Complementation of Phase Event Automata

Lena Funk, Vincent Langenfeld, Nico Hauff, Andreas Podelski

11.09.2023

University of Freiburg, Chair of Software Engineering

Requirements

Requirement R_1

"The airbag must deploy within 50.0 milliseconds of detecting a collision."

- Requirements are often written in natural language, making them prone to errors.
- With automatic requirements analysis, we can check a set of requirements for generic properties to uncover defects.

Scalable Analysis of Real Time Requirements

- Each requirement is formalised and translated into a Phase Event Automaton (PEA).
- A program that encodes the simultaneous execution of all the PEAs is constructed.
- Generic properties are encoded in the program as error locations.
- A reachability check for the error locations is performed.

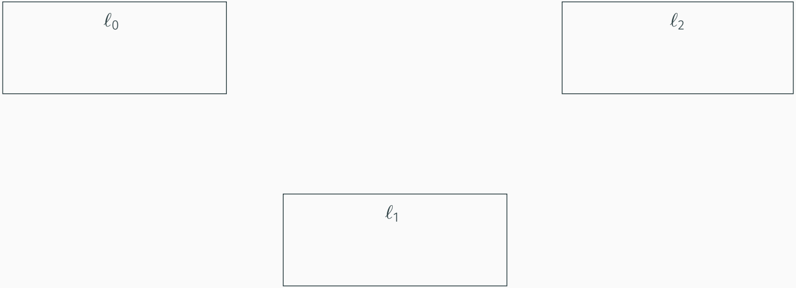
Application

- Problem: Given two requirements R_0 and R_1, is R_1 redundant?
- In other words: Does R_0 already cover the system behavior characterised by R_1 such that R_1 can be discarded?
- $\mathfrak{L}(\mathcal{A}_{R_0}) \cap \mathfrak{L}(\mathcal{A}_{R_1})^C \stackrel{?}{=} \emptyset$
- If the above intersection results in the empty set, we can discard R_1.

Requirement R_1

"The airbag must deploy within 50.0 milliseconds of detecting a collision."

