Runtime Monitoring for Hennessy-Milner Logic with Recursion over Systems with Data

Ongoing Work

Léo Exibard

Thursday, June 30$^{\text{th}}$, 2022

Logic Colloquium 2022, Reykjavík

Context

Formal Verification

Ensure a system satisfies a property

Non-terminating Systems

$\boxed{\text{System}} \rightsquigarrow$ Trace $ab\cdots \in A^\omega$

Property of the *system* $\to$ branching time modal logic

Example

The first token does not appear again:

$$\bigwedge_{a\in A}[a]\max X.\Big([a]\mathtt{ff} \wedge \bigwedge_{b\neq a}[b]X\Big)$$

Runtime Monitoring

➔ Check whether the property holds *along the execution*

Runtime Monitoring

Monitor

Processes trace to raise a *verdict*:

- Satisfaction `yes`
- Violation `no`

Irrevocability

Once produced, a verdict cannot *change*

Questions

- What properties are monitorable (for a given monitor model)?
- How to synthesise monitors from formulas?

Hennessy-Milner Logic with Recursion

Syntax

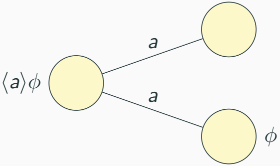
$$\phi, \psi \in \mathit{recHML} ::= \texttt{tt} \mid \langle a \rangle \phi \mid \phi \vee \psi \mid \mathit{min}\ X.\phi \mid X$$
$$\mid \texttt{ff} \mid [a]\phi \mid \phi \wedge \psi \mid \mathit{max}\ X.\phi$$

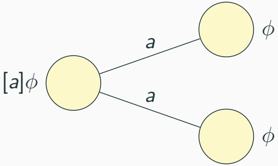
Recursion: Fixpoints

- min: least fixpoint
- max: greatest fixpoint

Branching Time

Models = processes





Monitoring recHML

Monitors

$$m, n \in Mon ::= v \in Vrd \quad \mid a.m \mid m + n \quad \mid rec\ X.m \mid X$$

$$v \in Vrd ::= \mathrm{end} \mid \mathrm{no} \mid \mathrm{yes}$$

Monitorable Fragments

$$\phi, \psi \in sHML ::= \mathtt{tt} \quad \mid \mathtt{ff} \mid [a]\phi \quad \mid \phi \wedge \psi \mid max\ X.\phi \quad \mid X \text{ (violation)}$$

$$\phi, \psi \in cHML ::= \mathtt{tt} \quad \mid \mathtt{ff} \mid \langle a \rangle\phi \quad \mid \phi \vee \psi \mid min\ X.\phi \quad \mid X \text{ (satisfaction)}$$

Monitor Synthesis

- Compositional (Francalanza, Aceto, and Ingólfsdóttir 2015)
- Monitors can be determinised (Aceto et al. 2020)

Temporal Logics over Systems with Data

Data Domains

Infinite set with decidable theory: $(\mathbb{N}, =)$, $(\mathbb{Q}, <)$, $(\Sigma^*, <_{\text{prec}})$

Linear Time

• Freeze LTL • $FO^2[<, \sim]$ • etc (see Demri and Quaas 2021)

Branching Time

Modal μ-calculus with:

- Freeze (Jurdzinski and Lazic 2007)
- Quantifiers (partly inspired from Groote and Mateescu 1998)

$$\begin{aligned}\phi, \psi \in \mathit{recHML} ::= \mathtt{tt} \mid \langle b(\star)\rangle\phi \mid \phi \vee \psi \mid \mathit{min}\, X.\phi \mid X(\vec{v}) \mid \exists x.\phi \\ \mid \mathtt{ff} \mid [b(\star)]\phi \mid \phi \wedge \psi \mid \mathit{max}\, X.\phi \mid \forall x.\phi\end{aligned}$$

$b(\star)$: quantifier-free FO formula

$Ex : \forall x\ [\star = x]\, \max X.\Big([\star = x]\mathtt{ff} \wedge [\star \neq x] X(x)\Big)$

Monitoring with Data

Monitors

$$m, n \in \mathit{Mon} ::= v \in \mathit{Vrd} \mid b(\star).m \mid \mathit{guess}\ x.m \mid m + n \mid \mathit{rec}\ X(\vec{v}).m \mid X(\vec{v})$$

$$v \in \mathit{Vrd} ::= \mathrm{end} \ \mid\ \mathrm{no} \ \mid\ \mathrm{yes}$$

Evaluating non-determinism

Run monitors in parallel

The 'guess' construct (Apt and Plotkin 1986)

- Guess the satisfaction (resp. violation) witness
  - Equality: accumulate forbidden values
  - Richer theories: accumulate constraints

Register Automata

Data-free setting: recHML $\equiv$ ω-regular

Register Automata (Kaminski and Francez 1994)

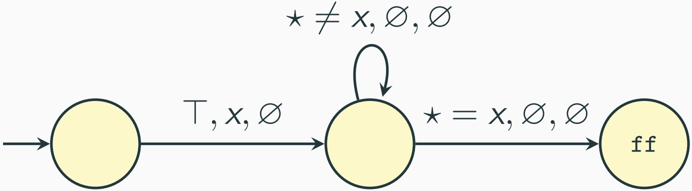
Finite automata with a finite set **R** of registers

- Store data
- Test register content

Transitions $q \xrightarrow{\phi, A, G} q'$

- $\varphi \in \mathrm{QF}(R, \star)$: test
- $A \subseteq R$: assignment
- $G \subseteq R$: guessing

$$\forall x\, [\star = x] \max X. \Big([\star = x]\mathtt{ff} \wedge [\star \neq x] X(x) \Big)$$



recHML and Register Automata

Theorem (work in progress)

recHML and register automata are equi-expressive. More precisely:

- recHML $\equiv$ alternating RA
- sHML $\equiv$ non-deterministic RA
- cHML $\equiv$ universal RA
- sHML$^{\mathsf{nf}}$ $\equiv$ deterministic RA (in particular, no guessing)

$$
\begin{tikzcd}
 & \mathrm{ARA} & \\
\mathrm{NRA} \arrow[ur, dash, "\subsetneq" description] \arrow[rr, dash, "\text{dual}" description] & & \mathrm{URA} \arrow[ul, dash, "\supsetneq" description] \\
 & \mathrm{DRA} \arrow[ul, dash, "\supsetneq" description] \arrow[ur, dash, "\subsetneq" description] &
\end{tikzcd}
$$

Consequences

Strict hierarchy between fragments

- → In particular, sHMLnf is not a normal form
- → Monitors do not determinise (need for parallelism)

Undecidability Results

- (semantic) membership to a fragment is undecidable
- Monitorability is undecidable

Non-maximality

The data values in the first block are pairwise distinct:

$$\{d_0 \dots d_n \# \cdots \mid \forall 0 \leq i < j \leq n, d_i \neq d_j\}$$

→ Violations can be detected by a NRA, not by a URA.

Conclusion

- Monitor synthesis extends to systems with data
- However, the logic is not as well-behaved...
- Leverage correspondence with register automata

Future Work

- Maximal fragments
- Fragments with efficient monitors
- First-order formulas in modalities
- How to evaluate accumulated constraints?