Verification of Data-Centric Dynamic Systems

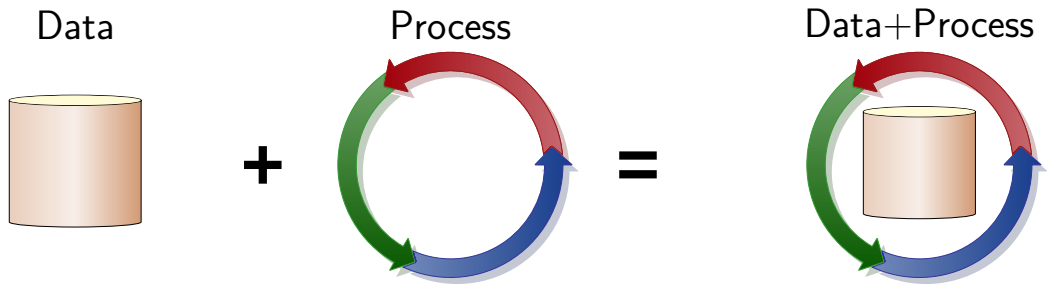
Babak Bagheri Hariri

Supervisor: Diego Calvanese

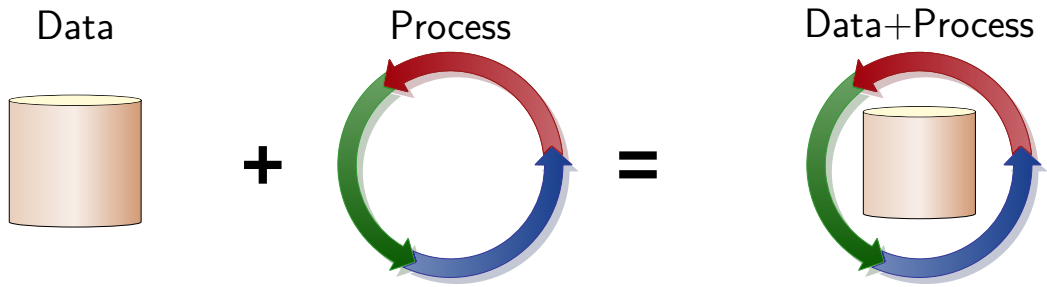
KRDB Research Centre for Knowledge and Data
Free University of Bozen-Bolzano

September, 2012

Modeling both structural and behavioral aspects



Modeling both structural and behavioral aspects



In our research we study **the boundaries of decidability**:

Design

- formalisms for modeling **knowledge** and **behavior**,
- languages for expressing **dynamic properties**,

such that:

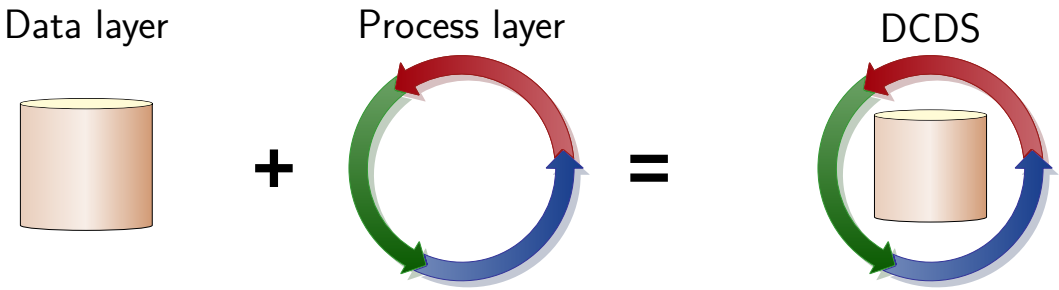
Verification of dynamic properties over the given formalism **is decidable**.

Data-Centric Dynamic Systems (DCDS)

We introduce DCDS, to

- explore different variants of modeling data and process
- abstract away from irrelevant factors of different scenario.

Data-Centric Dynamic Systems (DCDS)

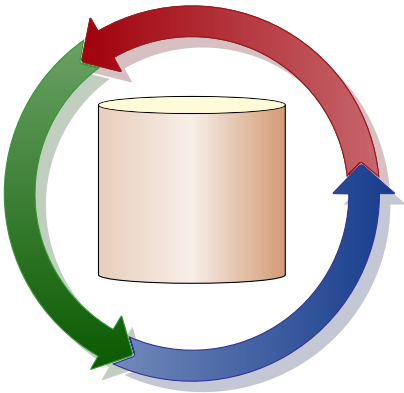


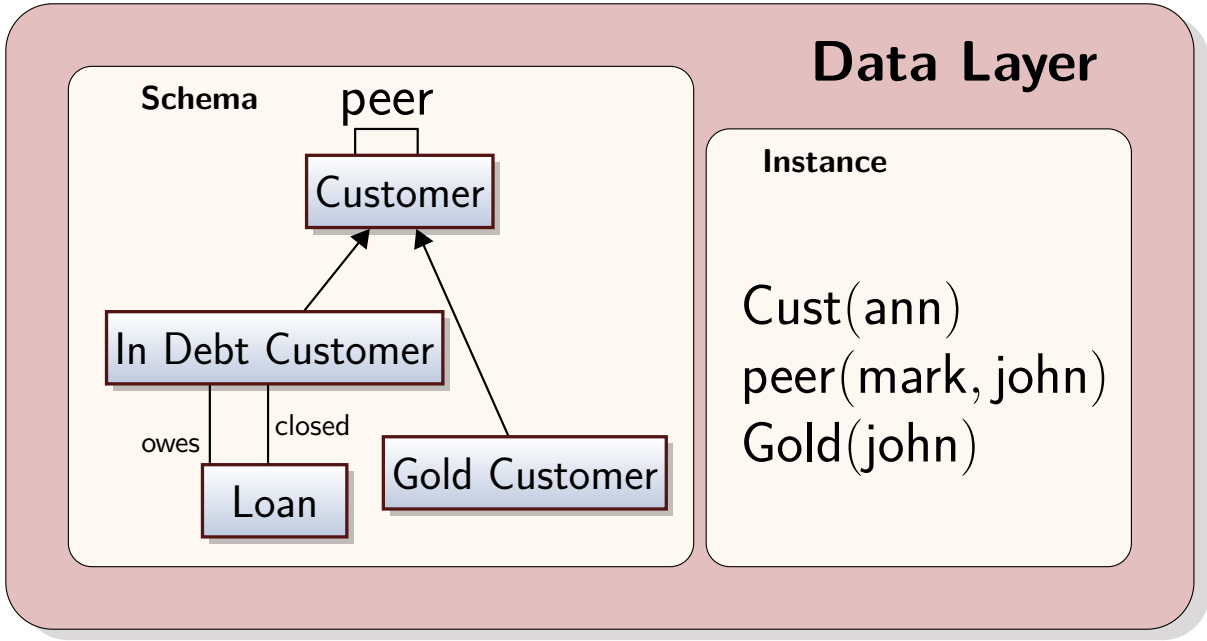
DCDS:

- Data Layer: Relational databases / ontologies
  - Data schema
  - Data instance: state of the DCDS
- Process Layer:
  - Atomic actions
  - Conditions for application of actions
  - Service calls: communication with external environment
    - ★ Deterministic services: e.g., historical exchange rate of Euro/USD
    - ★ Nondeterministic services: e.g., current exchange rate of Euro/USD

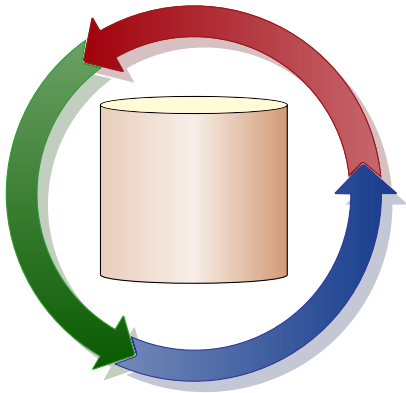
      Allow one also to take into account user-input.

DCDS, example



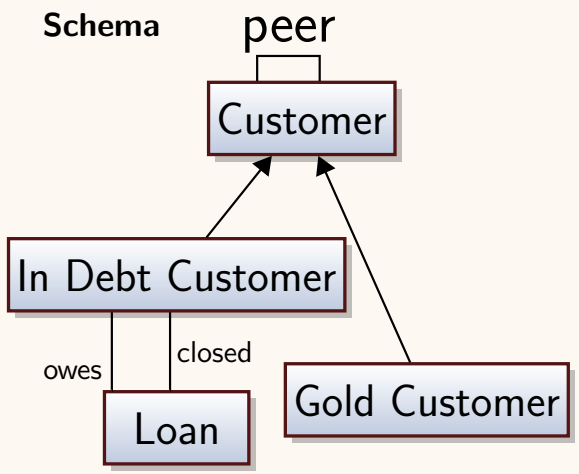


DCDS, example



Data Layer

Schema



Instance

$\mathsf{Cust}(\mathsf{ann})$
$\mathsf{peer}(\mathsf{mark}, \mathsf{john})$
$\mathsf{Gold}(\mathsf{john})$

Process Layer

Conditions

$$\mathsf{peer}(x, y) \wedge \mathsf{Gold}(y) \longmapsto \mathsf{GetLoan}(x)$$

Service Calls

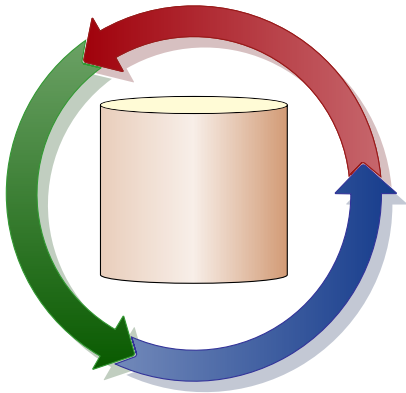
$$\mathit{UInput}(x)$$

Actions

$\mathsf{GetLoan}(\mathbf{x})$:

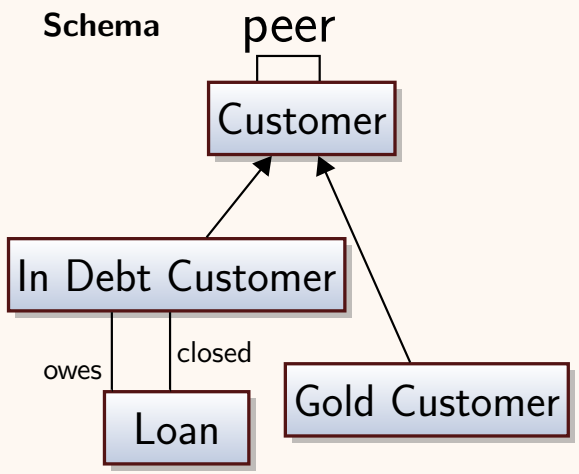
$$\begin{aligned}
\exists y.\mathsf{peer}(x, y) &\rightsquigarrow \{\mathsf{owes}(x, \mathit{UInput}(x))\},\\
\mathsf{Cust}(z) &\rightsquigarrow \{\mathsf{Cust}(z)\},\\
\mathsf{Loan}(z) &\rightsquigarrow \{\mathsf{Loan}(z)\},\\
\mathsf{InDebt}(z) &\rightsquigarrow \{\mathsf{InDebt}(z)\},\\
\mathsf{Gold}(z) &\rightsquigarrow \{\mathsf{Gold}(z)\}
\end{aligned}$$

DCDS, example



Data Layer

Schema



Instance

$\mathsf{Cust}(\mathsf{ann})$
$\mathsf{peer}(\mathsf{mark}, \mathsf{john})$
$\mathsf{Gold}(\mathsf{john})$
$\mathsf{owes}(\mathsf{mark}, \mathit{UInput}(\mathsf{mark}))$

Process Layer

Conditions

$$\mathsf{peer}(x, y) \wedge \mathsf{Gold}(y) \longmapsto \mathsf{GetLoan}(x)$$

Service Calls

$$\mathit{UInput}(x)$$

Actions

$\mathsf{GetLoan}(\mathbf{x})$:

$$\begin{aligned}
\exists y.\mathsf{peer}(x, y) &\rightsquigarrow \{\mathsf{owes}(x, \mathit{UInput}(x))\},\\
\mathsf{Cust}(z) &\rightsquigarrow \{\mathsf{Cust}(z)\},\\
\mathsf{Loan}(z) &\rightsquigarrow \{\mathsf{Loan}(z)\},\\
\mathsf{InDebt}(z) &\rightsquigarrow \{\mathsf{InDebt}(z)\},\\
\mathsf{Gold}(z) &\rightsquigarrow \{\mathsf{Gold}(z)\}
\end{aligned}$$

