Verification of Data-Aware Processes

Verification Logics

Diego Calvanese, Marco Montali

Research Centre for Knowledge and Data (KRDB)
Free University of Bozen-Bolzano, Italy

29th European Summer School in Logic, Language, and Information
(ESSLLI 2017)
Toulouse, France – 17–28 July 2017

Outline

Verification logics

Temporal/dynamic logics expressing (un)desired properties about the dynamic system of interest.

- Branching-time logics are directly interpreted over the transition system.
  Notable examples: CTL, μ-calculus
- Linear-time logics are interpreted over the runs represented by the transition system. A property holds over the transition system if it is true in every run.
  Notable example: LTL

We consider in particular LTL and μ-calculus ($\mu\mathcal{L}$) as two representative verification logics.

The verification logic $\mu\mathcal{L}$

Remember that $\mu\mathcal{L}$:

- Is equipped with two local temporal operators to move over one ($\langle - \rangle$) or all ($[-]$) successors of the current state.
- Employs fixpoint constructs to express sophisticated properties defined via induction or co-induction.
- Subsumes virtually all propositional temporal/dynamic logics, such as PDL, LTL, CTL, CTL*.

  Note: The encoding into $\mu\mathcal{L}$ of linear-time properties incurs in an exponential blow-up.

Verification logics for DCDSs

Propositional temporal logics do not suffice!

We need **first-order temporal logics**:

- To inspect data: **FO queries**
- To capture system dynamics: **temporal modalities**
- To track the evolution of objects: FO **quantification across** states

Example:

It is always the case that every order is eventually either cancelled or paid.

$$\mathbf{G}(\forall x.\mathit{Order}(x) \rightarrow \mathbf{F}(\mathit{State}(x, \texttt{cancelled}) \vee \mathit{State}(x, \texttt{paid})))$$

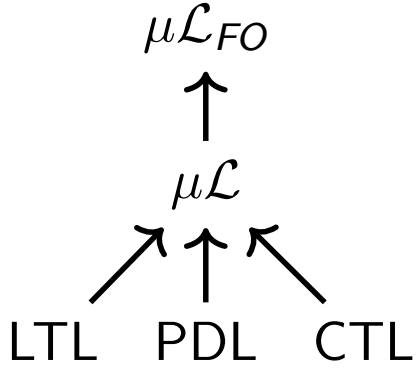
Outline

First-order μ-calculi for DCDSs

We employ variants of **first-order μ-calculus** ($\mu\mathcal{L}_{FO}$):

$$\Phi ::= Q \mid \neg\Phi \mid \Phi_1 \wedge \Phi_2 \mid \exists x.\Phi \mid \langle - \rangle \Phi \mid Z \mid \mu Z.\Phi$$

- Extends the propositional μ-calculus $\mu\mathcal{L}$ with first-order quantification.
- The first-order quantifiers range over all objects in the transition system (and not only over those in the current state or in the current run).



We also adopt the standard abbreviations, including:

- $[-]\Phi$ for $\neg\langle - \rangle \neg\Phi$
- $\nu Z.\Phi$ for $\neg\mu Z.\Phi_{[Z/\neg Z]}$

Example

$\forall x.\mathsf{Student}(x) \rightarrow \mu Z.((\exists y.\mathsf{Graduate}(x, y)) \vee \langle - \rangle Z)$

For each student x (in the current state), there exists an evolution that eventually leads to the graduation of x (with some final mark y).

Semantics of $\mu\mathcal{L}_{FO}$

$\mu\mathcal{L}_{FO}$ formulae: interpreted over RTS $\Upsilon = \langle \Delta, \mathcal{R}, S, s_0, db, \Rightarrow \rangle$, using valuations:

- v: individual variable valuation mapping each individual variable x to a value in Δ.
- V: predicate variable valuation, parameterized by v, and mapping each predicate variable Z to a subset $V(v, Z)$ of S.

