Trace-Diagnostic for Signal Temporal Properties: an Evolutionary Approach

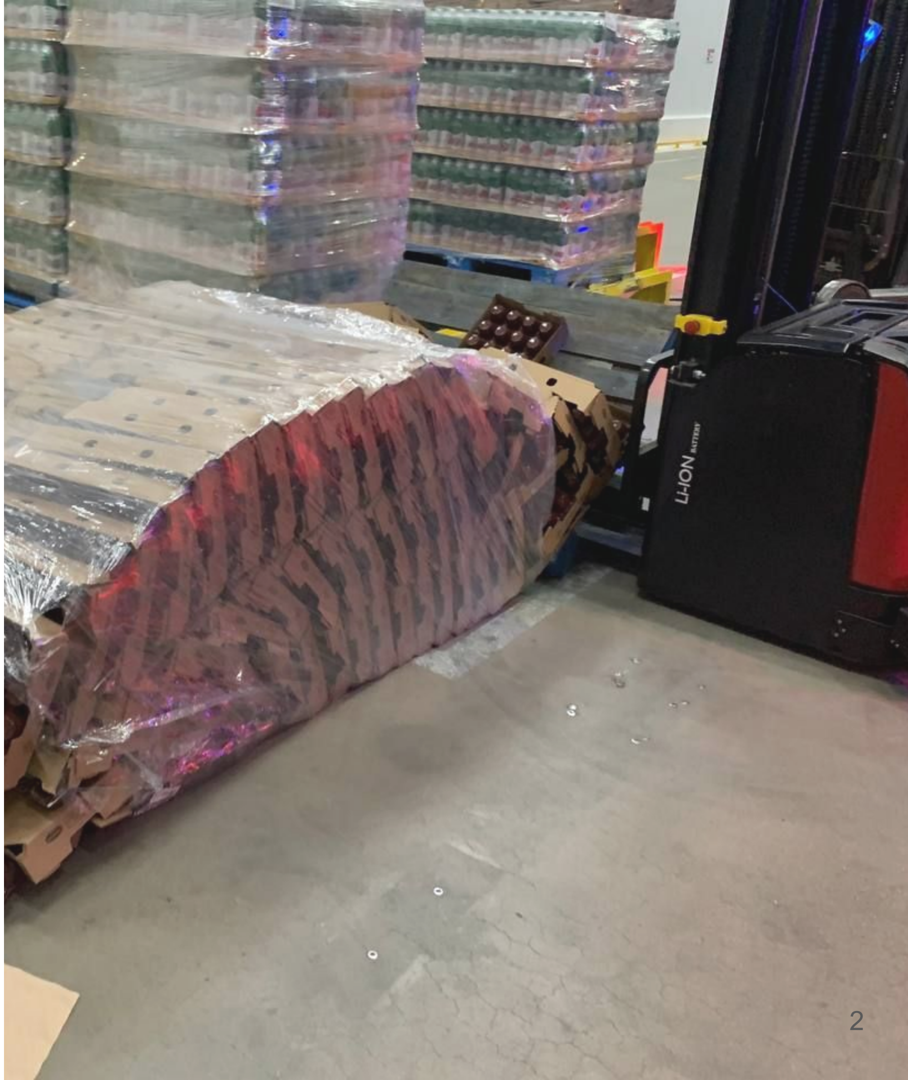
Author: Gabriel F P Araujo

Supervisor: Profa. Genaina Rodrigues

Ricardo Caldas and prof. Claudio Menghi and prof. Patrizio Pelliccione

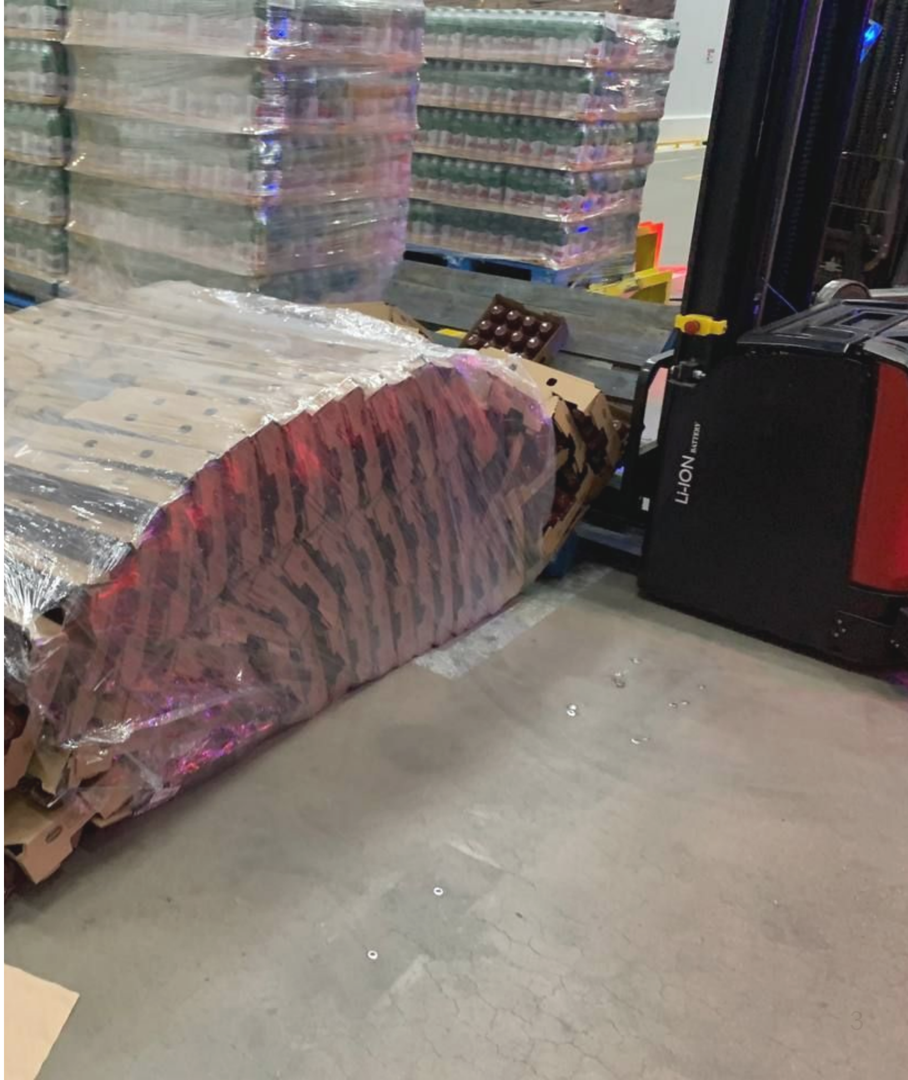
Story time!

- Autonomous forklift fault while picking a pallet.
- We corrected a bug but we forgot to fix one of the parameters



Testing autonomous systems

- Testing robots is expensive
