Symbolic Model checking of Relative Safety LTL properties

A. Bombardelli[1, 2] A. Cimatti[1] S. Tonetta[1] M. Zamboni[1]

[1]Fondazione Bruno Kessler
via Sommarive 18, Povo 38123, Italy

[2]University of Trento
via Sommarive 9, Povo 38123, Italy

Outline

Motivation

Motivation

Problem:

$$\mathcal{M} \models_{LTL} \varphi$$ ⚙ ⌛ ☹

Motivation

Problem:

$$\mathcal{M} \models_{LTL} \varphi$$

Idea: Can we reduce the problem to invariant checking?
Doable for safetyLTL (assuming absense of deadlocks/livelocks)

Motivation

Problem:

$$\mathcal{M} \models_{LTL} \varphi$$ ⚙ ⌛ ☹

Idea: Can we reduce the problem to invariant checking?
Doable for safetyLTL (assuming absense of deadlocks/livelocks)
Goal/Contribution: Generalize the approach for a larger fragment of LTL using relative safety

SafetyLTL to invariant and deadlocks/livelocks

$$\mathcal{M} \models \mathbf{G} q$$

Let's use invariant checking: INVAR q

SafetyLTL to invariant and deadlocks/livelocks

$$\mathcal{M} \models \mathbf{G} q$$

Let's use invariant checking: INVAR q

$$\pi_f := \overset{a \wedge b \wedge q}{\bullet} \rightarrow \overset{q}{\cdots} \rightarrow \overset{\neg q}{\bullet}$$

SafetyLTL to invariant and deadlocks/livelocks

$$\mathcal{M} \models \mathbf{G} q$$

Let's use invariant checking: INVAR q

$$\pi_f := \overset{a \wedge b \wedge q}{\bullet} \rightarrow \overset{q}{\ldots} \rightarrow \overset{\neg q}{\bullet}$$

Nice counterexample BUT it is finite! Let's extend it!

$$\overset{a \wedge b \wedge q}{\bullet} \rightarrow \overset{q}{\ldots} \rightarrow \overset{\neg q}{\bullet} \rightarrow \overset{a \wedge b \wedge q}{\bullet} \rightarrow \underbrace{X}_{\text{Deadlock}} \quad ☹$$

SafetyLTL to invariant and deadlocks/livelocks

$$\mathcal{M} \models \mathbf{G} q$$

Let's use invariant checking: INVAR q

$$\pi_f := \overset{a \wedge b \wedge q}{\bullet} \rightarrow \overset{q}{\cdots} \rightarrow \overset{\neg q}{\bullet}$$

Nice counterexample BUT it is finite! Let's extend it!

$$\overset{a \wedge b \wedge q}{\bullet} \rightarrow \overset{q}{\cdots} \rightarrow \overset{\neg q}{\bullet} \rightarrow \overset{a \wedge b \wedge q}{\bullet} \rightarrow \underbrace{X}_{\text{Deadlock}} \quad ☹$$

We need to get rid of deadlocks!

SafetyLTL to invariant and deadlocks/livelocks

$$\mathcal{M} \models \mathbf{G} q$$

Let's use invariant checking: INVAR q

$$\pi_f := \overset{a\wedge b\wedge q}{\bullet} \rightarrow \overset{q}{\ldots} \rightarrow \overset{\neg q}{\bullet}$$

Nice counterexample BUT it is finite! Let's extend it!

$$\overset{a\wedge b\wedge q}{\bullet} \rightarrow \overset{q}{\ldots} \rightarrow \overset{\neg q}{\bullet} \rightarrow \overset{a\wedge b\wedge q}{\bullet} \rightarrow \underbrace{X}_{\text{Deadlock}} \quad ☹$$

We need to get rid of deadlocks!

Deadlock/livelock: A deadlock (livelock) state is a state from which no (fair) state is reachable

Beyond safetyLTL

Bounded response: $\varphi_{BR} := \mathbf{G}(in \wedge t = p \rightarrow \mathbf{F}(t \leq p + 5 \wedge out))$

Beyond safetyLTL

Bounded response: $\varphi_{BR} := \mathbf{G}(in \wedge t = p \rightarrow \mathbf{F}(t \leq p + 5 \wedge out))$

$$\alpha := \underbrace{\mathbf{G}(t' \geq t)}_{\text{Weak monotonicity}} \wedge \underbrace{\mathbf{GF}(t' - t \geq \epsilon)}_{\text{non-zenoness}}$$

Beyond safetyLTL

Bounded response: $\varphi_{BR} := \mathbf{G}(in \wedge t = p \rightarrow \mathbf{F}(t \leq p + 5 \wedge out))$

$$\alpha := \underbrace{\mathbf{G}(t' \geq t)}_{\text{Weak monotonicity}} \wedge \underbrace{\mathbf{GF}(t' - t \geq \epsilon)}_{\text{non-zenoness}}$$

Counterexample shape of $\alpha \rightarrow \varphi_{BR}$: $\underbrace{\overset{in \wedge t=p}{\bullet} \rightarrow \overset{t=p+2}{\bullet} \rightarrow \overset{t=p+5.01}{\bullet}}_{\text{Bad prefix}} \rightarrow \ldots$

Beyond safetyLTL

Bounded response: $\varphi_{BR} := \mathbf{G}(in \wedge t = p \to \mathbf{F}(t \leq p + 5 \wedge out))$

$$\alpha := \underbrace{\mathbf{G}(t' \geq t)}_{\text{Weak monotonicity}} \wedge \underbrace{\mathbf{GF}(t' - t \geq \epsilon)}_{\text{non-zenoness}}$$

Counterexample shape of $\alpha \to \varphi_{BR}$: $\underbrace{\overset{in \wedge t=p}{\bullet} \to \overset{t=p+2}{\bullet} \to \overset{t=p+5.01}{\bullet}}_{\text{Bad prefix}} \to \ldots$

It would be nice to be able to verify this property using invariants and getting rid of deadlocks/livelocks!

Contributions

1. Reduce safetyLTL to invariant checking $+$ Block deadlocks
2. Extend safety LTL to invariant with relative safety to cover a larger fragment $\alpha_S \wedge \alpha_L \rightarrow \varphi$
3. Generalize (ii) blocking unfair counterexamples

Background

Properties classification

Properties classification

Safety property:

$P_S \subseteq \Sigma^\omega$ is a *safety property* iff $\forall \pi \in \Sigma^\omega$ s.t. $\pi \not\models P_S$, $\exists \pi_f \in Pref(\pi)$ s.t. $\forall \pi^\omega \in \Sigma^\omega : \pi_f \pi^\omega \not\models P_S$.

$$\underbrace{\bullet \to \bullet \to \bullet}_{\text{"bad prefix"}\ \pi_f} \to \bullet \to \bullet \to \ldots \notin P_S$$

Properties classification

Safety property:

$P_S \subseteq \Sigma^\omega$ is a *safety property* iff $\forall \pi \in \Sigma^\omega$ s.t. $\pi \not\models P_S$, $\exists \pi_f \in Pref(\pi)$ s.t. $\forall \pi^\omega \in \Sigma^\omega : \pi_f \pi^\omega \not\models P_S$.

$$\underbrace{\bullet \rightarrow \bullet \rightarrow \bullet}_{\text{"bad prefix"}\ \pi_f} \rightarrow \bullet \rightarrow \bullet \rightarrow \ldots \notin P_S$$

Liveness property:

$P_L \subseteq \Sigma^\omega$ is a *liveness property* iff $\forall \pi \in \Sigma^\omega, \forall \pi_f \in Pref(\pi)\ \exists \pi^\omega \in \Sigma^\omega : \pi_f \pi^\omega \models P_L$.

$$\underbrace{\ldots}_{\text{Any prefix}} \rightarrow \underbrace{\bullet \rightarrow \bullet \rightarrow \ldots}_{\pi^\omega} \in P_L$$