Deterministic services semantics - via transition systems

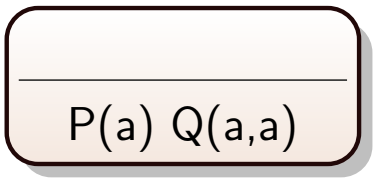
$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a, a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

$$\mathcal{I} = \{P(a), Q(a, a)\}$$

Deterministic services semantics - via transition systems

$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

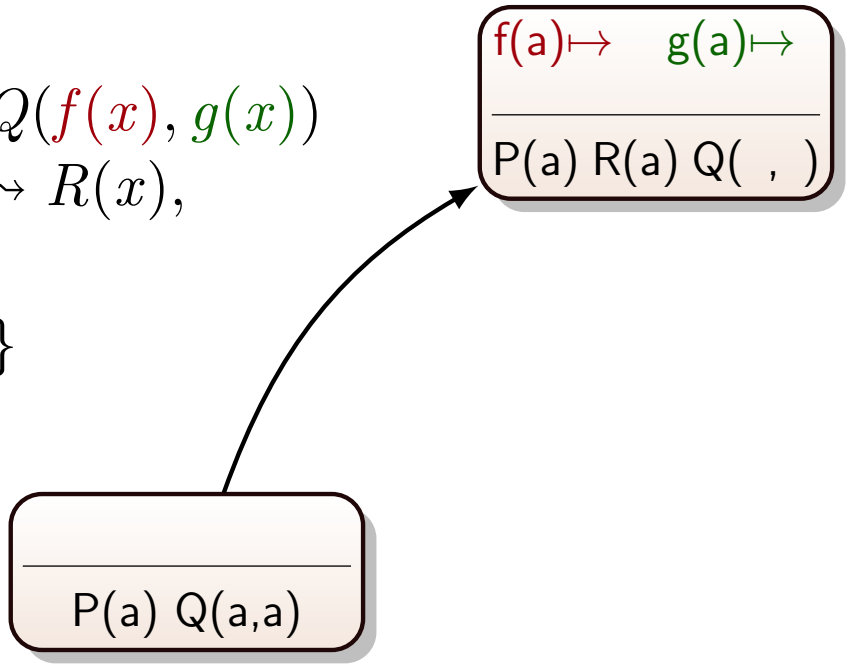
$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Deterministic services semantics - via transition systems

$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

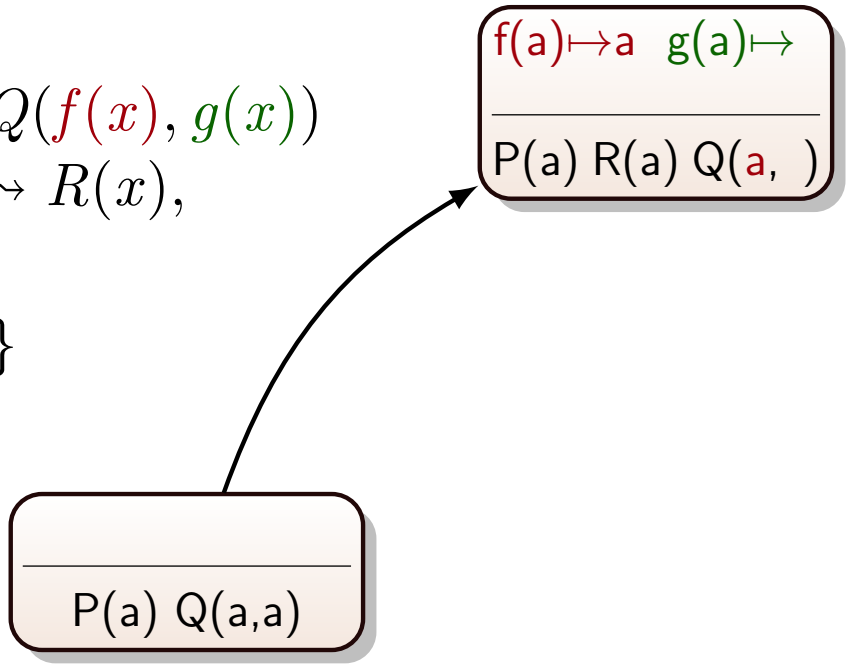
$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Deterministic services semantics - via transition systems

$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

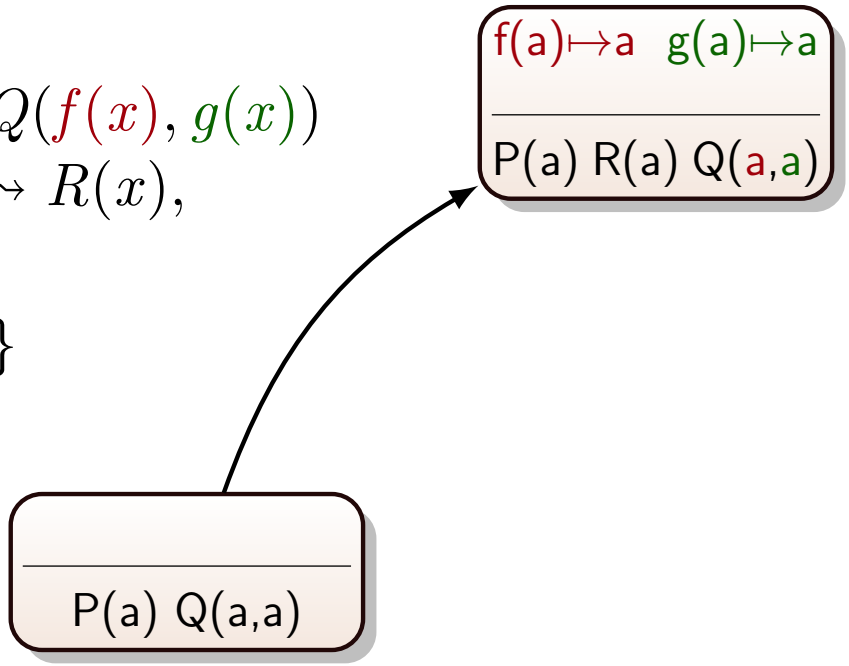
$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Deterministic services semantics - via transition systems