Evaluation of a $\mu\mathcal{L}_{FO}$ formula Φ over Υ: given by an **extension function** $(\Phi)^{\Upsilon}_{(v,V)}$ mapping Φ to the set of states in S where Φ holds.

$$(\varphi)^{\Upsilon}_{(v,V)} = \{s \mid s \in S \text{ and } \langle db(s), v \rangle \models \varphi\}$$

$$(\neg\Phi)^{\Upsilon}_{(v,V)} = S \setminus (\Phi)^{\Upsilon}_{(v,V)}$$

$$(\Phi_1 \wedge \Phi_2)^{\Upsilon}_{(v,V)} = (\Phi_1)^{\Upsilon}_{(v,V)} \cap (\Phi_2)^{\Upsilon}_{(v,V)}$$

$$(\exists x.\Phi)^{\Upsilon}_{(v,V)} = \{s \in S \mid \exists d \in \Delta. s \in (\Phi)^{\Upsilon}_{(v,V)[x/d]}\}$$

$$(\langle - \rangle \Phi)^{\Upsilon}_{(v,V)} = \{s \in S \mid \exists s' \in S. s \Rightarrow s' \text{ and } s' \in (\Phi)^{\Upsilon}_{(v,V)}\}$$

$$(Z)^{\Upsilon}_{(v,V)} = V(v, Z)$$

$$(\mu Z.\Phi)^{\Upsilon}_{(v,V)} = \bigcap \{\mathcal{E} \subseteq S \mid (\Phi)^{\Upsilon}_{(v,V)[Z/\mathcal{E}]} \subseteq \mathcal{E}\}$$

stands for (v', V), where v' is as v except that $v'(x) = d$

stands for (v, V'), where V' is as V except that $V'(v, Z) = \mathcal{E}$

Model checking $\mu\mathcal{L}_{FO}$

When the $\mu\mathcal{L}_{FO}$ formula Φ is closed, $(\Phi)^{\Upsilon}_{(v,V)}$ does not depend on the valuations v and V, and we simply denote it as $(\Phi)^{\Upsilon}$.

Model checking an RTS

Input:

- an RTS $\Upsilon = \langle \Delta, \mathcal{R}, S, s_0, db, \Rightarrow \rangle$
- a closed $\mu\mathcal{L}_{FO}$ formula Φ

Output: yes, iff $s_0 \in (\Phi)^{\Upsilon}$

In this case, we write $\Upsilon \models \Phi$.

Model checking a DCDS

Input:

- a DCDS $\mathcal{X}$
- a closed $\mu\mathcal{L}_{FO}$ formula Φ

Output: yes, iff $\Upsilon_{\mathcal{X}} \models \Phi$

In this case, we write $\mathcal{X} \models \Phi$.

Outline

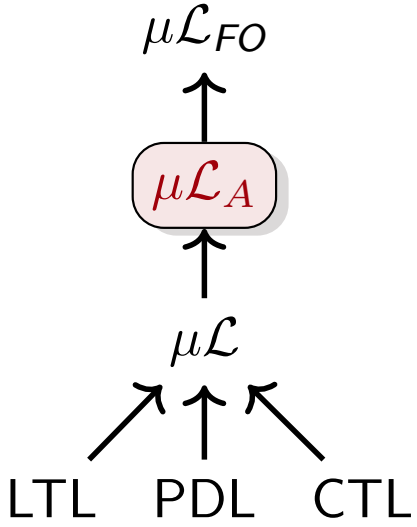
History-preserving μ-calculus ($\mu\mathcal{L}_A$)

Active-domain quantification: restricted to those individuals *present in the current database*.

$$\exists x.\Phi \quad \rightsquigarrow \quad \exists x.\textsc{live}(x) \wedge \Phi$$
$$\forall x.\Phi \quad \rightsquigarrow \quad \forall x.\textsc{live}(x) \rightarrow \Phi$$

where $\textsc{live}(x)$ states that x is present in the current active domain (easily expressible in FO).

Note: $\mu\mathcal{L}_A$ is a syntactic restriction of $\mu\mathcal{L}_{FO}$.