$\mathcal{A}_{R_1}$



$\mathcal{A}_{R_1}$

ℓ_0
$\neg collision \vee airbag$

ℓ_2
$\neg collision \wedge \neg airbag$

ℓ_1
$collision \wedge \neg airbag$

State invariant $s(\ell_i)$ over the state variables

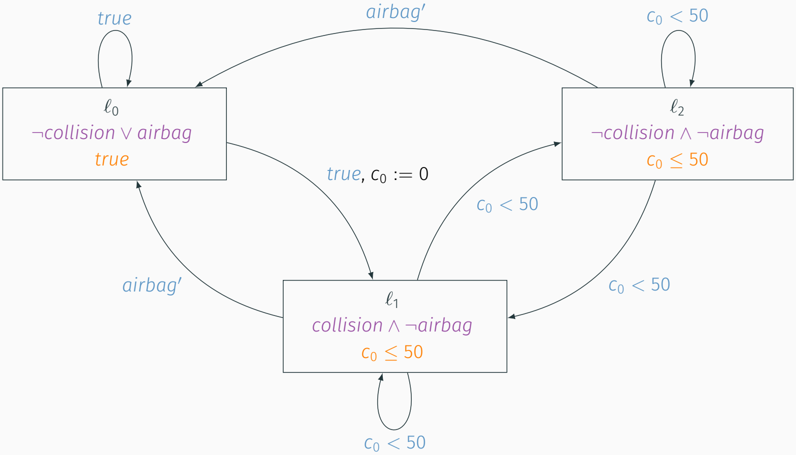
$\mathcal{A}_{R_1}$

ℓ_0
$\neg collision \vee airbag$
true

ℓ_2
$\neg collision \wedge \neg airbag$
$c_0 \leq 50$

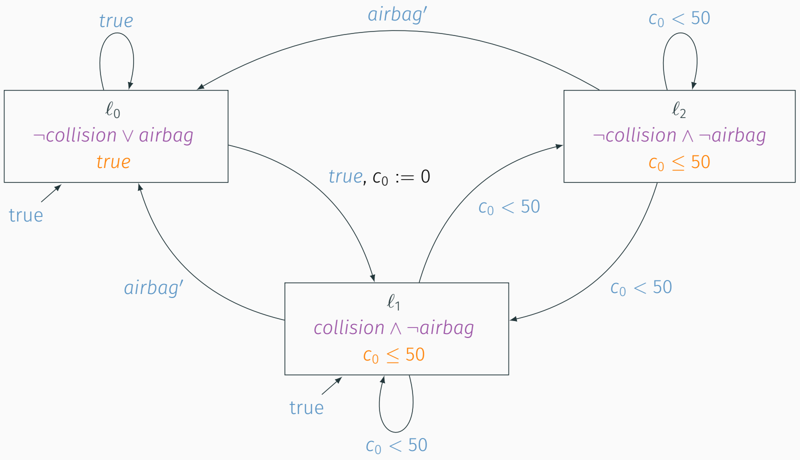
ℓ_1
$collision \wedge \neg airbag$
$c_0 \leq 50$

Clock invariant $I(\ell_i)$ over the clock variable c_0

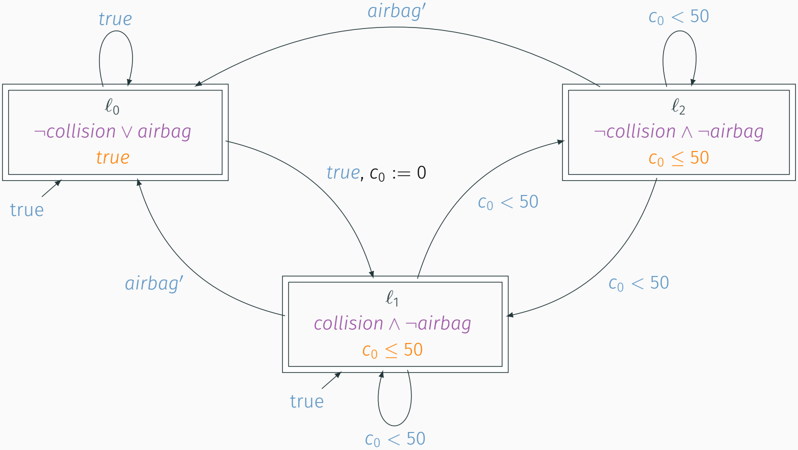


Transitions (ℓ, g, X, ℓ')

$\mathcal{A}_{R_1}$



Initial transitions (g, ℓ)



Terminal locations

Location Change, Configurations and Runs

- *Location Change*: Once a location's state or clock invariants are no longer satisfied, a transition to another location has to be taken. If none are enabled, the PEA is stuck.

Location Change, Configurations and Runs

- *Location Change*: Once a location's state or clock invariants are no longer satisfied, a transition to another location has to be taken. If none are enabled, the PEA is stuck.
- *Configuration*: Tuple that represents the state of the PEA

$$(\ell, \beta, \gamma, t)$$

Location Change, Configurations and Runs

- *Location Change*: Once a location's state or clock invariants are no longer satisfied, a transition to another location has to be taken. If none are enabled, the PEA is stuck.
- *Configuration*: Tuple that represents the state of the PEA

$$(\ell, \beta, \gamma, t)$$

- *Run*: Feasible sequence of configurations

Example

Run r:

$$r = \langle(\ell_0, \{collision = false, airbag = false\}, \{c_0 = 0\}, t = 15), \\ (\ell_1, \{collision = true, airbag = false\}, \{c_0 = 0\}, t = 30), \\ (\ell_0, \{collision = false, airbag = true\}, \{c_0 = 30\}, t = 15)\rangle$$

Corresponding word w:

$$w = \langle(\{collision = false, airbag = false\}, t = 15), \\ (\{collision = true, airbag = false\}, t = 30), \\ (\{collision = false, airbag = true\}, t = 15)\rangle \in \mathfrak{L}(\mathcal{A}_{R_1})$$

PEA Semantics - Discarded Words

Example

Non feasible sequence of configurations r^*:

$$r^* = \langle(\ell_0, \{collision = false, airbag = false\}, \{c_0 = 0\}, t = 15),$$
$$(\ell_1, \{collision = true, airbag = false\}, \{c_0 = 0\}, t = \mathbf{500})\rangle$$

Corresponding word w^*:

$$w^* = \langle(\{collision = false, airbag = false\}, t = 15),$$
$$(\{collision = true, airbag = false\}, t = 500)\rangle \in \mathfrak{L}(\mathcal{A}_{R_1})^C$$

Complementation Algorithm

Given: Any deterministic PEA $\mathcal{A}$ that accepts the language $\mathfrak{L}(\mathcal{A})$.