Properties classification

Safety property:

$P_S \subseteq \Sigma^\omega$ is a *safety property* iff $\forall \pi \in \Sigma^\omega$ s.t. $\pi \not\models P_S$, $\exists \pi_f \in Pref(\pi)$ s.t. $\forall \pi^\omega \in \Sigma^\omega : \pi_f \pi^\omega \not\models P_S$.

$$\underbrace{\bullet \rightarrow \bullet \rightarrow \bullet}_{\text{"bad prefix"} \pi_f} \rightarrow \bullet \rightarrow \bullet \rightarrow \ldots \notin P_S$$

Liveness property:

$P_L \subseteq \Sigma^\omega$ is a *liveness property* iff $\forall \pi \in \Sigma^\omega, \forall \pi_f \in Pref(\pi)\ \exists \pi^\omega \in \Sigma^\omega : \pi_f \pi^\omega \models P_L$.

$$\underbrace{\ldots}_{\text{Any prefix}} \rightarrow \underbrace{\bullet \rightarrow \bullet \rightarrow \ldots}_{\pi^\omega} \in P_L$$

Generic property:

$$P = P_S \cap P_L$$

LTL

Syntax: $\phi := \overbrace{\top \mid p \mid \phi \vee \phi \mid \neg\phi}^{\text{Propositional}} \mid \overbrace{\mathbf{X}\phi \mid \phi\mathbf{U}\phi}^{\text{future}} \mid \overbrace{\mathbf{Y}\phi \mid \phi\mathbf{S}\phi}^{\text{Past}}$

LTL

Syntax: $\phi := \overbrace{\top \mid p \mid \phi \vee \phi \mid \neg\phi}^{\text{Propositional}} \mid \overbrace{\mathbf{X}\phi \mid \phi\mathbf{U}\phi}^{\text{future}} \mid \overbrace{\mathbf{Y}\phi \mid \phi\mathbf{S}\phi}^{\text{Past}}$

Abbreviations: $\bot := \neg\top$ $\quad \mathbf{F}\phi := \top\mathbf{U}\phi$ $\quad \phi_1\mathbf{R}\phi_2 := \neg(\neg\phi_1\mathbf{U}\neg\phi_2)$ $\quad \mathbf{G}\phi := \bot\mathbf{R}\phi$
$\mathbf{Z}\phi := \neg\mathbf{Y}\neg\phi$ $\quad \phi_1\mathbf{T}\phi_2 := \neg(\neg\phi_1\mathbf{S}\neg\phi_2)$ $\quad \mathbf{H}\phi := \bot\mathbf{T}\phi$ $\quad \mathbf{O}\phi := \top\mathbf{S}\phi$

Syntax: $\phi := \overbrace{\top \mid p \mid \phi \vee \phi \mid \neg\phi}^{\text{Propositional}} \mid \overbrace{\mathbf{X}\phi \mid \phi\mathbf{U}\phi}^{\text{future}} \mid \overbrace{\mathbf{Y}\phi \mid \phi\mathbf{S}\phi}^{\text{Past}}$

Abbreviations: $\bot := \neg\top \quad \mathbf{F}\phi := \top\mathbf{U}\phi \quad \phi_1\mathbf{R}\phi_2 := \neg(\neg\phi_1\mathbf{U}\neg\phi_2) \quad \mathbf{G}\phi := \bot\mathbf{R}\phi$

$\mathbf{Z}\phi := \neg\mathbf{Y}\neg\phi \quad \phi_1\mathbf{T}\phi_2 := \neg(\neg\phi_1\mathbf{S}\neg\phi_2) \quad \mathbf{H}\phi := \bot\mathbf{T}\phi \quad \mathbf{O}\phi := \top\mathbf{S}\phi$

Semantics (graphical repr):

$$\varphi\mathbf{U}\psi \quad \overset{\varphi}{\bullet} \to \overset{\varphi}{\ldots} \to \overset{\psi}{\bullet} \to \overset{\omega}{\ldots}$$

$$\mathbf{X}\varphi \quad \bullet \to \overset{\varphi}{\bullet} \to \overset{\omega}{\ldots}$$

$$\varphi\mathbf{S}\psi \quad \cdots \to \overset{\psi}{\bullet} \to \overset{\varphi}{\ldots} \to \overset{\varphi}{\bullet} \to \overset{\omega}{\ldots}$$

$$\mathbf{Y}\varphi \quad \cdots \to \overset{\varphi}{\bullet} \to \bullet \to \overset{\omega}{\ldots}$$

Safety fragments of LTL

Safety LTL (nnf): $\phi := \phi \vee \phi \mid \phi \wedge \phi \mid p \mid \underbrace{\neg p}_{\text{Neg on leaves}} \mid \mathbf{X}\phi \mid \underbrace{\phi \mathbf{R} \phi}_{\text{No until}} \mid \mathbf{Y}\phi \mid \phi\mathbf{S}\phi \mid \mathbf{Z}\phi \mid \phi\mathbf{T}\phi$

G α-past: $\phi := \mathbf{G}\phi_P \qquad \phi_P := p \mid \neg\phi_P \mid \phi_P \vee \phi_P \mid \mathbf{Y}\phi_P \mid \phi_P\mathbf{S}\phi_P$

Relation with safety:

$$Safety \cap LTL \equiv safetyLTL \equiv G\alpha\text{-past}$$[Chang, Manna Pnuelli 92]

Relative safety[Henzinger92]

Let P and A be two properties. P is safety relative to A iff

$$\forall \pi \in A \text{ s.t. } \pi \notin P, \exists \pi_f \in Pref(\pi) \text{ s.t.}$$
$$\forall \pi^\omega \in \Sigma^\omega : \text{ if } \pi_f \pi^\omega \in A \text{ then } \pi_f \pi^\omega \notin P$$

Relative safety[Henzinger92]

Let P and A be two properties. P is safety relative to A iff

$$\forall \pi \in A \text{ s.t. } \pi \notin P, \exists \pi_f \in Pref(\pi) \text{ s.t.}$$
$$\forall \pi^\omega \in \Sigma^\omega : \text{ if } \pi_f \pi^\omega \in A \text{ then } \pi_f \pi^\omega \notin P$$

Less formally: "Considering a world in which A is true, P becomes safety."

Relative safety[Henzinger92]

Let P and A be two properties. P is safety relative to A iff

$$\forall \pi \in A \text{ s.t. } \pi \notin P, \exists \pi_f \in \mathit{Pref}(\pi) \text{ s.t.}$$
$$\forall \pi^\omega \in \Sigma^\omega : \text{ if } \pi_f \pi^\omega \in A \text{ then } \pi_f \pi^\omega \notin P$$

Less formally: "Considering a world in which A is true, P becomes safety."

Notable examples:

- φ_S is safety relative to $\top$.
- $\mathbf{G}p \rightarrow \mathbf{G}q$ is safety relative to $\mathbf{G}p$.
- Bounded response is safety relative to non-zenoness and weak monotonicity.
- $p\mathbf{U}q$ is safety relative to $\mathbf{F}q$

Contributions

Steps

1. Construct a safetyLTL model checking procedure reducing to invariant checking
2. Extend the approach to the fragment: $\underbrace{\alpha_S}_{\text{Safety property}} \wedge \underbrace{\alpha_L}_{\text{Liveness property}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$ (complete with assumptions)
3. Generalize the approach to generic $\underbrace{\alpha}_{\text{LTL}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$ using an iterative approach to block unfair counterexamples

Note: Extension can be done because $\alpha \rightarrow \varphi$ safety relative to α.

Invariant-based SafetyLTL verification (high level overview)

Invariant-based SafetyLTL verification (high level overview)

- Combines automata construction of LTL[Clarke, Grumberg, Hamaguchi CAV 1994] with the notion of informative prefix[Kupferman, Vardi 2001].