Example

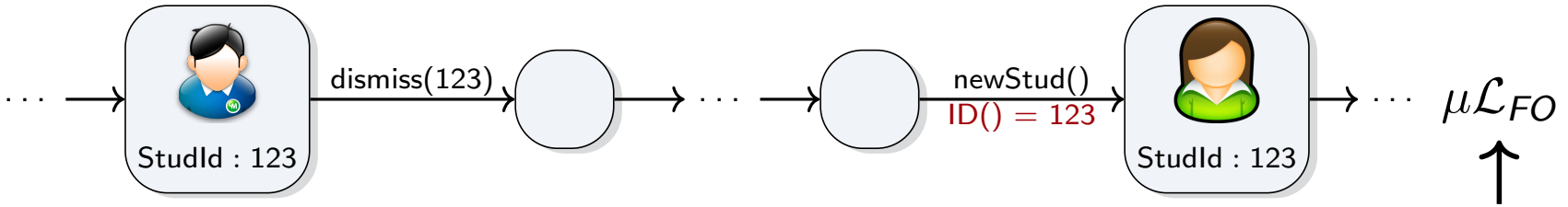
$$\nu W.(\forall x.\textsc{live}(x) \wedge \mathsf{Student}(x) \rightarrow \mu Z.(\exists y.\textsc{live}(y) \wedge \mathsf{Graduate}(x,y) \vee \langle - \rangle Z) \wedge [-]W)$$

Along every path, it is always true, for each student x, that there exists an evolution eventually leading to a graduation of the student (with some final mark y).

Note: No guarantee that all such students graduate within the same run.

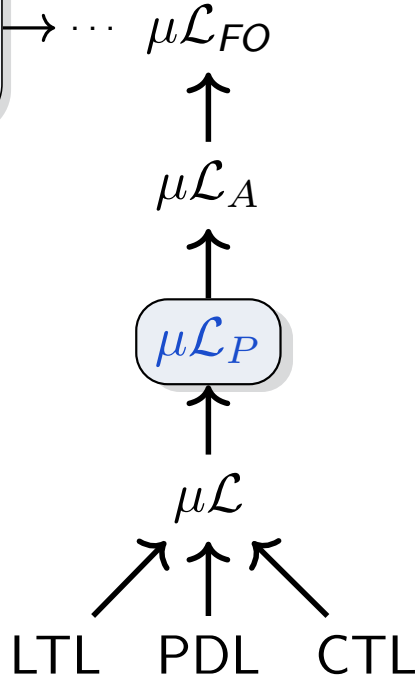
Persistence-preserving μ-calculus ($\mu\mathcal{L}_P$)

In some cases, objects maintain their identity only if they **persist** in the active domain (cf. business artifacts and their IDs).



$\mu\mathcal{L}_P$ restricts $\mu\mathcal{L}_A$ to **quantification over persisting objects only**, i.e., objects that *continue* to be LIVE.

$$
\begin{aligned}
\exists x.\Phi &\rightsquigarrow \exists x.\textsc{live}(x) \wedge \Phi \\
\forall x.\Phi &\rightsquigarrow \forall x.\textsc{live}(x) \rightarrow \Phi \\
\langle - \rangle \Phi(\vec{x}) &\rightsquigarrow \begin{cases} \textsc{live}(\vec{x}) \wedge \langle - \rangle \Phi(\vec{x}) & \text{(strong persistence)} \\ \textsc{live}(\vec{x}) \rightarrow \langle - \rangle \Phi(\vec{x}) & \text{(weak persistence)} \end{cases} \\
[-] \Phi(\vec{x}) &\rightsquigarrow \begin{cases} \textsc{live}(\vec{x}) \wedge [-] \Phi(\vec{x}) & \text{(strong persistence)} \\ \textsc{live}(\vec{x}) \rightarrow [-] \Phi(\vec{x}) & \text{(weak persistence)} \end{cases}
\end{aligned}
$$



Note: $\mu\mathcal{L}_P$ is a syntactic restriction of $\mu\mathcal{L}_A$.

Strong vs. weak persistence

Strong persistence: property falsified by an object that disappears

$$\nu W.(\forall x.\textsc{live}(x) \wedge \mathsf{Student}(x) \rightarrow \mu Z.(\exists y.\textsc{live}(y) \wedge \mathsf{Graduate}(x,y) \vee (\textsc{live}(x) \wedge \langle - \rangle Z)) \wedge [-]W)$$

Along every path, it is always true, for each student x, that there exists an evolution in which x **persists in the database until** she eventually graduates.

Weak persistence: property verified by an object that disappears

$$\nu W.(\forall x.\textsc{live}(x) \wedge \mathsf{Student}(x) \rightarrow \mu Z.(\exists y.\textsc{live}(y) \wedge \mathsf{Graduate}(x,y) \vee (\textsc{live}(x) \rightarrow \langle - \rangle Z)) \wedge [-]W)$$

Along every path, it is always true, for each student x, that there exists an evolution in which **either** x **does not persist, or** she eventually graduates.

