally, an AC-MAS $\mathcal{P}$ is **uniform** iff for states $s, t, s' \in \mathcal{S}$ and $t' \in \mathcal{D}(U)$,
  - $s \to t$ and $s \oplus t \simeq s' \oplus t'$ imply $s' \to t'$

s	
a	b
b	c
d	e

t	
a	f
f	c

s'	
1	2
2	3
4	5

t'	
1	6
6	3

- Uniform AC-MAS cover most cases of interest
  - GSM [HDDM+11], KAB [BCM+13], Situation Calculus [DLP16], Reactive Modules [BL16]
  - by assuming suitable restrictions on the language (e.g., no function symbols)

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

Theorem (Preservation Result)

Consider

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

If

1. $|U'| \geq 2 \cdot \sup_{s \in \mathcal{P}}\{|adom(s)|\} + |vars(\varphi)|$
2. $|U| \geq 2 \cdot \sup_{s' \in \mathcal{P}'}\{|adom(s')|\} + |vars(\varphi)|$

then

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

The condition on domains allows us to mimick the transitions in each system.

Can we apply this result to obtain finite abstractions?

Abstraction

Abstractions are defined in an agent-based, modular way.

- Let $i = \langle \mathcal{D}, Act, Pr \rangle$ be an agent defined on domain U.
  Given domain U', the **abstract agent** $i' = \langle \mathcal{D}, Act, Pr' \rangle$ on U' is s.t.
  - Pr' is the smallest function s.t. for every $D' \in \mathcal{D}'(U')$, if
    1. $D' \simeq D$ for some witness ι
    2. $\alpha(\vec{u}) \in Pr(D)$

    then $\alpha(\iota(\vec{u})) \in Pr'(D')$.

- Let $\mathcal{P} = \langle Ag, s_0, \rightarrow \rangle$ be an AC-MAS.
  The **abstraction** $\mathcal{P}' = \langle Ag', s'_0, \rightarrow' \rangle$ of $\mathcal{P}$ is an AC-MAS s.t.
  - Ag' be the set of abstract agents on U'
  - $s'_0 \simeq s_0$
  - $\rightarrow'$ is the smallest function s.t. if
    1. $s \xrightarrow{\alpha(\vec{u})} t$
    2. $s \oplus t \simeq s' \oplus t'$ for some witness ι

    then $s' \xrightarrow{\alpha(\iota(\vec{u}))} t'$.

Abstraction

- Let $N_{Ag} = \sum_{i \in Ag} \max_{\{\alpha(\vec{x}) \in Act_i\}} |\vec{x}|$ be the sum of the maximum numbers of parameters contained in the action types of each agent

Lemma (Abstraction Existence)

Consider

- *a* **uniform** *AC-MAS* $\mathcal{P}$
- *a set* U' *s.t.* $|U'| \geq 2 \sup_{s \in \mathcal{P}} |adom(s)| + N_{Ag}$

Then, there exists an abstraction $\mathcal{P}'$ *of* $\mathcal{P}$ *that is* **uniform** *and* **bisimilar** *to* $\mathcal{P}$*.*

How can we obtain **finite** abstractions?

Bounded Models and Finite Abstractions

- An AC-MAS $\mathcal{P}$ is b-**bounded** iff for all $s \in \mathcal{P}$, $|adom(s)| \leq b$
- Bounded systems can still be infinite!
- Bounded systems arise naturally
  - e.g., in reactive modules each agent controls a finite number of variables

Theorem (Finite Abstraction)

Consider

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

Given a finite domain U' s.t.

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

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

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

In particular,

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

$\Rightarrow$ Under specific conditions, we can model check an infinite-state system by verifying its finite abstraction.

Finite Abstract Auction I

- Suppose that at most n items are put on sale simultaneously
  - the auction AC-MAS $\mathcal{A}$ is bounded by $b = (2|Ag| - 1)n + 2$
- Consider a finite U' such that $|U'| \geq 2b + |vars(\phi)|$
- Define abstract agents auctioneer a' and bidders b'_i s.t.
  - the local db schemas $\mathcal{D}'_a$ and $\mathcal{D}'_i$ are the same as for a and b_i
  - the sets of actions Act'_a and Act'_i are the same as for a and b_i
  - the protocol function Pr'_a is the same as for a
  - as to Pr'_i, $bid_i(item, bid) \in Pr'_i(D')$ whenever
    - bid is an abstract value that does not represent any bid in D'
    - ...

Finite Abstract Auction II

The abstract auction AC-MAS $\mathcal{A}' = \langle Ag', s'_0, \tau' \rangle$ is defined as

- $Ag' = \{a', b'_1, \ldots, b'_\ell\}$
- s'_0 is the empty interpretation of $\mathcal{D}$
- $\rightarrow'$ mimics $\rightarrow$
  - e.g., if $\alpha_i = bid_i(item, bid)$, then $s \xrightarrow{\alpha(\vec{u})}{}' t$ whenever t modifies s by replacing any tuple $(item, \ldots, bid_i, \ldots, status)$ in $D_s(Bidding)$ with $(item, \ldots, bid'_i, \ldots, status)$, where the value $bid' \in U'$ has been found as above.
    In particular, $bid < bid' \leq true\_value$ in t.