$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

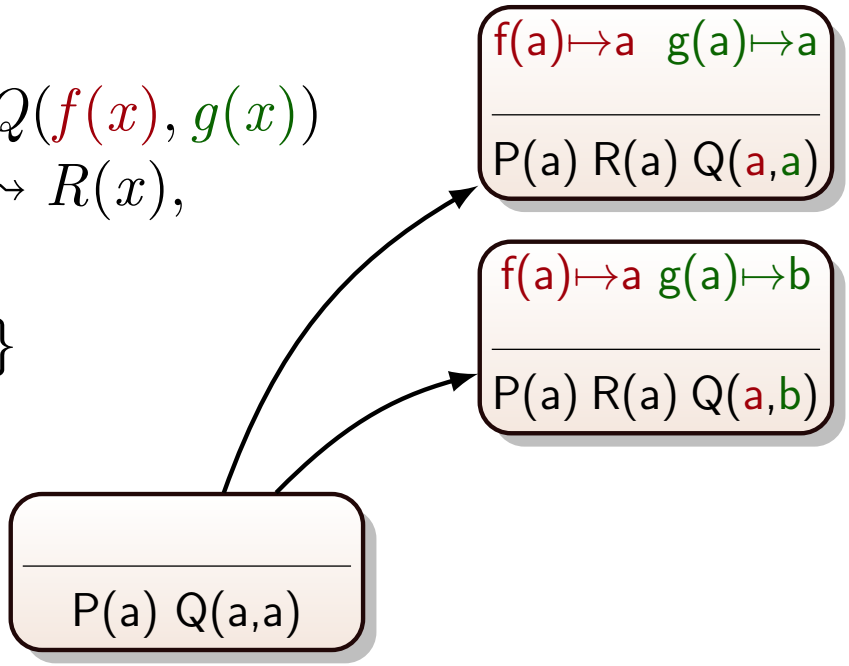
$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Deterministic services semantics - via transition systems

$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

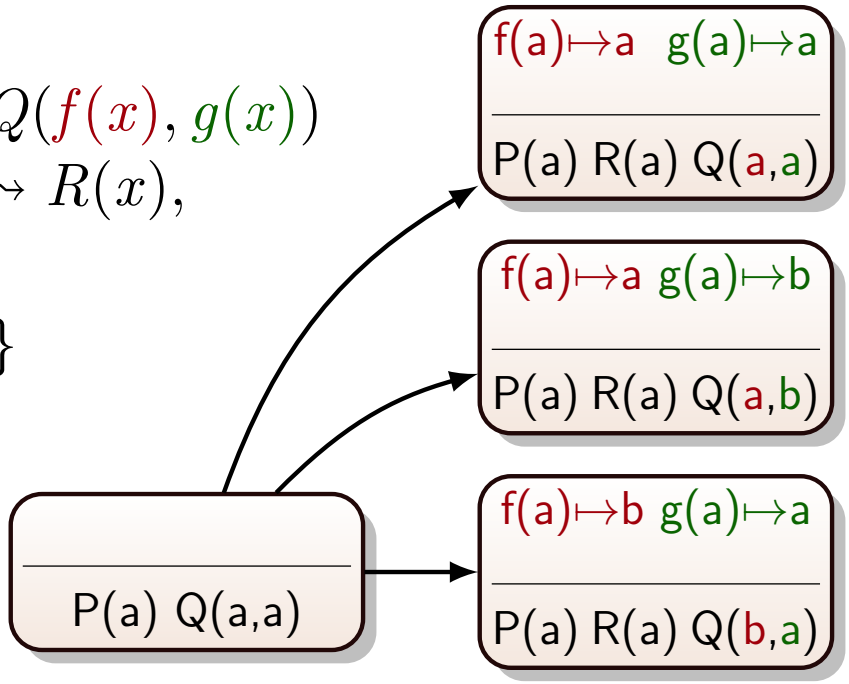
$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Deterministic services semantics - via transition systems

$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

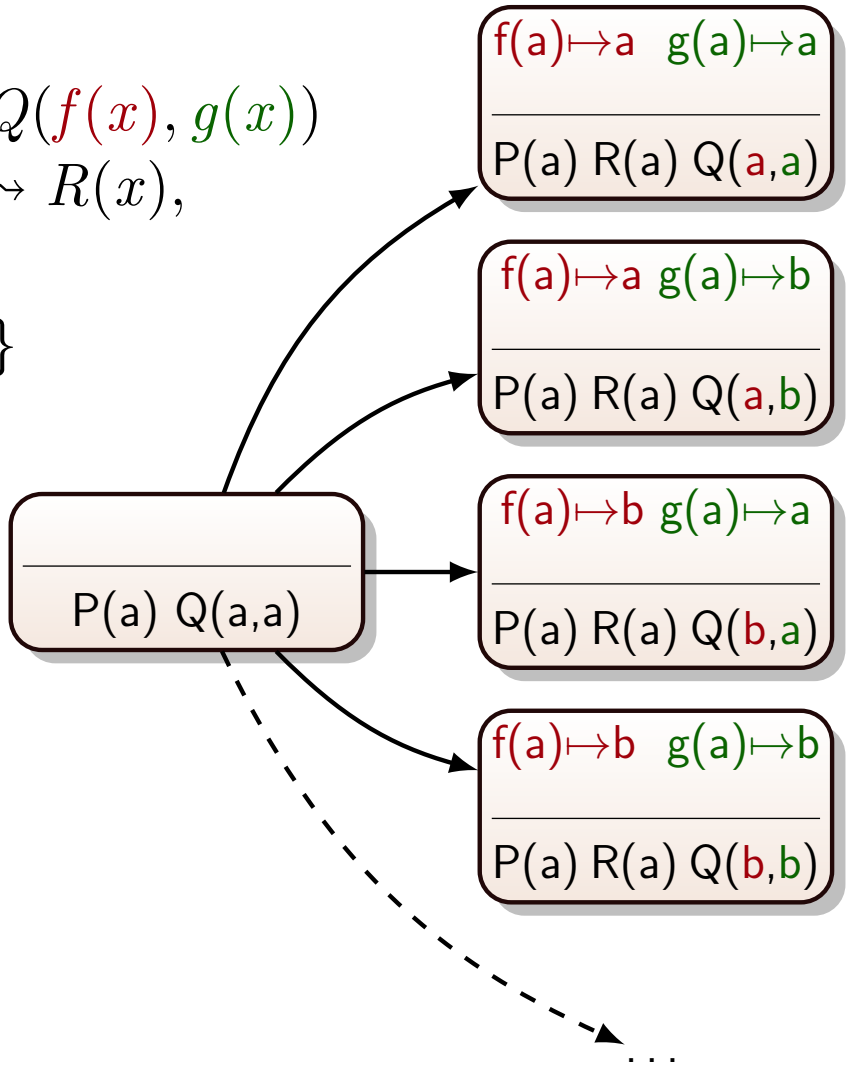
$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Deterministic services semantics - via transition systems