  - Takes time
  - Dangerous
  - Lots of modules to debug
- We need a smart way to test and find problems:
  - Unit tests
  - Integration tests
  - Hardware testing
  - Hardware in the loop



So…

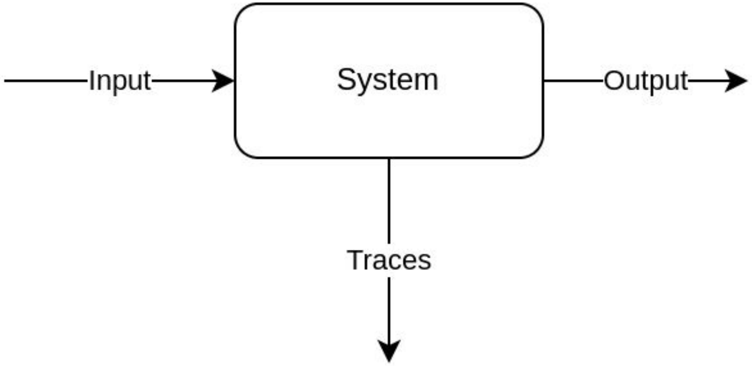
- How do we know that the robots are working?
- How do we know that any change in the codebase will not break other features?
- Are the parameters right?
- Does the different modules work together?

“*In the beginning…*”

As engineers, we can always:

- **Test** system
- **Record** data
- **Analyse** data

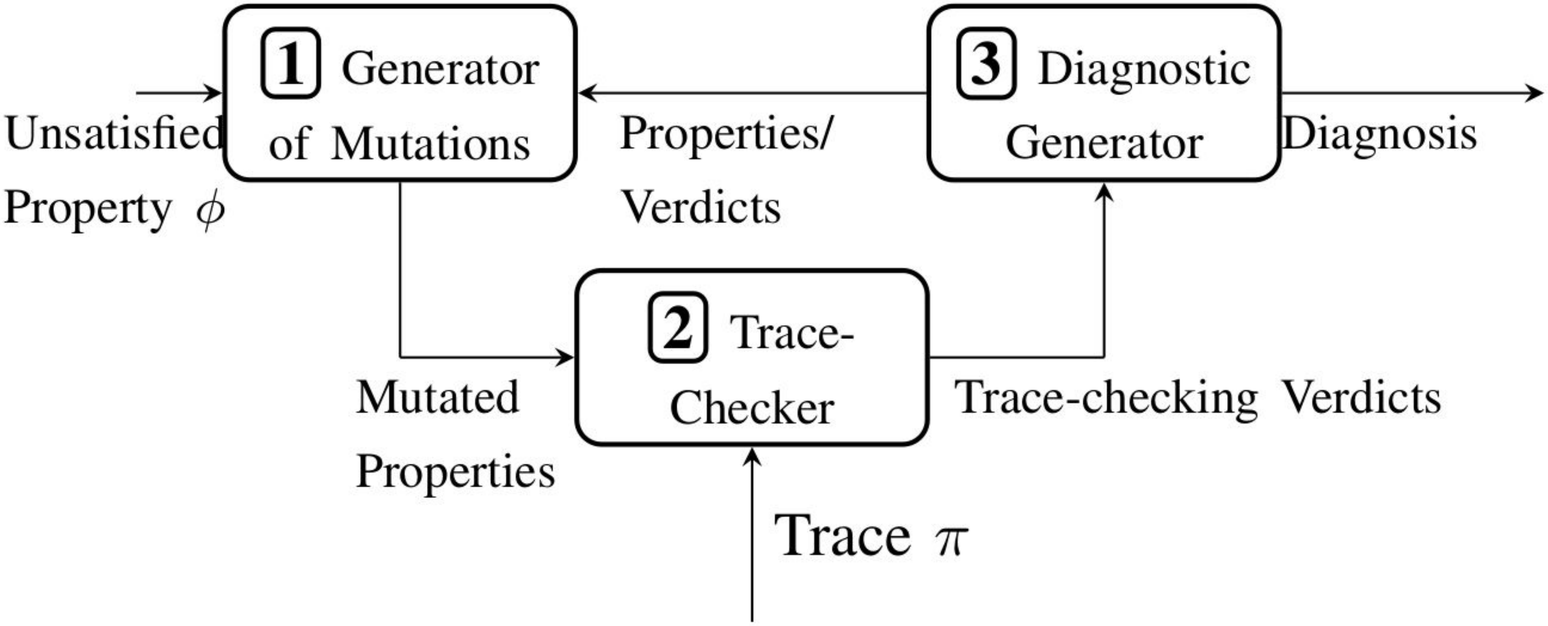
To diagnose *Any* system



Trace-checking

- Engineers record and analyse a system traces to check whether they obey the system's requirements
- We can automate the checking using a tool, a trace-checking tool
- For each property the trace-checking tool outputs a verdict (property satisfied or unsatisfied)

Our Approach



To specify properties: HLS (Hybrid Logic of Signals)

- Extends existing specification languages
  - Time-based languages (e.g. STL) and
  - Sequence-based languages (e.g. LTL)
- Target Cyber-Physical Systems
- Design goals
  - Timestamp variables
  - Index variables
  - Real-valued variables
- ThEodorE is the trace-checker for HLS properties

[1] Menghi, C., Vigano, E., Bianculli, D., & Briand, L. C. (2021). Trace-checking CPS properties: Bridging the cyber-physical gap. *Proceedings - International Conference on Software Engineering*, 847–859. https://doi.org/10.1109/ICSE43902.2021.00082