Invariant-based SafetyLTL verification (high level overview)

- Combines automata construction of LTL[Clarke, Grumberg, Hamaguchi CAV 1994] with the notion of informative prefix[Kupferman, Vardi 2001].
- **informative prefix:** $\approx$ "an algorithmic definition for bad prefixes"

Invariant-based SafetyLTL verification (high level overview)

- Combines automata construction of LTL[Clarke, Grumberg, Hamaguchi CAV 1994] with the notion of informative prefix[Kupferman, Vardi 2001].
- **informative prefix:** $\approx$ "an algorithmic definition for bad prefixes"
- Notion used to construct invariant, the invariant is valid iff there is no informative prefix for the property.

Invariant-based SafetyLTL verification (high level overview)

- Combines automata construction of LTL[Clarke, Grumberg, Hamaguchi CAV 1994] with the notion of informative prefix[Kupferman, Vardi 2001].
- **informative prefix:** $\approx$ "an algorithmic definition for bad prefixes"
- Notion used to construct invariant, the invariant is valid iff there is no informative prefix for the property.

If $\mathcal{M}$ is deadlock free, then

$$\mathcal{M} \models \phi_S \Leftrightarrow \mathcal{M} \models_{INVAR} \phi_S$$

Basic algorithm (no loop)

Check: $\underbrace{\mathcal{M}}_{\langle V, \mathcal{I}, \mathcal{T}, \mathcal{F} \rangle \text{fair STS}} \models \underbrace{\alpha_S}_{\text{safety}} \wedge \underbrace{\alpha_L}_{\text{liveness}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$

$\boxed{\mathcal{M}' := \mathcal{M} \times \mathcal{M}_{\alpha_S}}$

Basic algorithm (no loop)

Check: $\underbrace{\mathcal{M}}_{\langle V, \mathcal{I}, \mathcal{T}, \mathcal{F} \rangle \text{fair STS}} \models \underbrace{\alpha_S}_{\text{safety}} \wedge \underbrace{\alpha_L}_{\text{liveness}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$

$\boxed{\mathcal{M}' := \mathcal{M} \times \mathcal{M}_{\alpha_S}}$

$\boxed{\mathcal{M}' \models_{INVAR} \varphi}$

Basic algorithm (no loop)

Check: $\underbrace{\mathcal{M}}_{\langle V, \mathcal{I}, \mathcal{T}, \mathcal{F}\rangle \text{fair STS}} \models \underbrace{\alpha_S}_{\text{safety}} \wedge \underbrace{\alpha_L}_{\text{liveness}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$

$\mathcal{M}' := \mathcal{M} \times \mathcal{M}_{\alpha_S}$

$\downarrow$

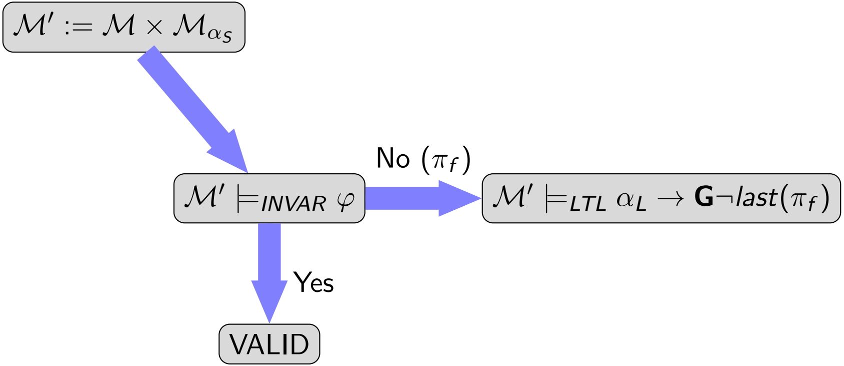
$\mathcal{M}' \models_{INVAR} \varphi$

$\downarrow$ Yes

VALID

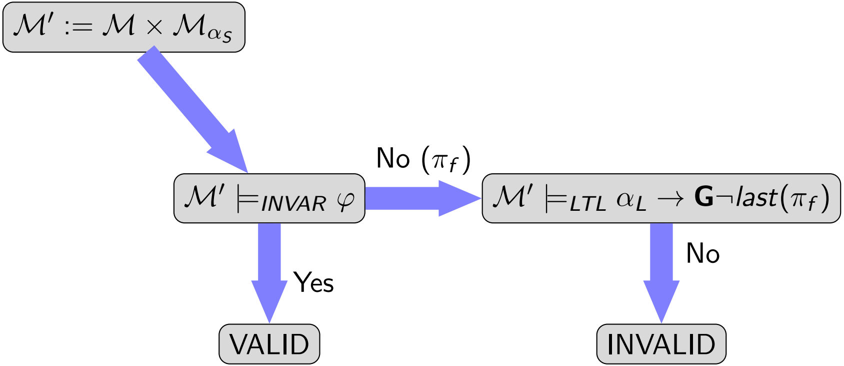
Basic algorithm (no loop)

Check: $\underbrace{\mathcal{M}}_{\langle V, \mathcal{I}, \mathcal{T}, \mathcal{F} \rangle \text{fair STS}} \models \underbrace{\alpha_S}_{\text{safety}} \wedge \underbrace{\alpha_L}_{\text{liveness}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$



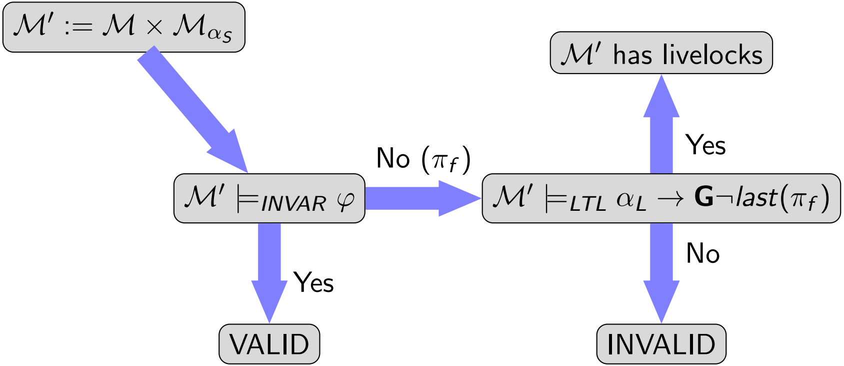
Basic algorithm (no loop)

Check: $\underbrace{\mathcal{M}}_{\langle V,\mathcal{I},\mathcal{T},\mathcal{F}\rangle \text{fair STS}} \models \underbrace{\alpha_S}_{\text{safety}} \wedge \underbrace{\alpha_L}_{\text{liveness}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$



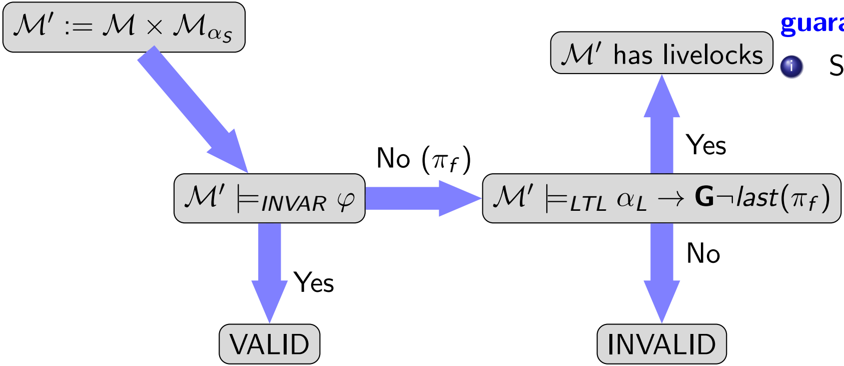
Basic algorithm (no loop)

Check: $\underbrace{\mathcal{M}}_{\langle V,\mathcal{I},\mathcal{T},\mathcal{F}\rangle\text{fair STS}} \models \underbrace{\alpha_S}_{\text{safety}} \wedge \underbrace{\alpha_L}_{\text{liveness}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$