$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

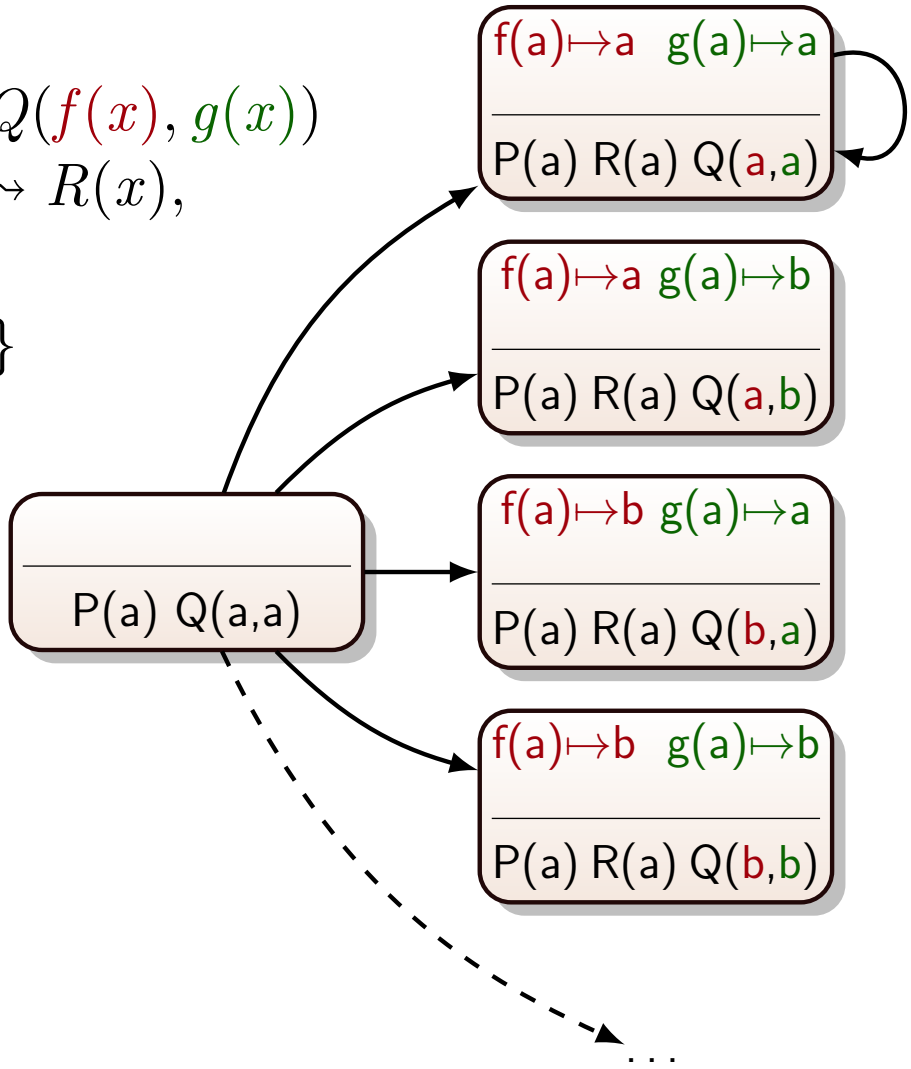
$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Deterministic services semantics - via transition systems

$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

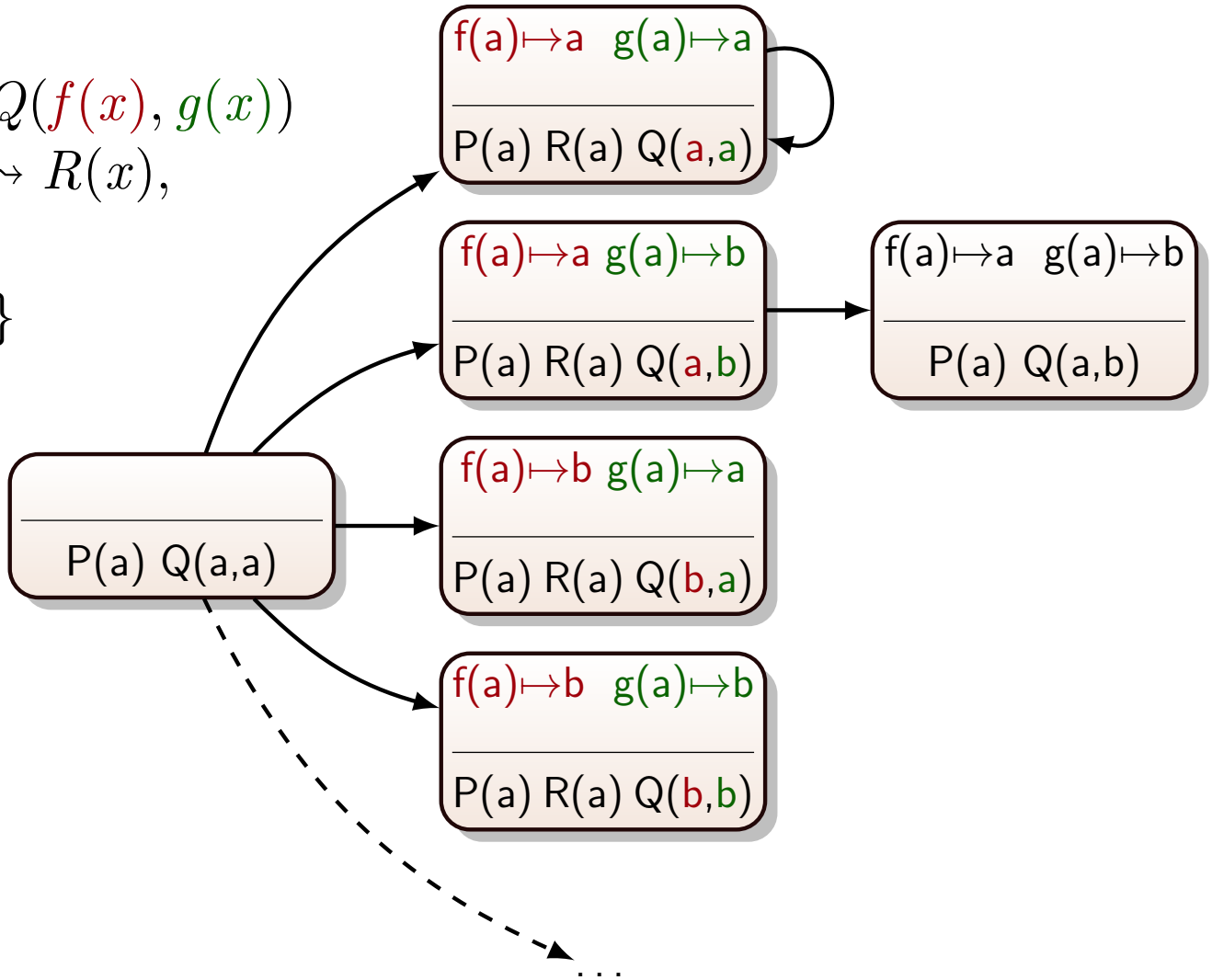
$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Deterministic services semantics - via transition systems