Outline

First-order linear temporal logics for DCDSs

LTL-FO extends propositional LTL with the possibility of querying the system states using first-order formulas with quantification across:

$$\Phi ::= \varphi \mid \neg\Phi \mid \Phi_1 \wedge \Phi_2 \mid \exists x.\Phi \mid \mathbf{X}\,\Phi \mid \Phi_1\,\mathbf{U}\,\Phi_2$$

We also adopt the standard abbreviations, including:

- $\mathbf{F}\,\Phi$ for $\mathsf{true}\,\mathbf{U}\,\Phi$ (Φ holds in the future)
- $\mathbf{G}\,\Phi$ for $\neg\,\mathbf{F}\,\neg\Phi$ (Φ holds globally)

Example

$\forall x.\mathsf{Student}(x) \rightarrow \mathbf{F}\,\exists y.\mathsf{Graduate}(x, y)$

For each student x (in the current state), x will graduate sometimes in the future (with some final mark y).

Note: all encountered students graduate within the same run.

Semantics of LTL-FO

Formulae of LTL-FO$_A$ are interpreted over the infinite runs of a given **serial** RTS $\Upsilon = \langle \mathcal{R}, S, s_0, db, \Rightarrow \rangle$.

- **Serial RTS**: every state has at least one successor state.

An (infinite) run τ over Υ

is an infinite sequence $s_0 s_1 \cdots$ of states in S, where the first state of the sequence corresponds to the initial state of Υ, and for every $i \in \mathbb{N}$, it is true that $s_i \Rightarrow s_{i+1}$. Given $j \in \mathbb{N}$, by $\tau(j)$ we denote the j-th state s_j of τ.

We inductively define when τ satisfies an LTL-FO$_A$ formula Φ at position i under v, written $\langle \tau, i, v \rangle \models_{\text{LTL}} \Phi$

$\langle \tau, i, v \rangle \models_{\text{LTL}} \varphi$ if $\langle db(\tau(i)), v \rangle \models \varphi$

$\langle \tau, i, v \rangle \models_{\text{LTL}} \neg\Phi$ if it is not the case that $\langle \tau, i, v \rangle \models_{\text{LTL}} \Phi$

$\langle \tau, i, v \rangle \models_{\text{LTL}} \Phi_1 \wedge \Phi_2$ if $\langle \tau, i, v \rangle \models_{\text{LTL}} \Phi_1$ and $\langle \tau, i, v \rangle \models_{\text{LTL}} \Phi_2$

$\langle \tau, i, v \rangle \models_{\text{LTL}} \exists x.\Phi$ if there exists $d \in \Delta$ such that $\langle \tau, i, v[x/d] \rangle \models_{\text{LTL}} \Phi$

$\langle \tau, i, v \rangle \models_{\text{LTL}} \mathbf{X}\,\Phi$ if $\langle \tau, i+1, v \rangle \models_{\text{LTL}} \Phi$

$\langle \tau, i, v \rangle \models_{\text{LTL}} \Phi_1\,\mathbf{U}\,\Phi_2$ if there exists $k \geq i$ such that $\langle \tau, k, v \rangle \models_{\text{LTL}} \Phi_2$ and for every j, if $i \leq j < k$ then $\langle \tau, j, v \rangle \models_{\text{LTL}} \Phi_1$

Model checking LTL-FO

When the LTL-FO formula Φ is closed, the satisfaction relation does not depend on the valuation v, and we simply denote satisfaction as $\langle \tau, i \rangle \models_{\text{LTL}} \Phi$.

LTL model checking an RTS

Input:

- an RTS $\Upsilon = \langle \Delta, \mathcal{R}, S, s_0, db, \Rightarrow \rangle$
- a closed LTL-FO formula Φ

Output: yes, iff for every run τ over Υ, $\langle \tau, 0 \rangle \models_{\text{LTL}} \Phi$.

In this case, we write $\Upsilon \models_{\text{LTL}} \Phi$.

LTL Model checking a DCDS

Input:

- a DCDS $\mathcal{X}$
- a closed LTL-FO formula Φ

Output: yes, iff $\Upsilon_{\mathcal{X}} \models_{\text{LTL}} \Phi$.