Basic algorithm (no loop)

Check: $\underbrace{\mathcal{M}}_{\langle V, \mathcal{I}, \mathcal{T}, \mathcal{F} \rangle \text{fair STS}} \models \underbrace{\alpha_S}_{\text{safety}} \wedge \underbrace{\alpha_L}_{\text{liveness}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$

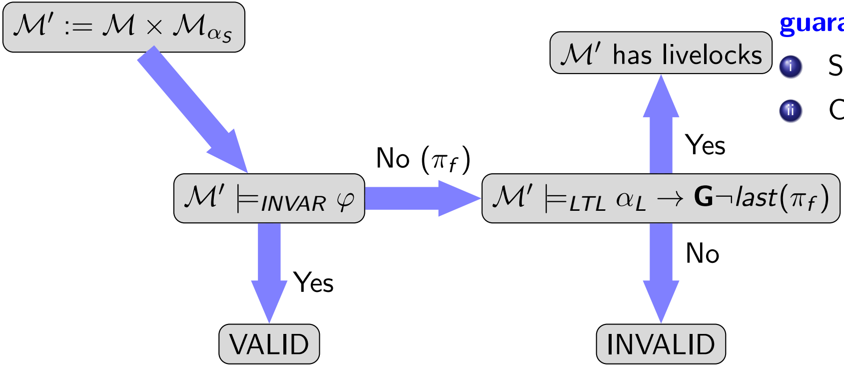


guarantees:

- Soundness Yes

Basic algorithm (no loop)

Check: $\underbrace{\mathcal{M}}_{\langle V,\mathcal{I},\mathcal{T},\mathcal{F}\rangle \text{fair STS}} \models \underbrace{\alpha_S}_{\text{safety}} \wedge \underbrace{\alpha_L}_{\text{liveness}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$

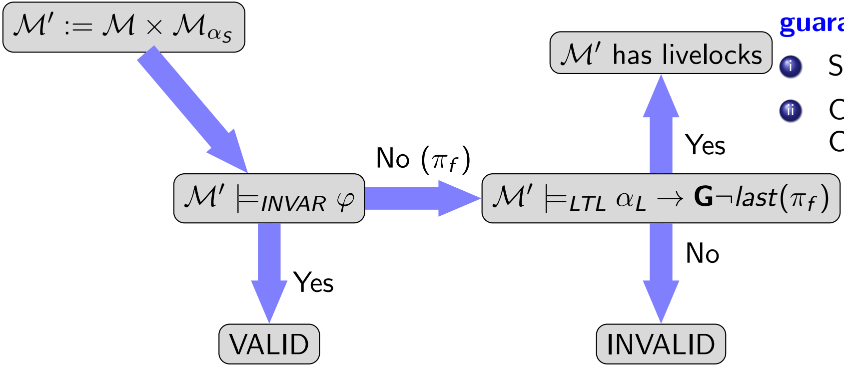


guarantees:

1. Soundness Yes
2. Completeness No

Basic algorithm (no loop)

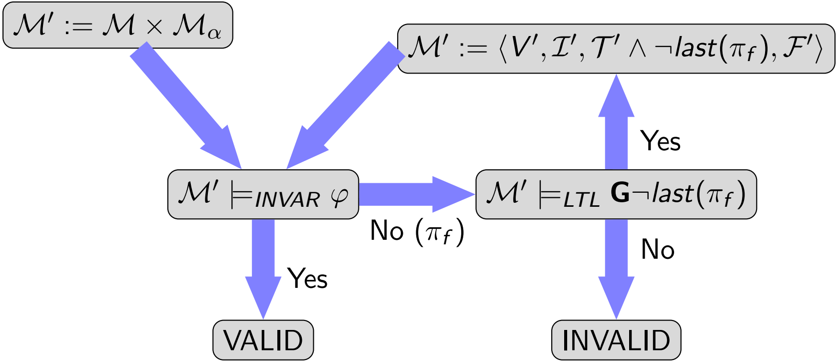
Check: $\underbrace{\mathcal{M}}_{\langle V,\mathcal{I},\mathcal{T},\mathcal{F}\rangle \text{fair STS}} \models \underbrace{\alpha_S}_{\text{safety}} \wedge \underbrace{\alpha_L}_{\text{liveness}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$



guarantees:

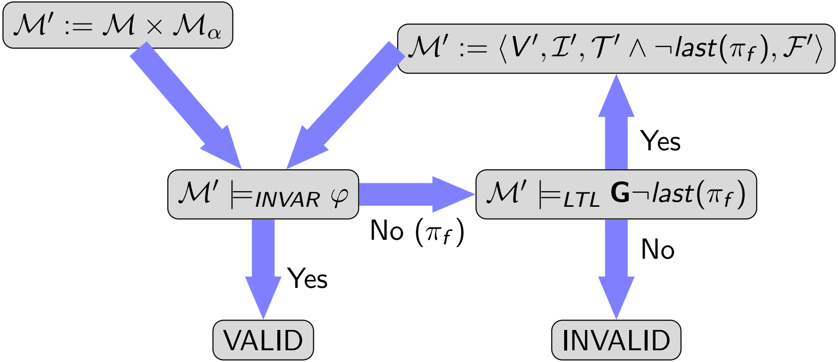
1. Soundness — Yes
2. Completeness — No
   Complete assuming:
   1. Invariant and LTL checks terminate,
   2. $\alpha_S := \mathbf{G}\alpha_P$ (construction does not introduce deadlocks),
   3. $\mathcal{M}$ is live w.r.t. $\alpha_S \wedge \alpha_L$

Check: $\underbrace{\mathcal{M}}_{\langle V,\mathcal{I},\mathcal{T},\mathcal{F}\rangle \text{fair STS}} \models \underbrace{\alpha}_{\text{LTL}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$



Iterative algorithm

Check: $\underbrace{\mathcal{M}}_{\langle V, \mathcal{I}, \mathcal{T}, \mathcal{F}\rangle \text{fair STS}} \models \underbrace{\alpha}_{\text{LTL}} \rightarrow \underbrace{\varphi}_{\text{safetyLTL}}$



Complete for finite state STS

Optimization: extending safetyLTL verification with lookahead

Idea:

- If $\mathcal{M}'$ has *livelocks*, multiple iterations are required
- Computing steps ahead for counterexamples can rule out deadlock states.

Discard:

$$\underbrace{\bullet \to \bullet \to \bullet}_{\text{Bad prefix } \pi_f} \to \underbrace{X}_{\text{Deadlock}}$$

Consider:

$$\underbrace{\bullet \to \bullet \to \bullet}_{\text{Bad prefix } \pi'_f} \to \overset{la=1}{\bullet} \to \underbrace{\dots}_{n-2 \text{ steps}} \to \overset{la=n}{\bullet}$$

Procedure:

- When the original invariant is falsified start incrementing *la*
- New INVAR: $la < n$

