A science of CPS robustness

Paulo Tabuada
Cyber-Physical Systems Laboratory
UCLA Dept. of Electrical and Computer Engineering

UCLA

Motivation: the problem with current specification mechanisms

- Cyber-Physical Systems (CPSs) are typically non-robust
  - a small deviation from the design assumptions can lead to a large deviation in the desired behavior;
- Specifications for cyber components are typically written as:

$$\varphi \Rightarrow \psi \tag{1}$$

where φ is an assumption about the environment and ψ is a system guarantee.
- The usual semantics of propositional logic dictates that (1) is equivalent to:

$$\neg\varphi \vee \psi. \tag{2}$$

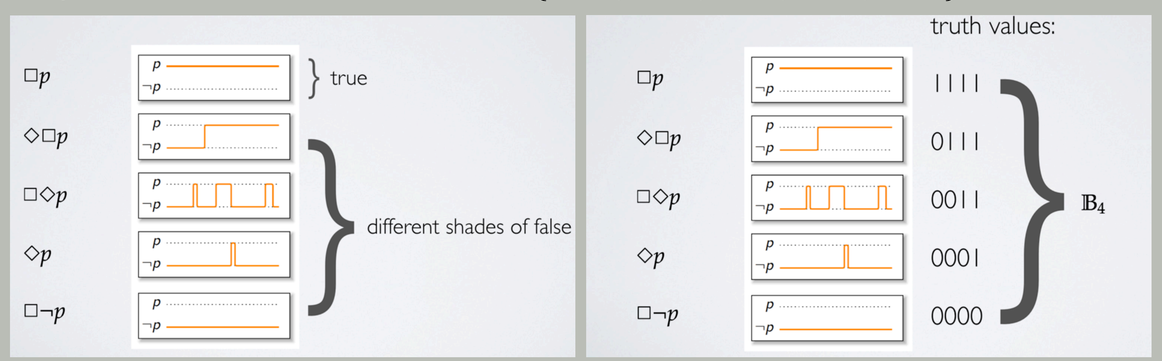
- If the assumption is violated, nothing can be said about the guarantee.
- Can we change the semantics of (1) so that "small" violations of φ lead to "small" violations of ψ?

Genesis of Robust Linear-time Temporal Logic (rLTL)

- rLTL adopts a 5-valued semantics to capture robustness: the truth value of an rLTL formula is interpreted as corresponding to *true* or to different shades of *false*. Consider the LTL formula:

$$\Box\, p \tag{3}$$

being true if the atomic proposition p holds at every time step, and false otherwise. Among all the different ways in which $\Box\, p$ can be violated, there is the following preference order:
  - p only fails to hold at finitely many time instants (i.e., that $\Diamond\Box\, p$ holds);
  - p holds at infinitely many time instants (i.e., that $\Box\Diamond\, p$ holds);
  - p holds at finitely many time instants (i.e., that $\Diamond\, p$ holds);
  - p fails to hold at every time instant (i.e., that $\Box\, \neg p$ holds).
- This suggests a new semantics for LTL, for which the robust version of the always operator, $\boxdot$, is *five valued* in $\mathbb{B}_5 = \{0000, 0001, 0011, 0111, 1111\}$.



Robust Linear-time Temporal Logic (rLTL)

- rLTL has the same syntax as LTL. Formulas are built from:
  - atomic propositions: $p, q, r, \ldots \in \mathcal{P}$;
  - Boolean connectives: $\wedge$, $\vee$, $\neg$, and $\Rrightarrow$;
  - temporal operators: $\odot$, $\boxdot$, and $\diamondsuit$.
- We define the mapping $\mathrm{ltl} : \{1, \ldots, 4\} \times \mathrm{rLTL}(\mathcal{P}) \to \mathrm{LTL}(\mathcal{P})$ from each bit of an rLTL formula and the set of all rLTL formulas on $\mathcal{P}$, $\mathrm{rLTL}(\mathcal{P})$, to the set of all LTL formulas on $\mathcal{P}$, $\mathrm{LTL}(\mathcal{P})$ as:

Operator	Symbol	Semantics, for $p \in \mathcal{P}$, $\varphi, \psi \in \mathrm{rLTL}(\mathcal{P})$.
Atomic Proposition		$\forall 1 \leq i \leq 4 : \mathrm{ltl}(i, p) = p.$
Negation	$\neg$	$\forall 1 \leq i \leq 4 : \mathrm{ltl}(i, \neg\varphi) = \neg\mathrm{ltl}(1, \varphi).$
Disjunction	$\vee$	$\forall 1 \leq i \leq 4 : \mathrm{ltl}(i, \varphi \vee \psi) = \mathrm{ltl}(i, \varphi) \vee \mathrm{ltl}(i, \psi).$
Conjunction	$\wedge$	$\forall 1 \leq i \leq 4 : \mathrm{ltl}(i, \varphi \wedge \psi) = \mathrm{ltl}(i, \varphi) \wedge \mathrm{ltl}(i, \psi).$
Robust Implication	$\Rrightarrow$	$\forall 1 \leq i \leq 3 : \mathrm{ltl}(i, \varphi \Rrightarrow \psi) = (\mathrm{ltl}(i, \varphi) \Rightarrow \mathrm{ltl}(i, \psi)) \wedge \mathrm{ltl}(i+1, \varphi \Rrightarrow \psi)$, $\mathrm{ltl}(4, \varphi \Rrightarrow \psi) = (\mathrm{ltl}(4, \varphi) \Rightarrow \mathrm{ltl}(4, \psi)).$
Next	$\odot$	$\forall 1 \leq i \leq 4 : \mathrm{ltl}(i, \odot\, \varphi) = \bigcirc\, \mathrm{ltl}(i, \varphi).$
Robust Always	$\boxdot$	$\mathrm{ltl}(1, \boxdot\, \varphi) = \Box\, \mathrm{ltl}(1, \varphi)$, $\mathrm{ltl}(2, \boxdot\, \varphi) = \Diamond\Box\, \mathrm{ltl}(2, \varphi)$, $\mathrm{ltl}(3, \boxdot\, \varphi) = \Box\Diamond\, \mathrm{ltl}(3, \varphi)$, $\mathrm{ltl}(4, \boxdot\, \varphi) = \Diamond\, \mathrm{ltl}(4, \varphi).$
Robust Eventually	$\diamondsuit$	$\forall 1 \leq i \leq 4 : \mathrm{ltl}(i, \diamondsuit\, \varphi) = \Diamond\, \mathrm{ltl}(i, \varphi).$

- The rLTL semantics is defined as a function $V : (2^{\mathcal{P}})^\omega \times \mathrm{rLTL}(\mathcal{P}) \to \mathbb{B}_5$, where for any $\sigma \in (2^{\mathcal{P}})^\omega$, $\varphi \in \mathrm{rLTL}(\mathcal{P})$ and $1 \leq i \leq 4$, the ith bit $V_i(\sigma, \varphi)$ of the valuation $V(\sigma, \varphi)$ is given by

$$V_i(\sigma, \varphi) = W(\sigma, \mathrm{ltl}(i, \varphi)), \tag{4}$$

where $W(\sigma, \psi)$ is the truth value of the LTL formula ψ evaluated on σ.
- Tractability:
  Theorem: The verification and synthesis problems for an rLTL formula φ are decidable with the following time complexity:

	rLTL	LTL
Verification	$5^{\lvert\varphi\rvert}$	$2^{\lvert\varphi\rvert}$
Synthesis	$2^{5^{\lvert\varphi\rvert}}$	$2^{2^{\lvert\varphi\rvert}}$

.

rLTL Verification: now faster than ever before!

- rLTL Verification can be made faster by using *Temporal Testers* with the fragment $\widetilde{\mathrm{rLTL}}(\mathcal{P})$.
  - *Temporal Testers* are discrete transition systems equipped with justice conditions that detect if a computation satisfies an LTL formula.
  - Consider the rLTL fragment $\{\psi,\ \psi_1 \Rrightarrow \psi_2 \mid \psi, \psi_1, \psi_2 \in \widetilde{\mathrm{rLTL}}(\mathcal{P})\}$, where $\widetilde{\mathrm{rLTL}}(\mathcal{P})$ is the set of all rLTL formulas that do not contain robust implications or robust releases.
  - For an rLTL formula in the above fragment, and for each bit i, we construct optimized Temporal Testers of size