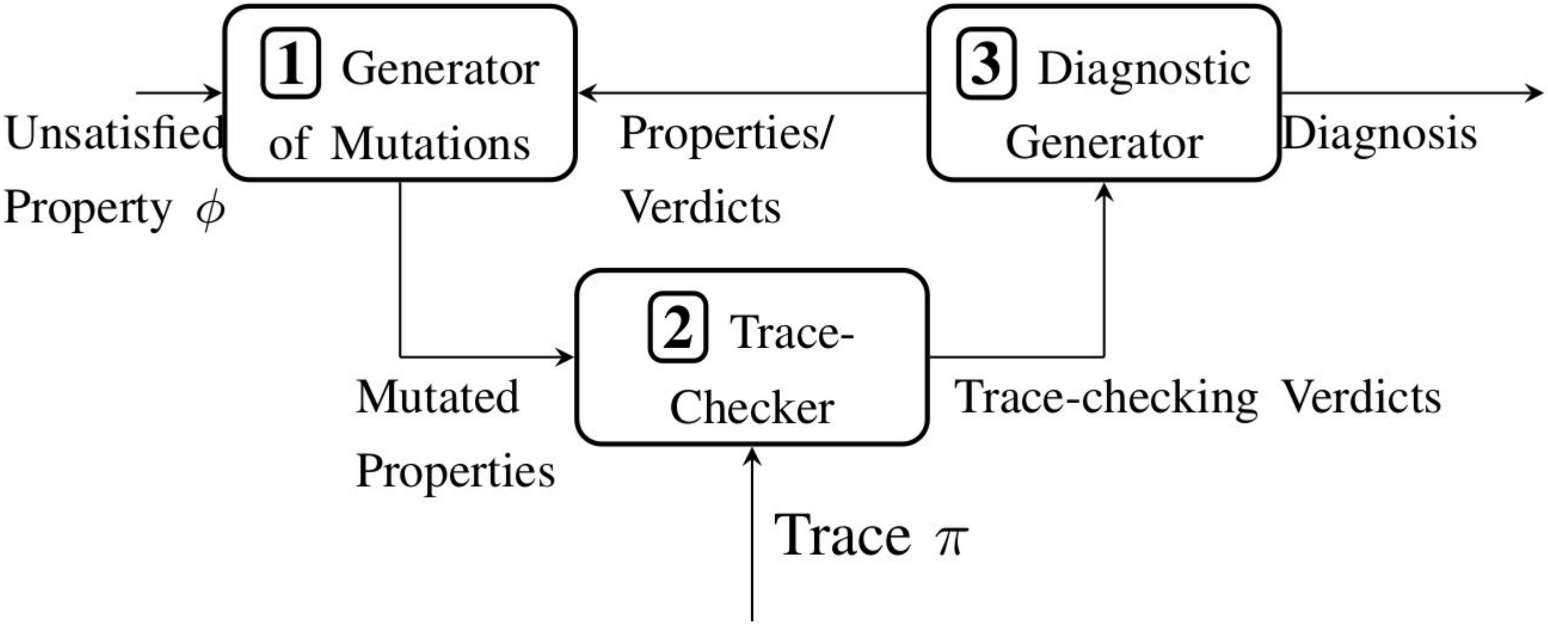
Genetic programming

- Search algorithm based on natural selection
- The genetic algorithm repeatedly modifies a population of individuals
- Used to expand the search space

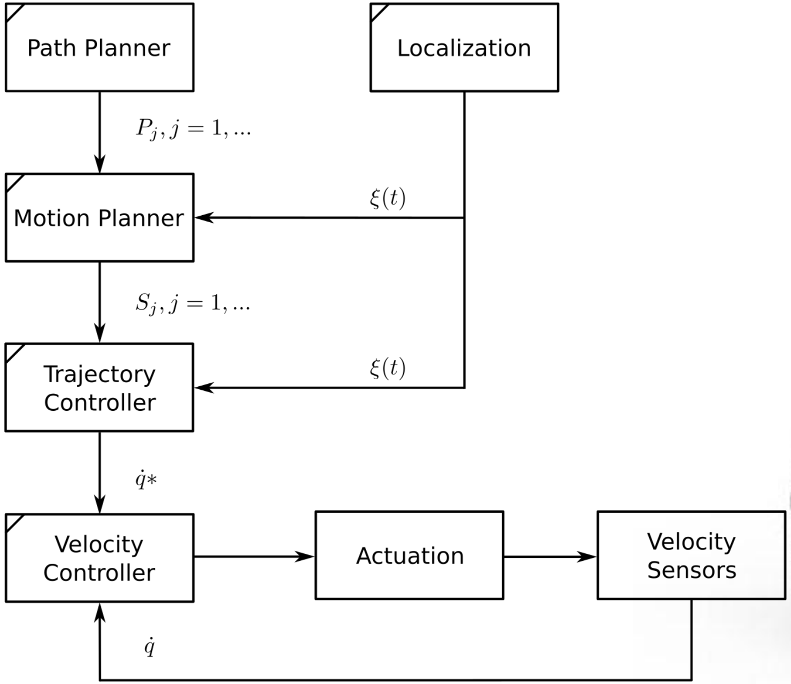
Decision Trees

- Supervised learning algorithm
- Used to classify GA’s mutated properties
- To find the root cause in the property