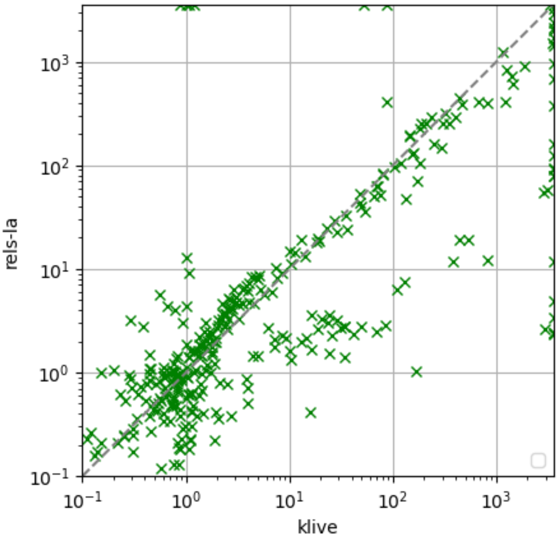
Experimental evaluation

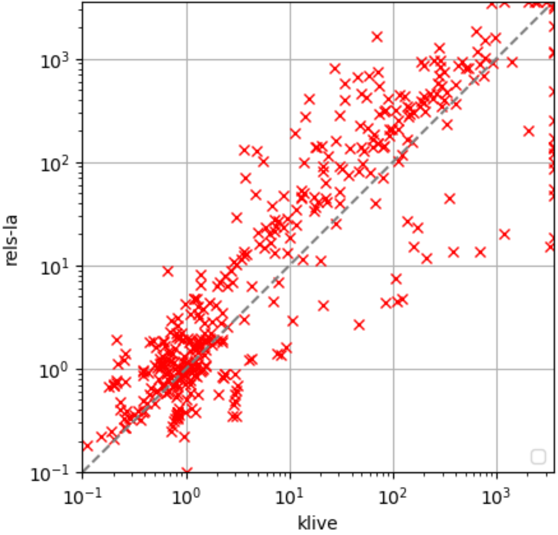
- Implemented inside nuXmv symbolic model checker on top of SMT based infinite state invariant checking.
- LTL check using K-liveness with IC3
- Invariant checking done with IC3
- Comparison with k-liveness[K. Claessen and N. Sörensson 2012], liveness to safety[A. Biere, C. Artho, and V. Schuppan 2002] (adapted for infinite state).

Models:

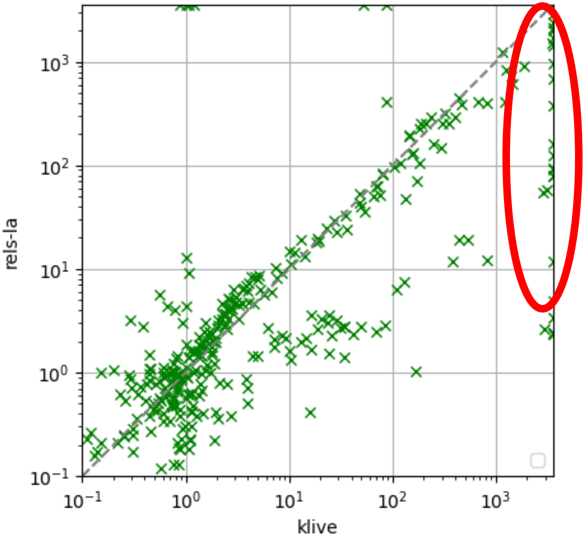
- A/G contracts (e.g. Wheel Brake System)
- Bounded response (infinite state)
- Asynchronous systems with fair scheduling $\bigwedge_i \mathbf{GF} run_i \rightarrow \varphi$
- NuSMV models (finite state)
- nuXmv models (infinite state)