In this case, we write $\mathcal{X} \models_{\text{LTL}} \Phi$.

First-order LTL with restricted quantification

History-preserving quantification: LTL-FO$_A$

FO quantification ranges over current active domain only:

$$\exists x.\Phi \quad \rightsquigarrow \quad \exists x.\textsc{live}(x) \wedge \Phi$$
$$\forall x.\Phi \quad \rightsquigarrow \quad \forall x.\textsc{live}(x) \rightarrow \Phi$$

Example: $\forall x.\textsc{live}(x) \wedge \mathsf{Customer}(x) \rightarrow \mathbf{F}\,\mathsf{Gold}(x)$

Persistence-preserving quantification: LTL-FO$_P$

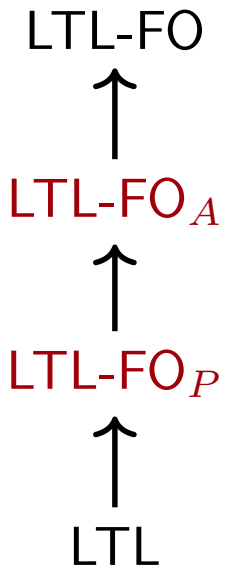
FO quantification ranges over persisting individuals only.

$$\exists x.\Phi \quad \rightsquigarrow \quad \exists x.\textsc{live}(x) \wedge \Phi$$
$$\forall x.\Phi \quad \rightsquigarrow \quad \forall x.\textsc{live}(x) \rightarrow \Phi$$

$$\mathbf{X}\,\Phi(\vec{x}) \quad \rightsquigarrow \quad \begin{cases} \textsc{live}(\vec{x}) \wedge \mathbf{X}\,\Phi(\vec{x}) & \text{(strong persistence)} \\ \textsc{live}(\vec{x}) \rightarrow \mathbf{X}\,\Phi(\vec{x}) & \text{(weak persistence)} \end{cases}$$

$$\Phi_1\,\mathbf{U}\,\Phi_2(\vec{x}) \quad \rightsquigarrow \quad \begin{cases} (\textsc{live}(\vec{x}) \wedge \Phi_1)\,\mathbf{U}\,\Phi_2(\vec{x}) & \text{(s.p.)} \\ (\textsc{live}(\vec{x}) \wedge \Phi_1)\,\mathbf{U}(\textsc{live}(\vec{x}) \rightarrow \Phi_2(\vec{x})) & \text{(w.p.)} \end{cases}$$

Example: $\forall x.(\textsc{live}(x) \wedge \mathsf{Gold}(x)) \rightarrow \neg(\textsc{live}(x)\,\mathbf{U}\,\neg\mathsf{Gold}(x))$



LTL with persistence-preserving quantification – Example

Consider:

$\forall x.\mathsf{Gold}(x) \rightarrow \mathbf{G}\,\mathsf{Gold}(x)$

$\forall x.\mathsf{Gold}(x) \rightarrow \neg\,\mathbf{F}\,\neg\mathsf{Gold}(x)$

$\forall x.\mathsf{Gold}(x) \rightarrow \neg(\mathsf{true}\,\mathbf{U}\,\neg\mathsf{Gold}(x))$

With strong persistence:

$\forall x.\textsc{live}(x) \rightarrow (\mathsf{Gold}(x) \rightarrow \neg(\textsc{live}(x)\,\mathbf{U}\,\neg\mathsf{Gold}(x)))$

$\forall x.(\textsc{live}(x) \wedge \mathsf{Gold}(x)) \rightarrow \neg(\textsc{live}(x)\,\mathbf{U}\,\neg\mathsf{Gold}(x))$

With weak persistence:

$\forall x.\textsc{live}(x) \rightarrow (\mathsf{Gold}(x) \rightarrow \neg(\textsc{live}(x)\,\mathbf{U}(\textsc{live}(x) \rightarrow \neg\mathsf{Gold}(x))))$

$\forall x.(\textsc{live}(x) \wedge \mathsf{Gold}(x)) \rightarrow \neg(\textsc{live}(x)\,\mathbf{U}(\textsc{live}(x) \rightarrow \neg\mathsf{Gold}(x)))$