Our Approach



Running example: Autonomous car

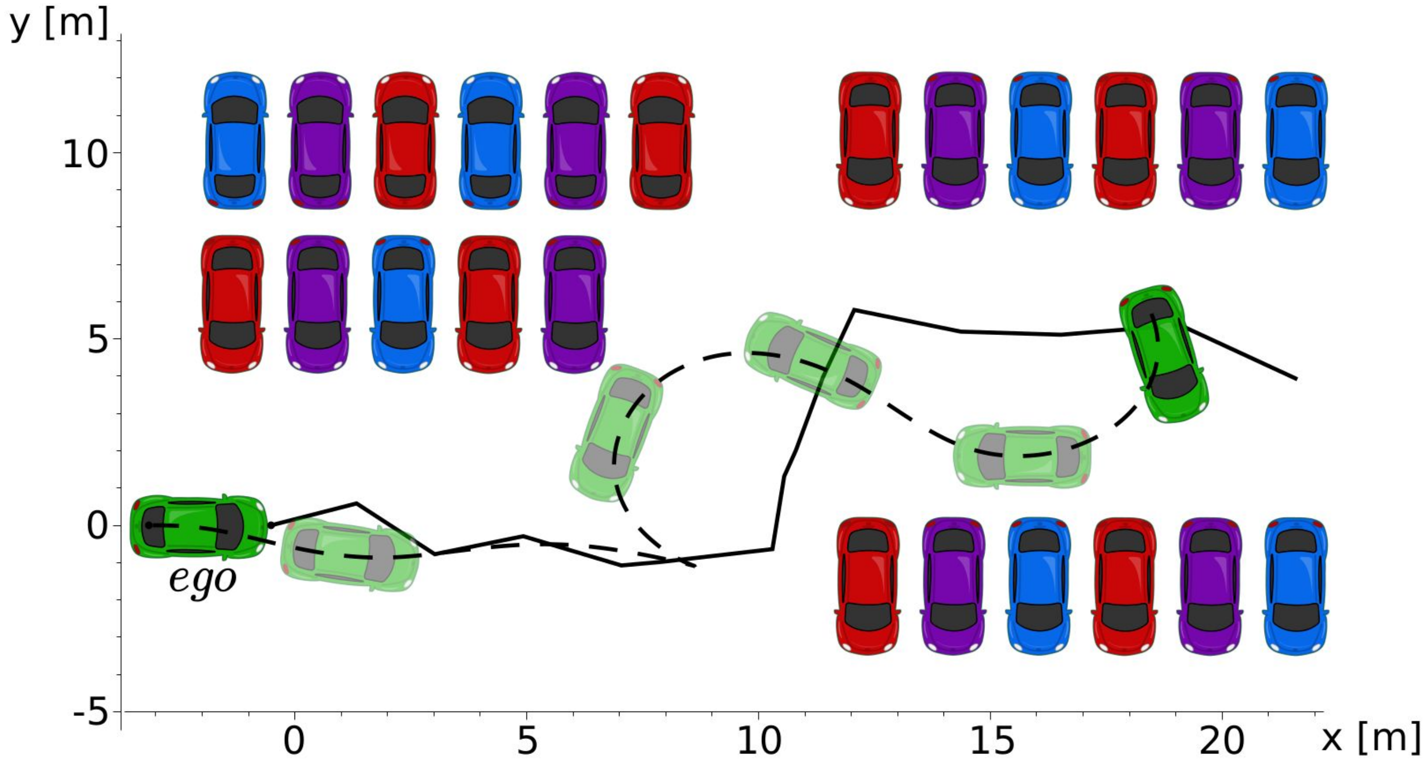


Running example: Requirement

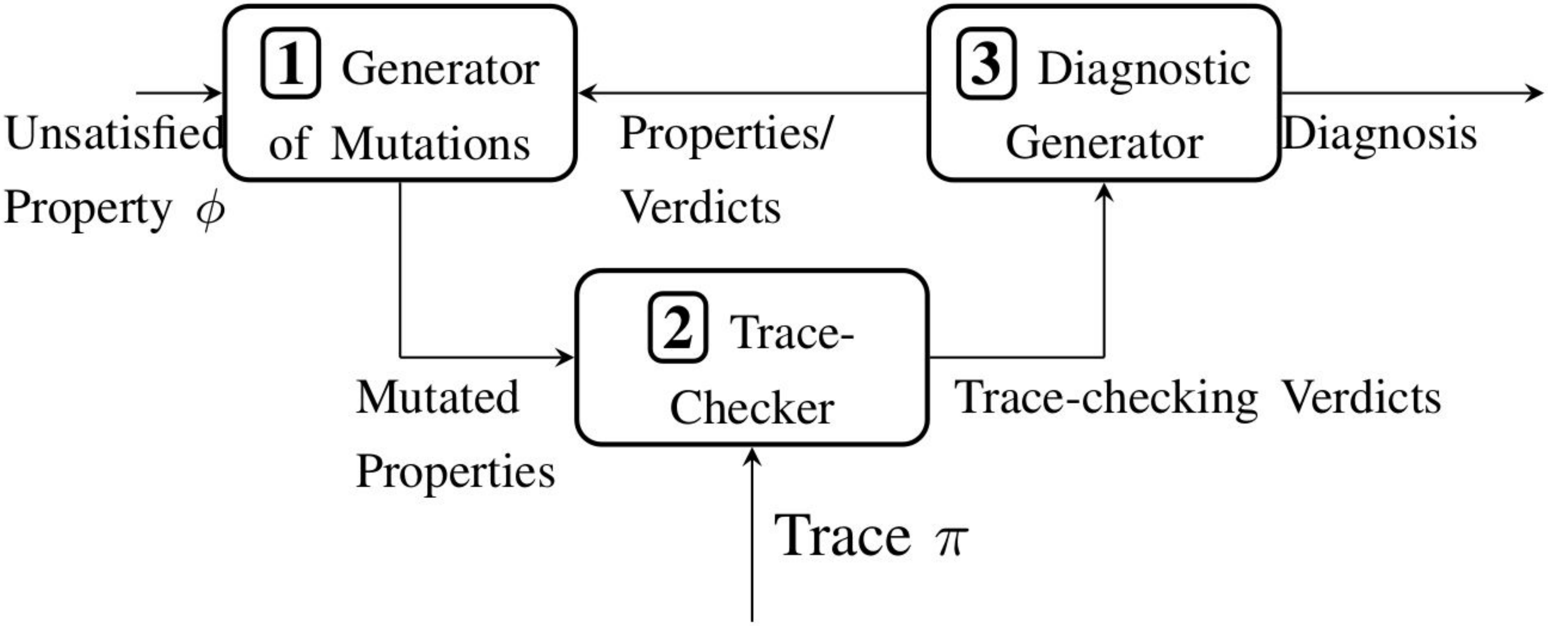
"The car has to follow the desired position in x axis with 20 cm tolerance and its distance to an obstacle must be greater than 45 cm"

$$\phi_o ::= \texttt{\textbf{forall}}\ \tau_0\ \texttt{\textbf{in}}\ [0,\infty)\ \texttt{\textbf{:}}\ (\texttt{d\_pos\_x}(\tau_0) - \texttt{v\_pos\_x}(\tau_0)) > 80\text{cm}\ \texttt{\textbf{and}}\ \texttt{d2obs}(\tau_0) > 45\text{cm}$$