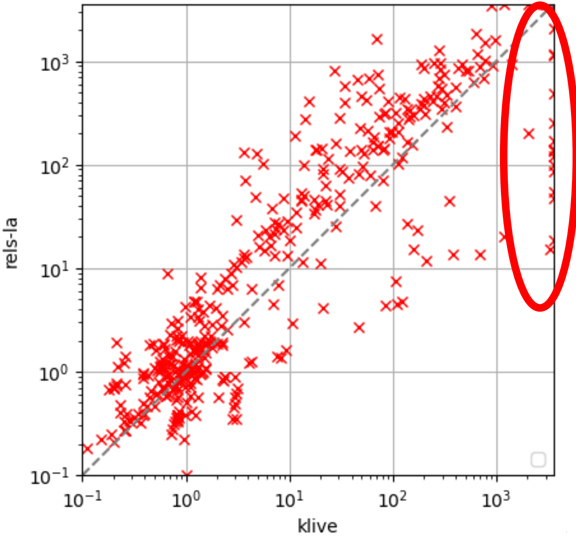
Comparison with k-liveness



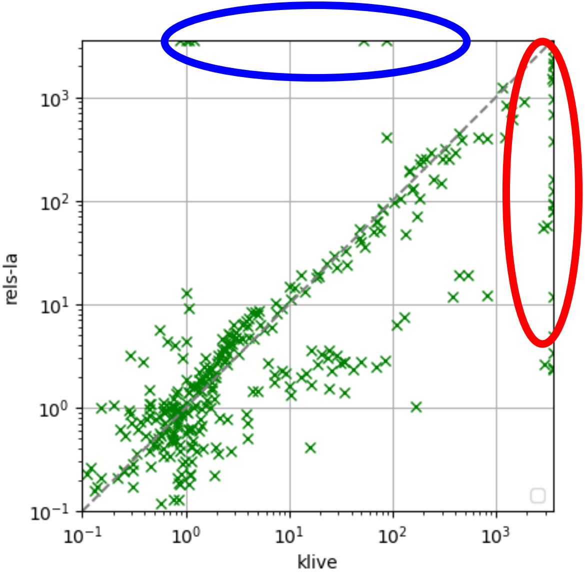


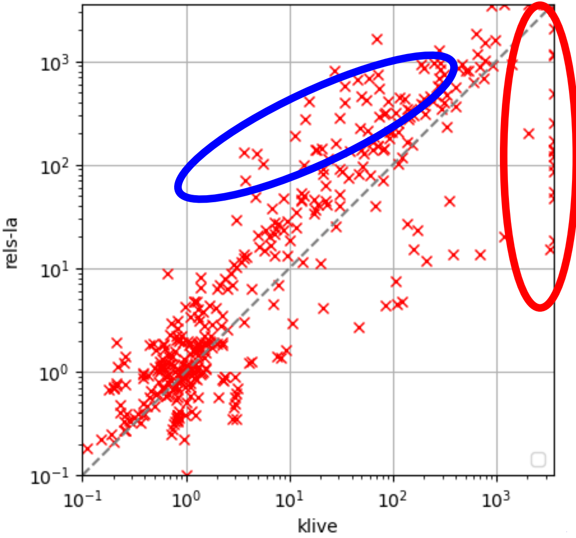
Comparison with k-liveness



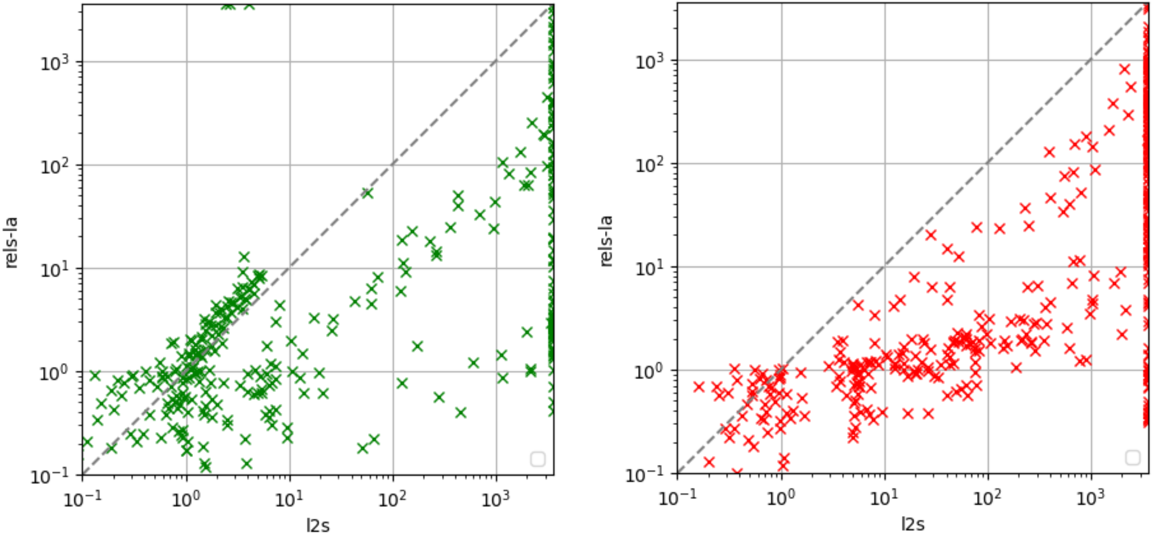


Comparison with k-liveness

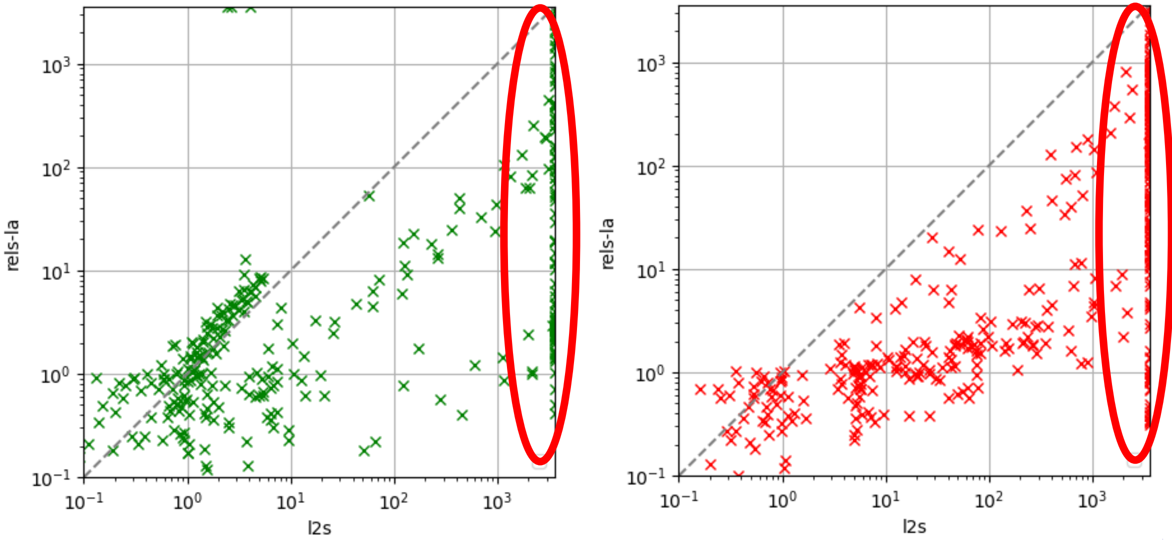




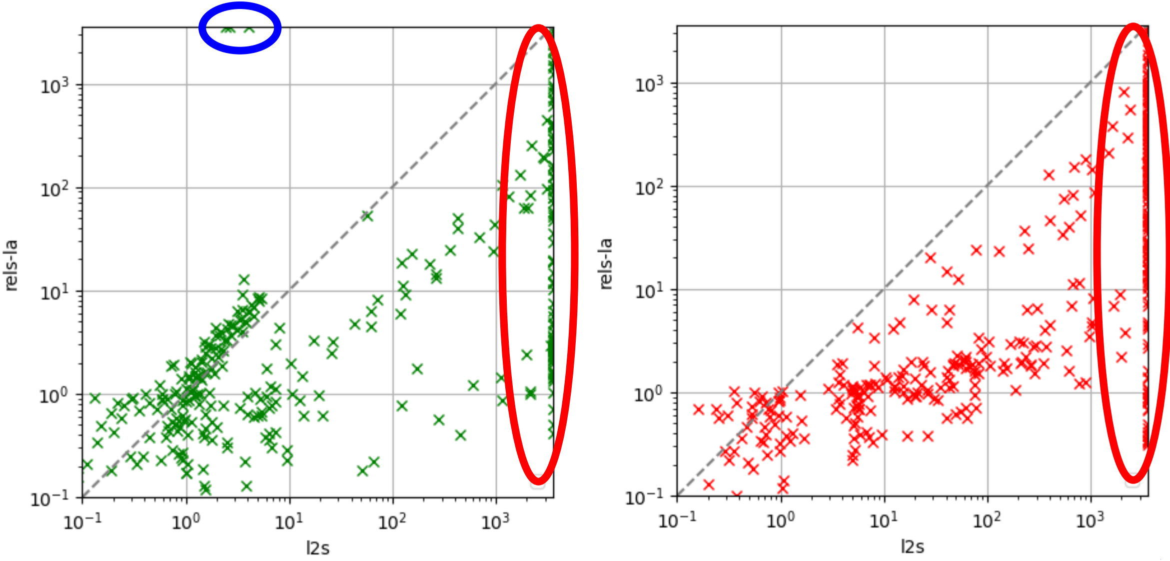
Comparison with liveness to safety



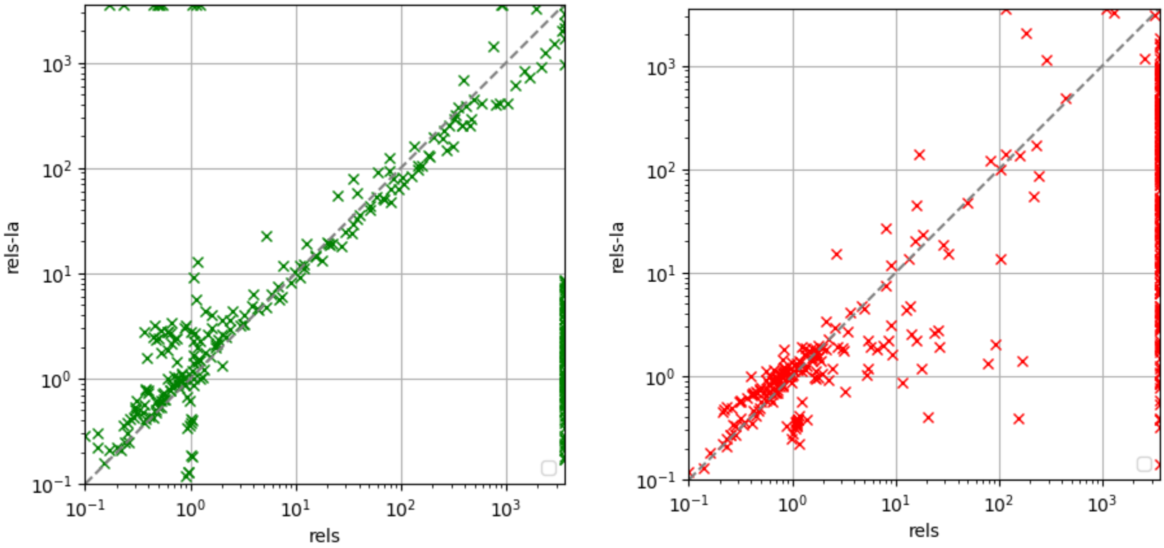
Comparison with liveness to safety



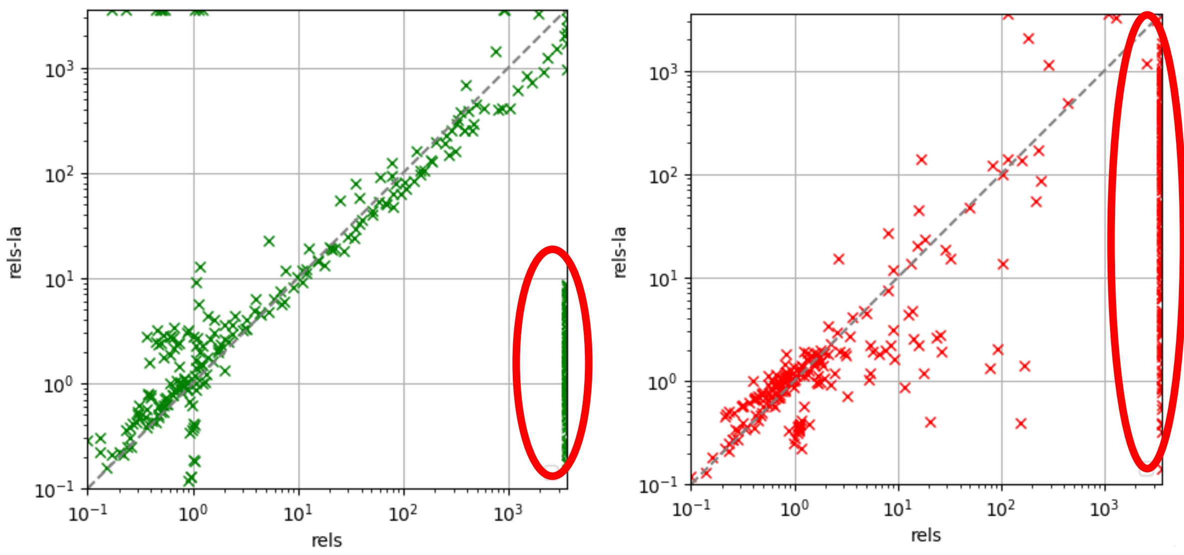
Comparison with liveness to safety



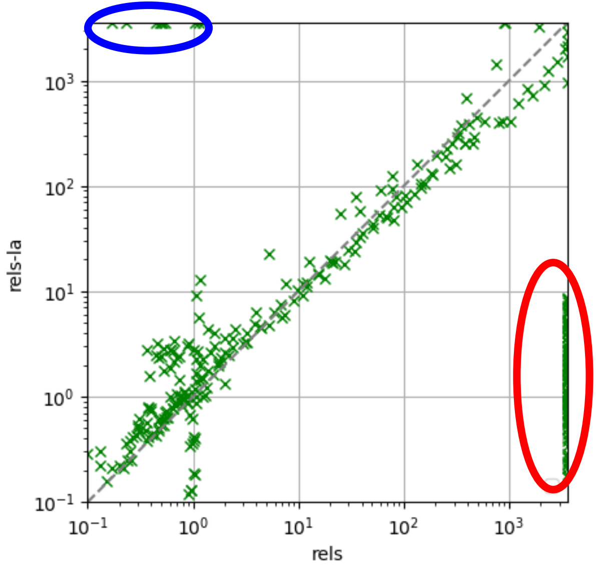
Impact of lookahead construction

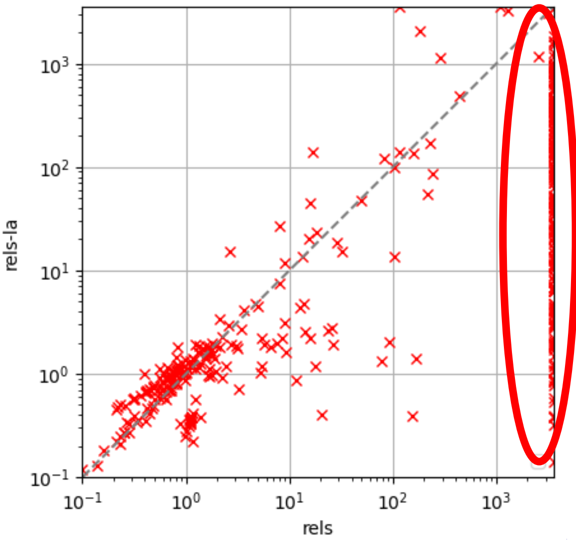


Impact of lookahead construction



Impact of lookahead construction





Conclusion

Concluding remarks

Considerations:

1. Deadlocks and livelocks are the main obstacle, many times the LTL to automata construction introduces the deadlocks with prophecy variables ($v_{\mathbf{X}\beta}$).
2. Providing a finite lookahead computation is sufficient to rule out many spurious counterexamples.
3. There are rooms for improvements (next slide)

Future directions

Improvements of the algorithm:

- Counterexample generalization exploiting k-liveness (using inductive invariants)
- Consider using temporal testers
- Extend with lockstep with BMC (as for k-liveness)
- Exploit SMT solver incrementality

Applications of the algorithm:

- Extend the fragment such that φ can be non-safety
- Targetting continuous time
- Apply to contract-based verification compositionally where A/G are formulae

Questions?

Appendix

LTL model checking

Standard LTL model checking:

$$\mathcal{M} \models_{LTL} \phi \Leftrightarrow \mathcal{M} \times \mathcal{M}_{\neg\phi} \models \neg \bigwedge_{f_i \in \mathcal{F}_{\neg\phi}} \mathbf{GF} f_i$$