$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

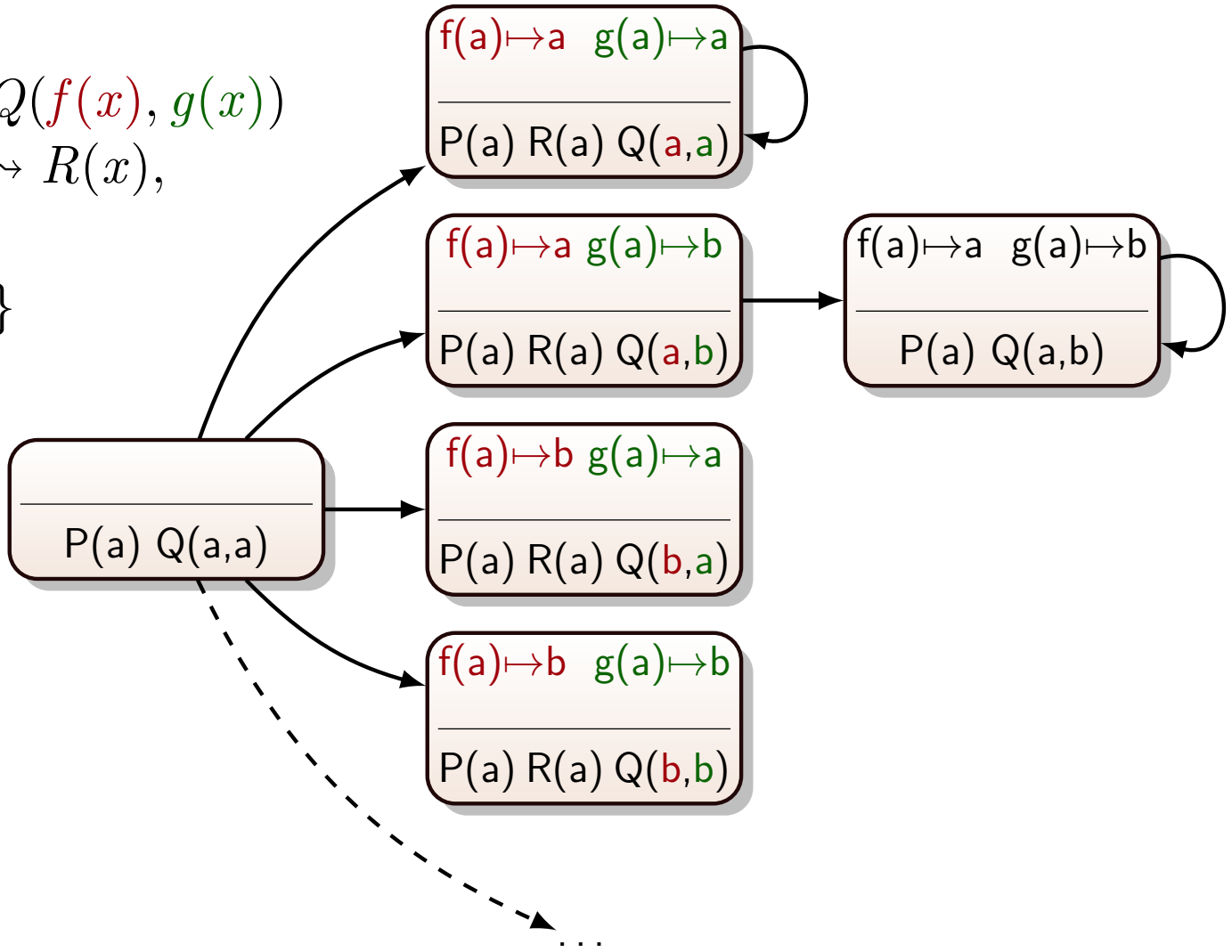
$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Deterministic services semantics - via transition systems

$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

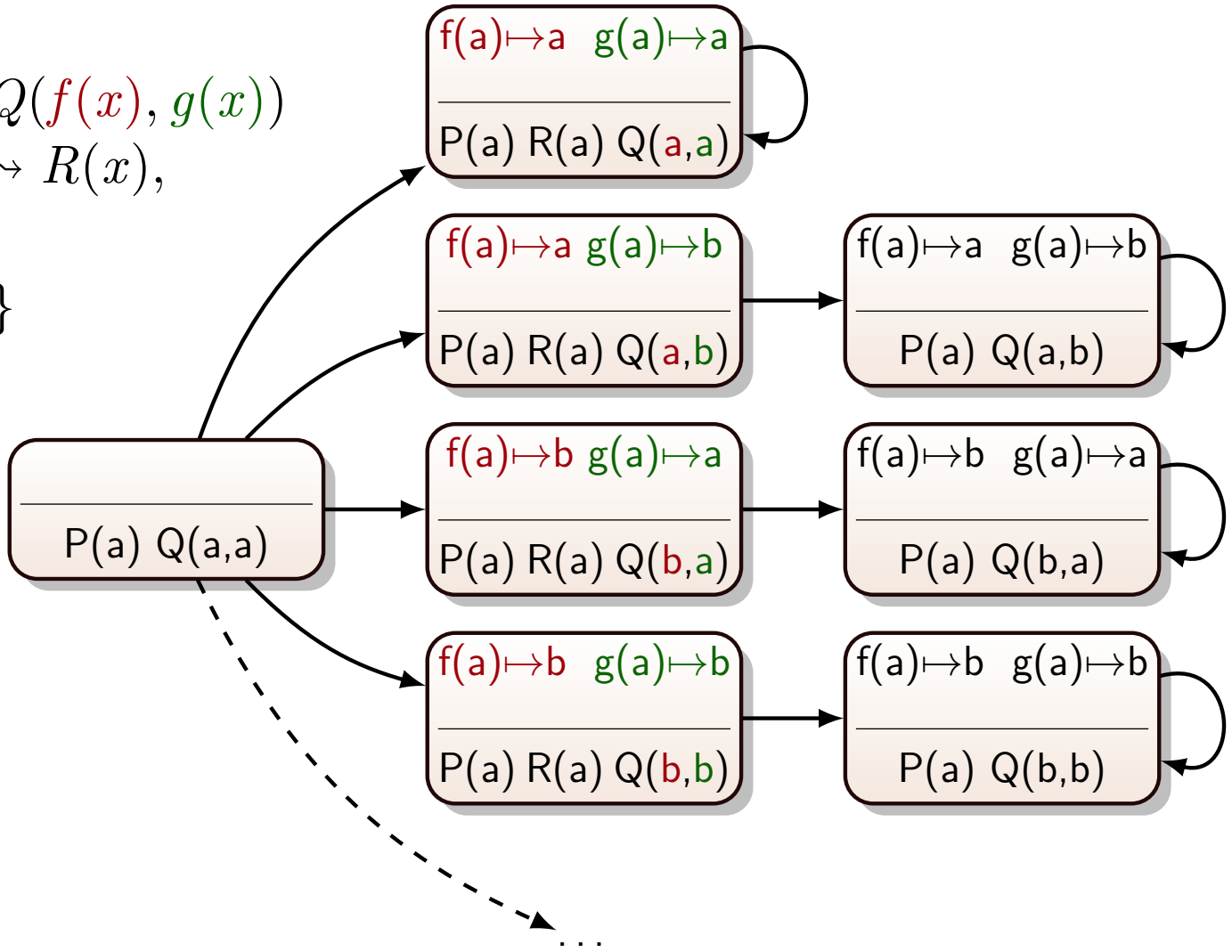
$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Deterministic services semantics - via transition systems