Running example: Run (failure)



Our Approach



Generator of mutations

- Based-on genetic programming
- Creates several versions of the formula changing the terms

$$\phi_1 ::= \textbf{forall}\ \tau_0\ \textbf{in}\ [0, \infty)\ \textbf{:}$$
$$(\texttt{d\_pos\_x}(\tau_0) - \texttt{v\_pos\_x}(\tau_0)) > \mathbf{80}\text{cm}$$
$$\textbf{and}\ \texttt{d2obs}(\tau_0) > 45\text{cm}$$

$$\phi_2 ::= \textbf{forall}\ \tau_0\ \textbf{in}\ [0, \infty)\ \textbf{:}$$
$$(\texttt{d\_pos\_x}(\tau_0) - \texttt{v\_pos\_x}(\tau_0)) > 20\text{cm}$$
$$\textbf{and}\ \texttt{d2obs}(\tau_0)\ \mathbf{<}\ 45\text{cm}$$

Trace Checker

- For each generated formula, we check them using ThEodorE

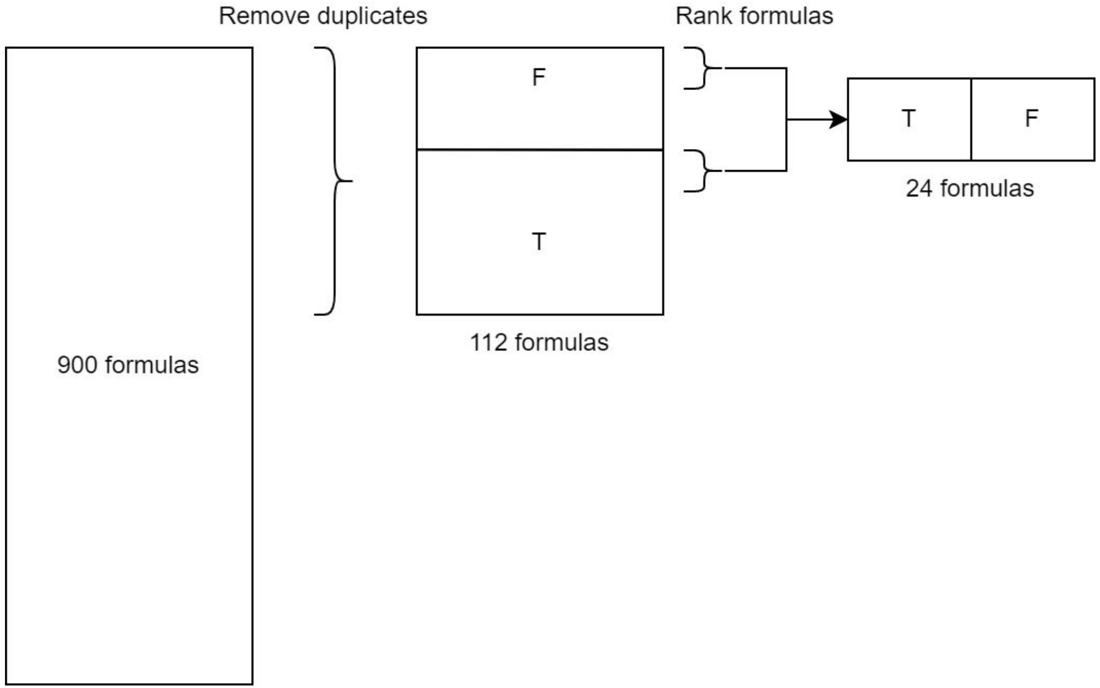
$$\phi_1 ::= \texttt{forall}\ \tau_0\ \texttt{in}\ [0, \infty)\ \texttt{:}$$
$$(\texttt{d\_pos\_x}(\tau_0) - \texttt{v\_pos\_x}(\tau_0)) > \texttt{80}\text{cm} \longrightarrow \text{False}$$
$$\texttt{and}\ \texttt{d2obs}(\tau_0) > 45\text{cm}$$

$$\phi_2 ::= \texttt{forall}\ \tau_0\ \texttt{in}\ [0, \infty)\ \texttt{:}$$
$$(\texttt{d\_pos\_x}(\tau_0) - \texttt{v\_pos\_x}(\tau_0)) > 20\text{cm} \longrightarrow \text{True}$$
$$\texttt{and}\ \texttt{d2obs}(\tau_0) \texttt{<} 45\text{cm}$$

[1] Menghi, C., Vigano, E., Bianculli, D., & Briand, L. C. (2021). Trace-checking CPS properties: Bridging the cyber-physical gap. *Proceedings - International Conference on Software Engineering*, 847–859. https://doi.org/10.1109/ICSE43902.2021.00082

Diagnostic Generator

After, we run a lot of new formulas (>1000), we choose the best ones based on how distant they are from the **original** formula.