- By assuming that $|U'| \geq 2b + |vars(\phi)|$ and Theorem 3 we have that $\mathcal{A}'$ is a finite abstraction of $\mathcal{A}$.
- In particular, $\mathcal{A}'$ is uniform and bisimilar to $\mathcal{A}$ and

$$\mathcal{A} \models \varphi \quad \text{iff} \quad \mathcal{A}' \models \varphi$$

Extensions

1. First-order extension of ATL: alternating bisimulations [BL16]

Extensions

1. First-order extension of ATL: alternating bisimulations [BL16]
2. Epistemic operators for individual and group knowledge [BLP14]

$$AG\ \forall it\ \neg\exists tv \bigvee_{j \neq i \vee j = a} K_j\ TValue_i(it, tv)$$

the true value of items for each bidder b_i is secret to all other bidders and the auctioneer

Extensions

1. First-order extension of ATL: alternating bisimulations [BL16]
2. Epistemic operators for individual and group knowledge [BLP14]

$$AG\ \forall it\ \neg\exists tv \bigvee_{j\neq i \vee j=a} K_j\ TValue_i(it, tv)$$

the true value of items for each bidder b_i is secret to all other bidders and the auctioneer

3. Non-uniform and bounded AC-MAS: one-way preservation result for FO-ACTL [Bel14]:

Theorem

For every AC-MAS $\mathcal{P}$ and $\varphi \in$ FO-ACTL, there exists a finite abstraction $\mathcal{P}'$ s.t.

$$\mathcal{P}' \models \varphi \quad \Rightarrow \quad \mathcal{P} \models \varphi$$

Extensions

1. First-order extension of ATL: alternating bisimulations [BL16]
2. Epistemic operators for individual and group knowledge [BLP14]

$$AG\ \forall it\ \neg \exists tv \bigvee_{j \neq i \vee j = a} K_j\ TValue_i(it, tv)$$

the true value of items for each bidder b_i is secret to all other bidders and the auctioneer

3. Non-uniform and bounded AC-MAS: one-way preservation result for FO-ACTL [Bel14]:

Theorem

For every AC-MAS $\mathcal{P}$ and $\varphi \in$ FO-ACTL, there exists a finite abstraction $\mathcal{P}'$ s.t.

$$\mathcal{P}' \models \varphi \quad \Rightarrow \quad \mathcal{P} \models \varphi$$

4. Model checking **bounded** AC-MAS w.r.t. FO-CTL is undecidable [BL13, LM14]

Extensions

1. First-order extension of ATL: alternating bisimulations [BL16]
2. Epistemic operators for individual and group knowledge [BLP14]

$$AG\ \forall it\ \neg\exists tv \bigvee_{j\neq i \vee j=a} K_j\ TValue_i(it, tv)$$

the true value of items for each bidder b_i is secret to all other bidders and the auctioneer

3. Non-uniform and bounded AC-MAS: one-way preservation result for FO-ACTL [Bel14]:

Theorem

For every AC-MAS $\mathcal{P}$ and $\varphi \in$ FO-ACTL, there exists a finite abstraction $\mathcal{P}'$ s.t.

$$\mathcal{P}' \models \varphi \quad \Rightarrow \quad \mathcal{P} \models \varphi$$

4. Model checking **bounded** AC-MAS w.r.t. FO-CTL is undecidable [BL13, LM14]
5. Complexity result [BLP14]:

Theorem

The model checking problem for finite AC-MAS w.r.t. FO-CTL is EXPSPACE-complete.

Results

and main limitations

- Bisimulation and finite abstraction for first-order Kripke models.
- We are able to model check AC-MAS w.r.t. full FO-CTL...
- ...however, our abstraction results hold only for **uniform** and **bounded** systems.
- This class includes many interesting systems
  - GSM [HDDM$^+$11], KAB [BCM$^+$13], Situation Calculus [DLP16], Reactive Modules [BL16])
- including English auctions.

Next Steps

- Constructive techniques for finite abstractions.
- Model checking techniques for finite-state systems are effective on DaS?
- How to perfom the boundedness check?
- What if the system is *unbounded*/*not uniform*?
  - can we include some (limited form of) arithmetic?

Thank you!

References

R. Alur and T. Henzinger.
Reactive modules.
Formal Methods in System Design, 15(1):7–48, 1999.

B. Bagheri, D. Calvanese, M. Montali, G. Giacomo, and A. Deutsch.
Verification of relational data-centric dynamic systems with external services.
In *Proceedings of the 32nd Symposium on Principles of Database Systems (PODS13)*, pages 163–174. ACM, 2013.

F. Belardinelli.
Verification of non-uniform and unbounded artifact-centric systems : Decidabi- lity through abstraction.
In *Proc. of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS14)*, 2014.

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

F. Belardinelli and A. Lomuscio.
Decidability of model checking non-uniform artifact-centric quantified interpreted systems.
In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI13)*, pages 725–731. AAAI Press, 2013.

Francesco Belardinelli and Alessio Lomuscio.
Abstraction-based verification of infinite-state reactive modules.
In *Proc. of the 22th European Conference on Artificial Intelligence (ECAI16)*, 2016.

F. Belardinelli, A. Lomuscio, and F. Patrizi.
Verification of agent-based artifact systems.
Journal of Artificial Intelligence Research, 51:333–376, 2014.

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.

A. Deutsch, R. Hull, F. Patrizi, and V. Vianu.
Automatic Verification of Data-Centric Business Processes.