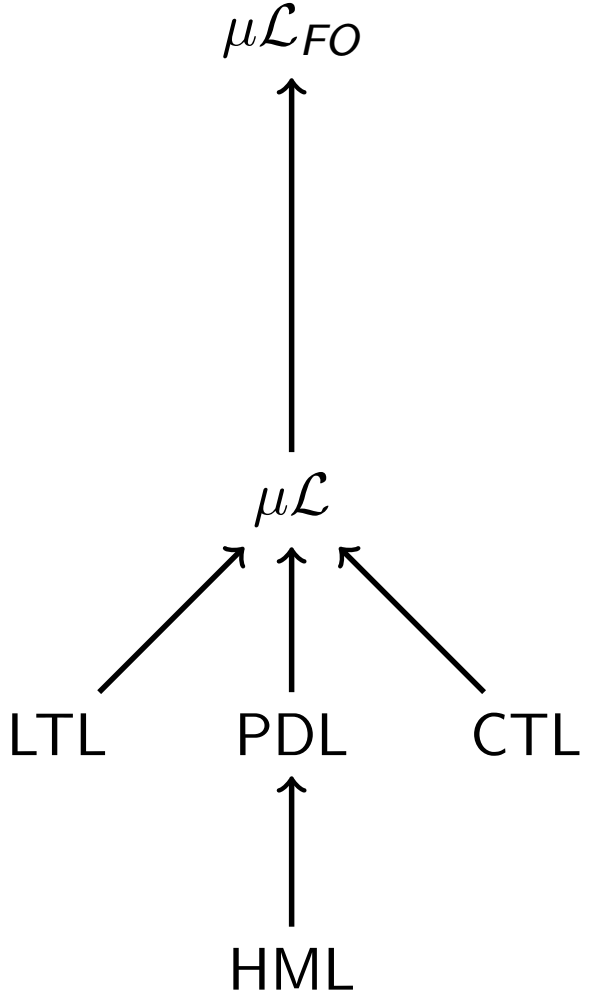
$$\begin{cases} P(x) \rightsquigarrow P(x) \wedge Q(f(x), g(x)) \\ Q(a,a) \wedge P(x) \rightsquigarrow R(x), \end{cases}$$

$$\mathcal{I} = \{P(a), Q(a,a)\}$$



Verification formalisms

- We propose different FO variants of μ-calculus.
- $\mu\mathcal{L}$ is not expressive enough to compare over time objects created by the process.
- Verification of $\mu\mathcal{L}_{FO}$ is undecidable, even for very restricted DCDSs!

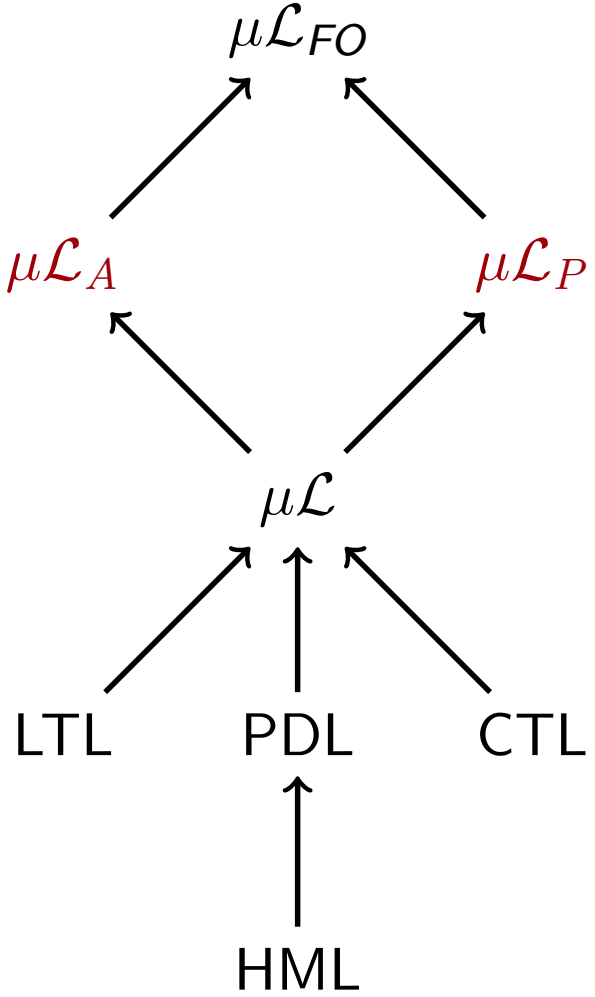


Verification formalisms

- We propose different FO variants of μ-calculus.
- $\mu\mathcal{L}$ is not expressive enough to compare over time objects created by the process.
- Verification of $\mu\mathcal{L}_{FO}$ is undecidable, even for very restricted DCDSs!