1. Make PEA $\mathcal{A}$ total and obtain PEA $\mathcal{A}_{total}$ with $\mathfrak{L}(\mathcal{A}) = \mathfrak{L}(\mathcal{A}_{total})$ (Totalisation).
2. Swap the terminal locations of $\mathcal{A}_{total}$ with its non-terminal locations and obtain $\mathcal{A}_{comp}$.

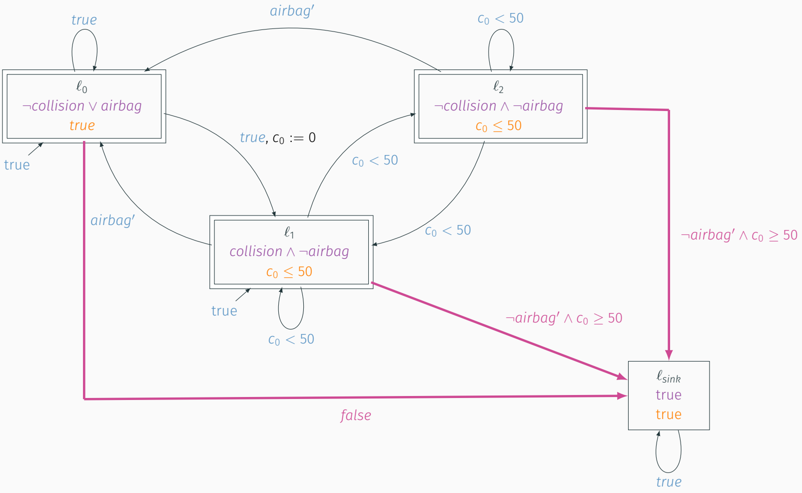
The resulting PEA $\mathcal{A}_{comp}$ *should* accept the complement language of $\mathcal{A}$, $\mathfrak{L}(\mathcal{A}_{comp}) = \mathfrak{L}(\mathcal{A})^C$.

Totalisation

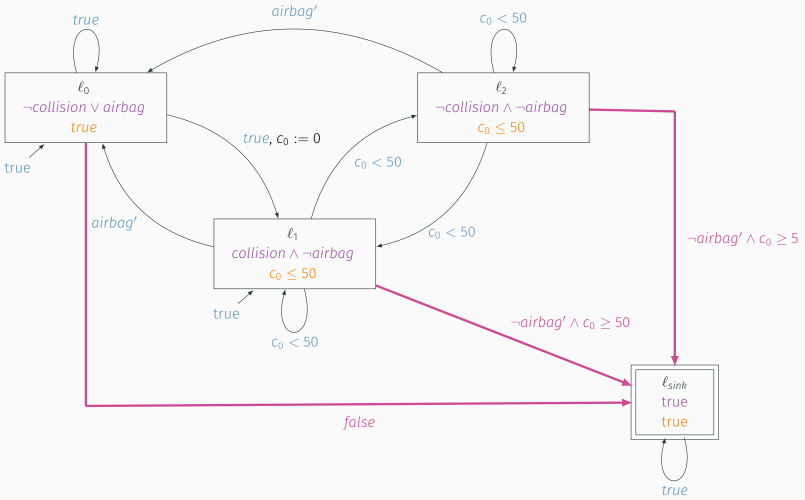
- Capture the sequences of configurations that are *not* runs in a *sink location* ℓ_{sink} that is not terminal.
- Each location has a *sink transition* $(\ell, g_{sink}, \emptyset, \ell_{sink})$ to the *sink location* ℓ_{sink} that is only enabled when no other outgoing transition is.

Result: PEA $\mathcal{A}_{total}$, that is total *and* deterministic: at any point in time and for any valuation of the state variables and clocks, there is *exactly one* transition enabled.

$\mathcal{A}_{R_1,total}$



$\mathcal{A}_{R_1,comp}$



Correctness

For non-strict PEAs (clock invariants contain only non-strict clock constraints):

- Correct.

■

For strict PEAs (clock invariants can contain strict clock constraints):

- Theoretically not correct... but still useful in practice!