LTL model checking

Standard LTL model checking:

$$\mathcal{M} \models_{LTL} \phi \Leftrightarrow \mathcal{M} \times \mathcal{M}_{\neg\phi} \models \neg \bigwedge_{f_i \in \mathcal{F}_{\neg\phi}} \mathbf{GF} f_i$$

Symbolic compilation of LTL[Clarke, Grumberg, Hamaguchi CAV 1994]:

$$\mathcal{M}_\phi := \langle V_\phi, \mathcal{I}_\phi, \mathcal{T}_\phi \rangle$$

LTL model checking

Standard LTL model checking:

$$\mathcal{M} \models_{LTL} \phi \Leftrightarrow \mathcal{M} \times \mathcal{M}_{\neg\phi} \models \neg \bigwedge_{f_i \in \mathcal{F}_{\neg\phi}} \mathbf{GF} f_i$$

Symbolic compilation of LTL[Clarke, Grumberg, Hamaguchi CAV 1994]:

$$\mathcal{M}_\phi := \langle V_\phi, \mathcal{I}_\phi, \mathcal{T}_\phi \rangle$$

$$V_\phi := V \cup \{v_{\mathbf{X}\beta} \mid \mathbf{X}\beta \in Sub(\phi)\} \cup \{v_{\mathbf{X}(\phi_1 \mathbf{U} \phi_2)} \mid \phi_1 \mathbf{U} \phi_2 \in Sub(\phi)\}$$

1. Introduce prophecy variables for temporal operators.

LTL model checking

Standard LTL model checking:

$$\mathcal{M} \models_{LTL} \phi \Leftrightarrow \mathcal{M} \times \mathcal{M}_{\neg\phi} \models \neg \bigwedge_{f_i \in \mathcal{F}_{\neg\phi}} \mathbf{GF} f_i$$

Symbolic compilation of LTL[Clarke, Grumberg, Hamaguchi CAV 1994]:

$$\begin{aligned}
\mathcal{M}_\phi &:= \langle V_\phi, \mathcal{I}_\phi, \mathcal{T}_\phi \rangle \\
V_\phi &:= V \cup \{ v_{\mathbf{X}\beta} \mid \mathbf{X}\beta \in Sub(\phi) \} \cup \\
&\quad \{ v_{\mathbf{X}(\phi_1 \mathbf{U} \phi_2)} \mid \phi_1 \mathbf{U} \phi_2 \in Sub(\phi) \} \\
\mathcal{I}_\phi &:= Enc(\phi)
\end{aligned}$$

1. Introduce prophecy variables for temporal operators.
2. Initially ϕ holds
3. *Enc* rewrites operators in terms of prophecy variables.