We introduce:

$\mu\mathcal{L}_P$ and $\mu\mathcal{L}_A$ as extensions of $\mu\mathcal{L}$ with (restricted) first order quantification.

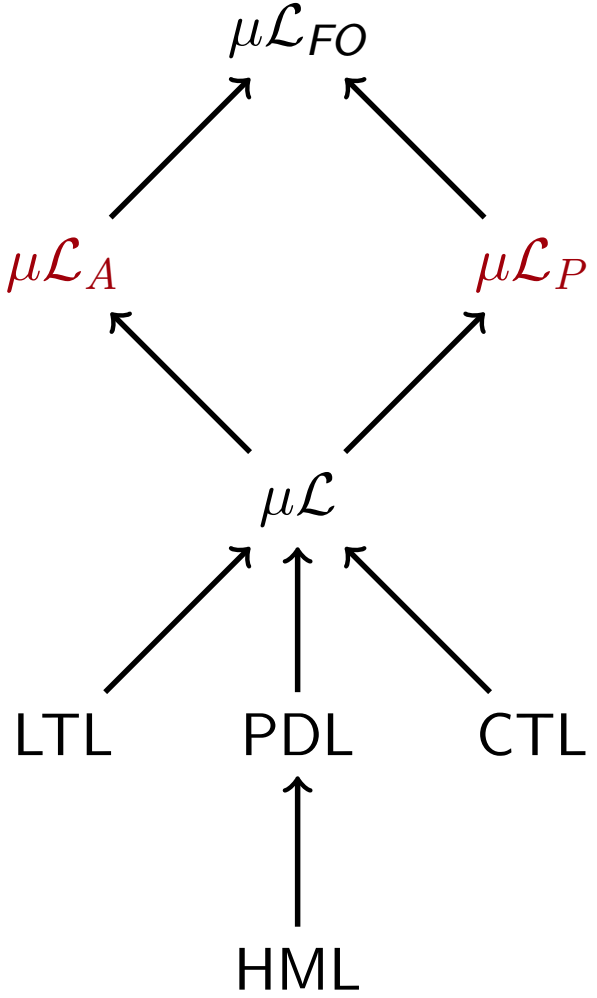


Verification formalisms

- We propose different FO variants of μ-calculus.
- $\mu\mathcal{L}$ is not expressive enough to compare over time objects created by the process.
- Verification of $\mu\mathcal{L}_{FO}$ is undecidable, even for very restricted DCDSs!

We introduce:

$\mu\mathcal{L}_P$ and $\mu\mathcal{L}_A$ as extensions of $\mu\mathcal{L}$ with (restricted) first order quantification.



Example in $\mu\mathcal{L}$: A Liveness property:

$$\mu Z.([\exists x.\mathsf{Gold}(x) \wedge \mathsf{InDebt}(x)] \vee \langle - \rangle Z)$$

$$\equiv \; F\,[\exists x.\mathsf{Gold}(x) \wedge \mathsf{InDebt}(x)]$$

Our results for the Relational DCDS setting

- The schema is a **relational schema**.
- States are **relational instances**.
- DCDSs can interact with external (non)deterministic services.

	DETERMINISTIC SERVICES				NONDETERMINISTIC SERVICES			
	$\mu\mathcal{L}_{FO}$		$\mu\mathcal{L}_{A}$	$\mu\mathcal{L}_{P}$		$\mu\mathcal{L}_{FO}$	$\mu\mathcal{L}_{A}$	$\mu\mathcal{L}_{P}$
unrestricted	**U**	$\leftarrow$	**U**	**U**	*unrestricted*	**U**	**U**	**U**
						$\uparrow$	$\uparrow$	
bounded-run	**?**		**D**	**D**	*bounded-state*	**U**	**U**	**D**

D: Verification is decidable **U**: Verification is undecidable

- The bounded-run and bounded-state conditions are undecidable.
- We study sufficient conditions based on variants of acyclicity.

Our results for the Semantic DCDS setting

- Schema is a TBox expressed in *DL-Lite*$_{\mathcal{A}}$.
- States are ABoxes.
- Queries are expressed in a variant of epistemic logic.

	$\mu\mathcal{L}_{FO}$	$\mu\mathcal{L}_{A}$	$\mu\mathcal{L}_{P}$
unrestricted	**U**	**U**	**U**
	$\uparrow$		
bounded-run	**U**	**D**	**D**

$\leftarrow$ $\leftarrow$ $\rightarrow$

D: Verification is decidable

U: Verification is undecidable

Next steps

- Enrich the setting to cover arithmetic operations.
- Enrich the setting with multiple artifacts, and study artifact creation/removal.
- Find classes of business process systems that correspond to our syntactic restrictions.
- Connect and compare our settings with other existing proposal for reasoning over data centeric dynamic systems.
- Implement a model checker on top of current model checkers for finite state systems.

Publications I


Bagheri Hariri, B., Calvanese, D., De Giacomo, G., and De Masellis, R. (2011a).
Verification of conjunctive-query based semantic artifacts.
In *Proc. of the 24th Int. Workshop on Description Logic (DL 2011)*, volume 745 of *ceur*.


Bagheri Hariri, B., Calvanese, D., De Giacomo, G., De Masellis, R., and Felli, P. (2011b).
Foundations of relational artifacts verification.
In *Proc. of 9th Int. Conf. on Business Process Management (BPM 2011)*, volume 6896 of *Lecture Notes in Computer Science*, pages 379–395.


Bagheri Hariri, B., Calvanese, D., De Giacomo, G., Deutsch, A., and Montali, M. (2012a).
Verification of relational data-centric dynamic systems with external services.
Submitted to an international conference.


Bagheri Hariri, B., Calvanese, D., De Giacomo, G., Deutsch, A., and Montali, M. (2012b).
Verification of relational data-centric dynamic systems with external services.
CoRR Technical Report arXiv:1203.0024, arXiv.org e-Print archive.


Bagheri Hariri, B., Calvanese, D., and etc (2012c).
Verification of description logic knowledge and action bases.
In *Proc. of the 20th European Conf. on Artificial Intelligence (ECAI 2012)*.


Calvanese, D., De Giacomo, G., Bagheri Hariri, B., and etc. (2012).
Techniques and tools for kab to manage action linkage with artifact layer.
ACSI Project Deliverable D2.4.1.

Thanks!

Questions, Comments, Suggestions ?