Conclusion

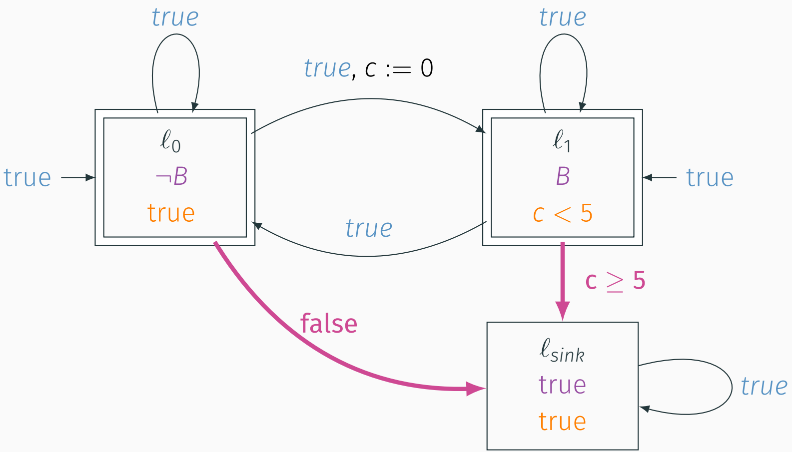
Our approach to complement PEAs...

... is proved to be correct for non-strict PEAs.

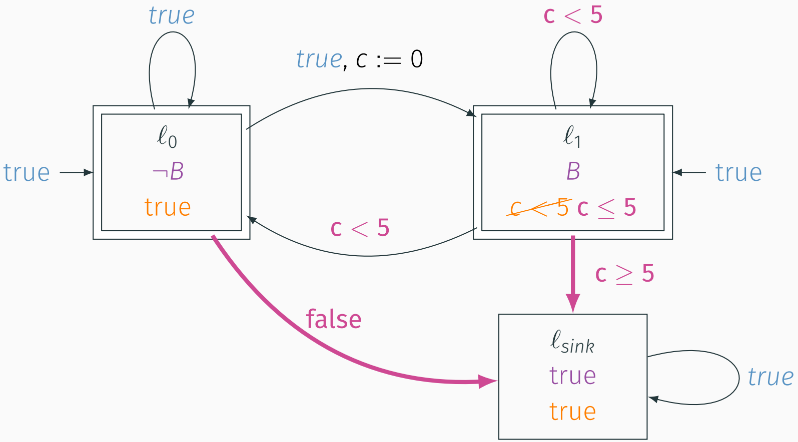
... can help us find redundancies in a set of requirements and thus helps to keep a set of requirements clear, concise and unambiguous.

Bonus Slides!

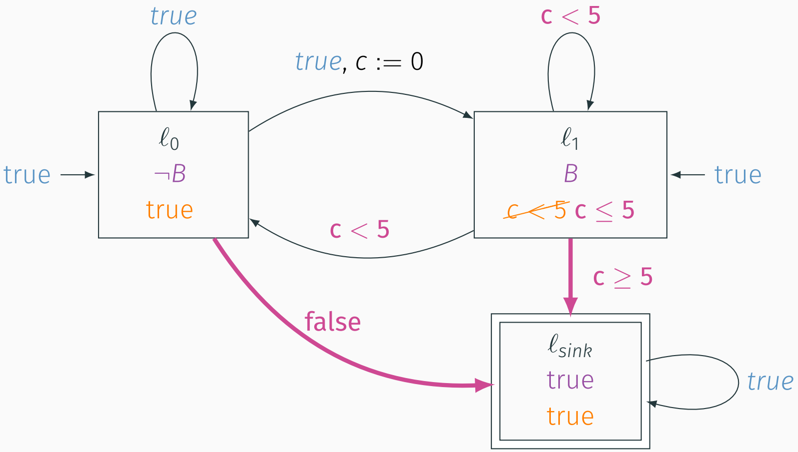
Locations with Strict Clock Constraints



$\mathcal{B}_{total}$



$\mathcal{B}_{comp}$



Why is this not correct?

- The set of words

$$W = \{\langle(\beta_0, t_0), ...(\{B = true\}, 5), ..., (\beta_n, t_n)\rangle \mid t_0, ..., t_n \in \mathbb{R}\}$$

  is in $\mathfrak{L}(\mathcal{B}_{total})$, but not in $\mathfrak{L}(\mathcal{B})$.
- For PEAS that have locations which include *strict* clock constraints in their invariants, it holds that

$$\mathfrak{L}(\mathcal{B}) \neq \mathfrak{L}(\mathcal{B}_{total})$$

  and

$$\mathfrak{L}(\mathcal{B})^C \neq \mathfrak{L}(\mathcal{B}_{comp}).$$