LTL model checking

Standard LTL model checking:

$$\mathcal{M} \models_{LTL} \phi \Leftrightarrow \mathcal{M} \times \mathcal{M}_{\neg\phi} \models \neg \bigwedge_{f_i \in \mathcal{F}_{\neg\phi}} \mathbf{GF} f_i$$

Symbolic compilation of LTL[Clarke, Grumberg, Hamaguchi CAV 1994]:

$$\mathcal{M}_\phi := \langle V_\phi, \mathcal{I}_\phi, \mathcal{T}_\phi \rangle$$

$$V_\phi := V \cup \{v_{\mathbf{X}\beta} \mid \mathbf{X}\beta \in Sub(\phi)\} \cup \{v_{\mathbf{X}(\phi_1 \mathbf{U} \phi_2)} \mid \phi_1 \mathbf{U} \phi_2 \in Sub(\phi)\}$$

$$\mathcal{I}_\phi := Enc(\phi) \quad \mathcal{T}_\phi := \bigwedge_{v_{\mathbf{X}\beta} \in V_\phi \setminus V} (v_{\mathbf{X}\beta} \leftrightarrow Enc(\beta)')$$

1. Introduce prophecy variables for temporal operators.
2. Initially ϕ holds
3. *Enc* rewrites operators in terms of prophecy variables.
4. Relate each β to its prophecy variable

LTL model checking

Standard LTL model checking:

$$\mathcal{M} \models_{LTL} \phi \Leftrightarrow \mathcal{M} \times \mathcal{M}_{\neg\phi} \models \neg \bigwedge_{f_i \in \mathcal{F}_{\neg\phi}} \mathbf{GF} f_i$$

Symbolic compilation of LTL[Clarke, Grumberg, Hamaguchi CAV 1994]:

$$\mathcal{M}_\phi := \langle V_\phi, \mathcal{I}_\phi, \mathcal{T}_\phi \rangle$$

$$V_\phi := V \cup \{v_{\mathbf{X}\beta} \mid \mathbf{X}\beta \in Sub(\phi)\} \cup \{v_{\mathbf{X}(\phi_1 \mathbf{U} \phi_2)} \mid \phi_1 \mathbf{U} \phi_2 \in Sub(\phi)\}$$

$$\mathcal{I}_\phi := Enc(\phi) \quad \mathcal{T}_\phi := \bigwedge_{v_{\mathbf{X}\beta} \in V_\phi \setminus V} (v_{\mathbf{X}\beta} \leftrightarrow Enc(\beta)')$$

$$\mathcal{F}_\phi := \{Enc(\phi_1 \mathbf{U} \phi_2) \to Enc(\phi_2) \mid \phi_1 \mathbf{U} \phi_2 \in Sub(\phi)\}$$

1. Introduce prophecy variables for temporal operators.
2. Initially ϕ holds
3. *Enc* rewrites operators in terms of prophecy variables.
4. Relate each β to its prophecy variable
5. Enforce fairness for until

Informative prefix[Kupferman, Vardi 2001]

Definition

Let ψ be an LTL formula in negative normal form, $Sub(\psi)$ be the set of sub-formulas of ψ and let π be a finite path of length n over the language of ψ. We say that π is *informative* for ψ iff there exists a mapping $L : \{0, \ldots, n\} \to 2^{Sub(\neg\psi)}$ such that:

1. $\neg\psi \in L(0)$.
2. $L(n) = \emptyset$.
3. For all $0 \leq i < n$, forall $\varphi \in L(i)$:
   - If φ is propositional, $\pi, i \models \varphi$.
   - If $\varphi = \varphi_1 \vee \varphi_2$, $\varphi_1 \in L(i)$ or $\varphi_2 \in L(i)$.
   - If $\varphi = \varphi_1 \wedge \varphi_2$, $\varphi_1 \in L(i)$ and $\varphi_2 \in L(i)$.
   - If $\varphi = \mathbf{X}\varphi_1$, $\varphi_1 \in L(i+1)$
   - If $\varphi = \varphi_1 \mathbf{U} \varphi_2$, $\varphi_2 \in L(i)$ or $[\varphi_1 \in L(i)$ and $\varphi_1 \mathbf{U} \varphi_2 \in L(i+1)]$.
   - If $\varphi = \varphi_1 \mathbf{R} \varphi_2$, $\varphi_2 \in L(i)$ and $[\varphi_1 \in L(i)$ or $\varphi_1 \mathbf{R} \varphi_2 \in L(i+1)]$.

Safety LTL model checking

High level idea:

- Rewrite $\neg\phi$ in *nnf*
- Construct STS of $\neg\phi$ (similar to LTL2SMV)
- Compute invariant $INV_\phi := \neg(\bigwedge_{v_{\mathbf{X}\beta}} \neg v_{\mathbf{X}\beta})$

Safety LTL model checking

High level idea:

- Rewrite $\neg\phi$ in *nnf*
- Construct STS of $\neg\phi$ (similar to LTL2SMV)
- Compute invariant $INV_\phi := \neg(\bigwedge_{v_{\mathbf{X}\beta}} \neg v_{\mathbf{X}\beta})$

$$\mathcal{M}_{\neg\phi} := \langle V_{\neg\phi}, \mathcal{I}_{\neg\phi}, \mathcal{T}_{\neg\phi} \rangle$$

$$I_{\neg\phi} = Enc(\neg\phi) \wedge \bigwedge_{v_{\mathbf{Y}\beta} \in V_{\neg\phi}} \neg v_{\mathbf{Y}\beta}$$

$$T_{\neg\phi} = \bigwedge_{v_{\mathbf{X}\beta} \in V_{\neg\phi}} v_{\mathbf{X}\beta} \rightarrow Enc(\beta)')\wedge$$

$$\bigwedge_{v_{\mathbf{Y}\beta} \in V_{\neg\phi}} v'_{\mathbf{Y}\beta} \rightarrow Enc(\beta)$$

Safety LTL model checking

$$\mathcal{M}_{\neg\phi} := \langle V_{\neg\phi}, \mathcal{I}_{\neg\phi}, \mathcal{T}_{\neg\phi} \rangle$$

High level idea:

- Rewrite $\neg\phi$ in *nnf*
- Construct STS of $\neg\phi$ (similar to LTL2SMV)
- Compute invariant $INV_\phi := \neg(\bigwedge_{v_{\mathbf{X}\beta}} \neg v_{\mathbf{X}\beta})$

$$I_{\neg\phi} = Enc(\neg\phi) \wedge \bigwedge_{v_{\mathbf{Y}\beta} \in V_{\neg\phi}} \neg v_{\mathbf{Y}\beta}$$

$$T_{\neg\phi} = \bigwedge_{v_{\mathbf{X}\beta} \in V_{\neg\phi}} v_{\mathbf{X}\beta} \rightarrow Enc(\beta)') \wedge$$

$$\bigwedge_{v_{\mathbf{Y}\beta} \in V_{\neg\phi}} v'_{\mathbf{Y}\beta} \rightarrow Enc(\beta)$$

$Enc(\varphi)$:

- $Enc(\phi_1 \wedge \phi_2) = Enc(\phi_1) \wedge Enc(\phi_2)$, $Enc(\phi_1 \vee \phi_2) = Enc(\phi_1) \vee Enc(\phi_2)$
- $Enc(\neg\phi_1) = \neg Enc(\phi_1)$
- $Enc(\mathbf{X}\phi_1) = v_{\mathbf{X}\phi_1}$
- $Enc(\phi_1 \mathbf{U} \phi_2) = Enc(\phi_2) \vee (Enc(\phi_1) \wedge v_{\mathbf{X}(\phi_1 \mathbf{U} \phi_2)})$

Extended motivating example

Bounded response: $\varphi := \mathbf{G}(in \wedge t = p \rightarrow \mathbf{F}(t \leq p + 5 \wedge out))$

$$\alpha := \underbrace{\mathbf{G}(t' \geq t)}_{\text{Weak monotonicity}} \wedge \underbrace{\mathbf{GF}(t' - t \geq \epsilon)}_{\text{non-zenoness}}$$

Assuming α, bounded response can be reduced to $(\alpha \rightarrow (\varphi \leftrightarrow \varphi_S))$

$$\varphi_S := \mathbf{G}(in \wedge t = p \rightarrow out \mathbf{R} t \leq p + 5)$$

Counterexample shape of φ_S: $\underbrace{\overset{in \wedge t = p}{\bullet} \rightarrow \overset{t = p + 2}{\bullet} \rightarrow \overset{t = p + 5.01}{\bullet}}_{\text{Bad prefix}} \rightarrow \ldots$

Any finite counterexample of φ_S that can be extended to infinity is a counterexample of $\alpha \rightarrow \varphi$