DG: Ranking mutated properties

We use the Smith-Waterman algorithm to calculate the similarity between two formulas.

Formulas	Score
ϕ_o ::= **forall** τ_0 **in** $[0, \infty)$: (d_pos_x(τ_0) − v_pos_x(τ_0)) > 20cm **and** d2obs(τ_0) > 45cm	–
ϕ_1 ::= **forall** τ_0 **in** $[0, \infty)$: (d_pos_x(τ_0) − v_pos_x(τ_0)) > 80cm **and** d2obs(τ_0) > 45cm	25
ϕ_2 ::= **forall** τ_0 **in** $[0, \infty)$: (d_pos_x(τ_0) − v_pos_x(τ_0)) > 20cm **and** d2obs(τ_0) < 45cm	25
ϕ_2 ::= **forall** τ_0 **in** $[0, \infty)$: (d_pos_x(τ_0) − v_pos_x(τ_0)) > 20cm **or** d2obs(τ_0) < 45cm	20

MATCH = 3
MISMATCH = -3

Diagnostic Generator

We label the terms in the formula:

$$\phi_o ::= \texttt{forall}\ \tau_0\ \texttt{in}\ [0, \infty)\ \texttt{:}\ (\mathrm{d\_pos\_x}(\tau_0) - \mathrm{v\_pos\_x}(\tau_0)) > 20\mathrm{cm}\ \texttt{and}\ \mathrm{d2obs}(\tau_0) > 45\mathrm{cm}$$

Quantifier_1

Relational_1

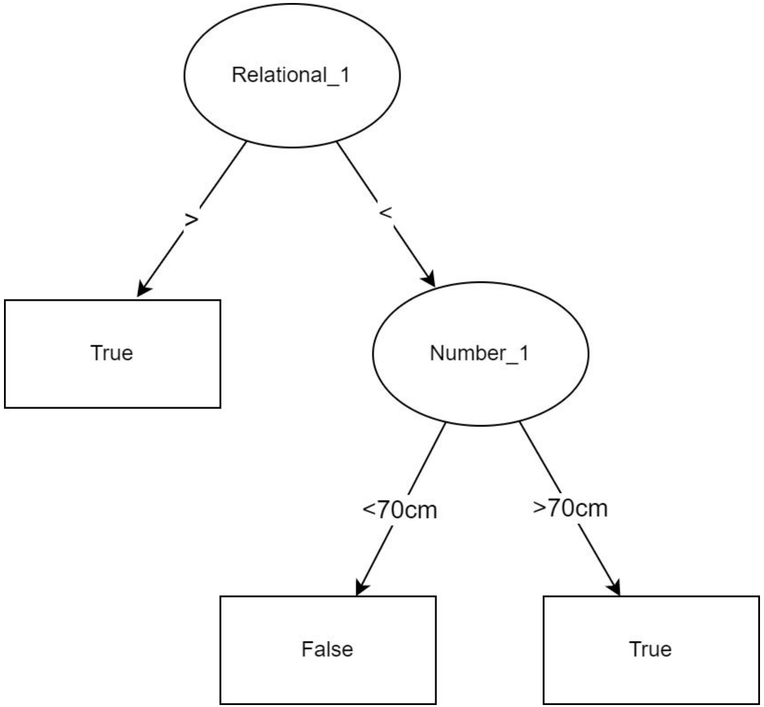
Number_1

Relational_2

Number_2

Diagnostic Generator

- Use J48 from WEKA to learn which terms change when verdict also changes
  - Unsupervised learning
  - Output a decision tree
  - Extract which terms on the formula are most relevant



Diagnostic Generator

We label the terms in the formula:

$$\phi_o ::= \texttt{forall}\ \tau_0\ \texttt{in}\ [0, \infty)\ \texttt{:}\ (\mathrm{d\_pos\_x}(\tau_0) - \mathrm{v\_pos\_x}(\tau_0)) > 20\mathrm{cm}\ \texttt{and}\ \mathrm{d2obs}(\tau_0) > 45\mathrm{cm}$$