$$2|\varphi| \leq |T(\varphi_i)| \leq 3|\varphi|, \tag{5}$$

testing if the a given infinite word satisfies the LTL formula $\mathrm{ltl}(i, \varphi)$.
- **Theorem:** For any rLTL formula $\varphi \in \{\psi,\ \psi_1 \Rrightarrow \psi_2 \mid \psi, \psi_1, \psi_2 \in \widetilde{\mathrm{rLTL}}(\mathcal{P})\}$, the rLTL verification problem is solved by performing at most 4 LTL verification steps, each using an automaton of size at most

$$\mathcal{O}\left(2^{|\varphi| - k(\varphi)} 3^{k(\varphi)}\right), \tag{6}$$

where $k(\varphi)$ is the number of $\boxdot$ operators in the rLTL formula φ.

rLTL Verification in practice - Evrostos: The rLTL Verifier

- The tool Evrostos solves the model checking problem for the aforementioned rLTL fragment. It consists of two components:
  - an rLTL-to-LTL translator;
  - the popular symbolic model checker NuSMV.
- The time required to solve the rLTL verification problem, t_{rLTL}, is larger than the corresponding time for the LTL verification problem, t_{LTL}. We write $t_{LTL} = 2^{|\varphi|}$, $t_{rLTL} = 2^{\zeta|\varphi|}$, and ask what is the exponent ζ (time complexity blowup) that describes the overhead:

$$\zeta = 1 + \frac{log\left(\frac{t_{rLTL}}{t_{LTL}}\right)}{|\varphi|}. \tag{7}$$

Time complexity for rLTL, being proportional to $3^{|\varphi|}$, implies an upper bound for ζ of $log_2(3) = 1.58$. As shown below for the telephone system model [4], the time complexity of rLTL for the fragment we are considering is close to that of LTL.

Formula	rLTL Truth Value	rLTL Time (s)	LTL Truth Value	LTL Time (s)	Time Complexity Blowup
$\boxdot\,(\neg(tt1 \wedge d12) \vee td2)$	0001	319.10	FALSE	265.29	1.04
$\boxdot\,(\neg(msg2) \vee ((d21 \wedge tcs12) \vee (d24 \wedge tcs42)))$	0011	26.71	FALSE	11.54	1.11
$\boxdot\,(\neg(tcs12 \wedge ring1) \vee ringt3)$	0001	157.95	FALSE	117.01	1.06
$\boxdot\,(\neg(msg3) \vee ((d31 \wedge tcs13) \vee (d34 \wedge tcs43)))$	0011	139.25	FALSE	55.91	1.12
$\boxdot\,(\neg(try1) \vee \odot\,(ringt1 \vee busyt1))$	0011	541.17	FALSE	220.24	1.16
$\boxdot\,(\neg(tt1 \wedge d13) \vee td3)$	1111	10.43	TRUE	3.01	1.26
$\boxdot\,(\neg(tcs13) \vee \boxdot\,(\neg(d31 \wedge (ringt3 \vee tt3))))$	0001	94.46	FALSE	91.92	1.00
$\boxdot\,(\neg(tcs42) \vee \boxdot\,(\neg(d24 \wedge (ringt2 \vee tt2))))$	0001	11.10	FALSE	7.53	1.05

Future work

- Relationships with existing notions of robustness in control theory;
- rLTL synthesis problem.

References

1. Evrostos: The rLTL Verifier
   Tzanis Anevlavis, Daniel Neider, Matthew Philippe and Paulo Tabuada
   Submitted to the 22nd ACM International Conference on Hybrid Systems: Computation and Control (HSCC 2019).
2. Verifying rLTL formulas: now faster than ever before!
   Tzanis Anevlavis, Matthew Philippe, Daniel Neider and Paulo Tabuada
   To appear in the 57th IEEE Conference on Decision and Control (CDC 2018).
3. Robust Linear Temporal Logic
   Paulo Tabuada and Daniel Neider
   25th EACSL Annual Conference on Computer Science Logic (CSL 2016).
4. Feature integration using a feature construct
   Malte Plath and Mark Ryan
   Science of Computer Programming 41, 1 (2001), 53 - 84.

NSF Project Award Number: 1645824, A science of CPS robustness