Quantifier_1

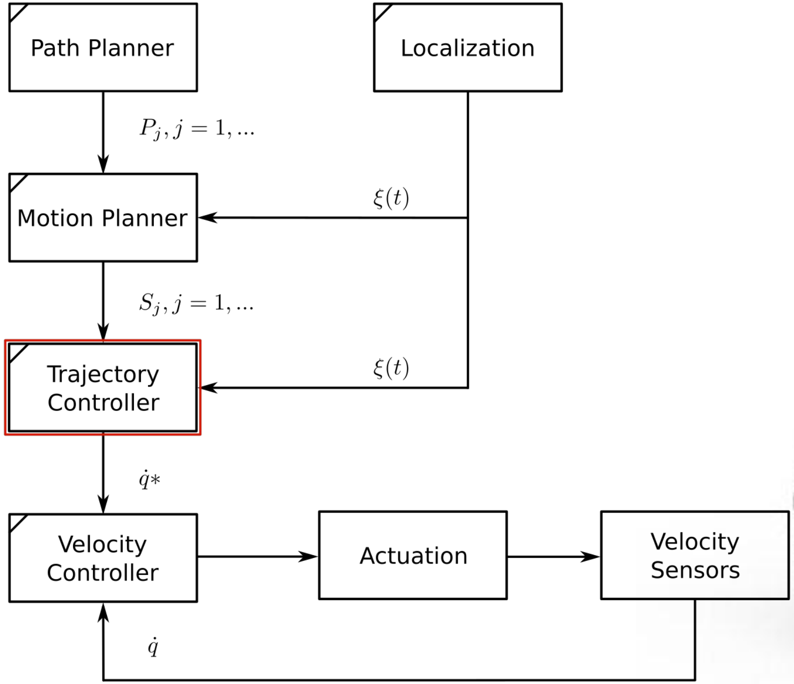
Relational_1

Number_1

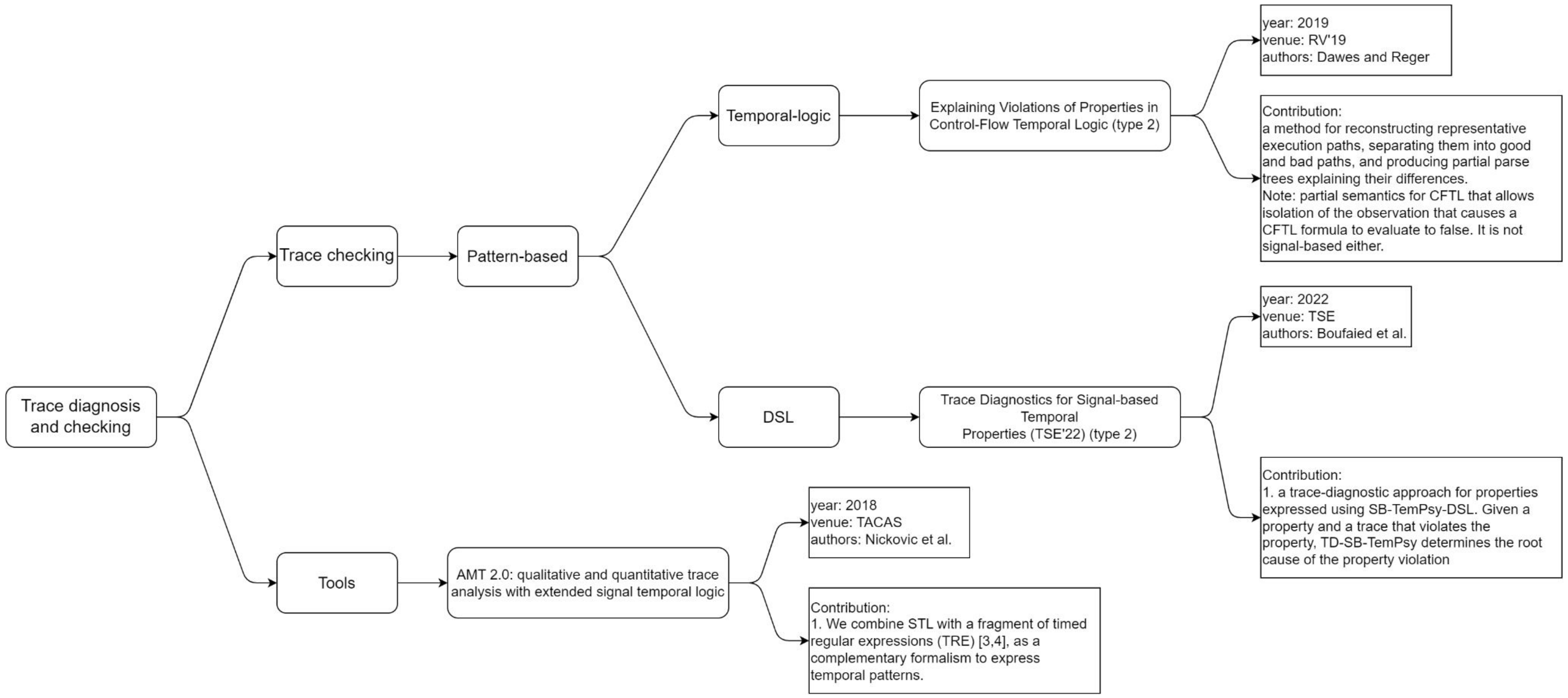
Relational_2

Number_2

Running example



Related work



Conclusions and what is next?

- We achieved so far:
    - Definition of the problem
    - Running example
    - Approach working
- Running the experiments for the evaluation.

Thank you