Search-Based Repair of DNN Controllers of AI-Enabled Cyber-Physical Systems Guided by System-Level Specifications

Deyun Lyu[1], Zhenya Zhang[1], Paolo Arcaini[2], Fuyuki Ishikawa[2], Thomas Laurent[3], Jianjun Zhao[1]

[1]Kyushu University, Japan

[2]National Institute of Informatics, Japan

[3]SFI Lero@Trinity College Dublin, Ireland

https://lyudeyun.github.io/

Background – CPS and AI Controller

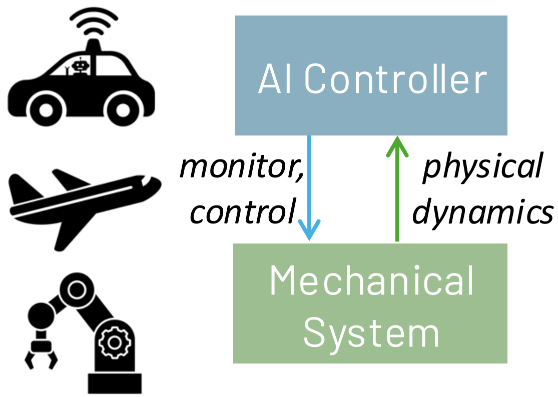
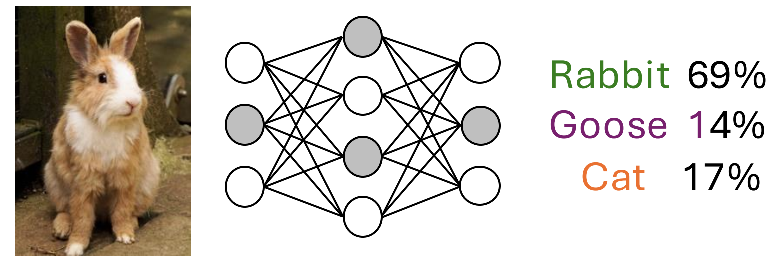
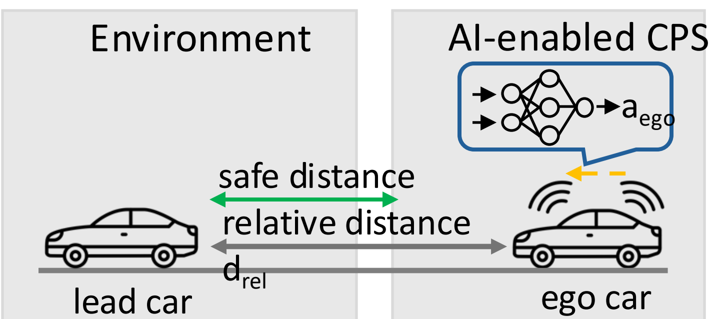
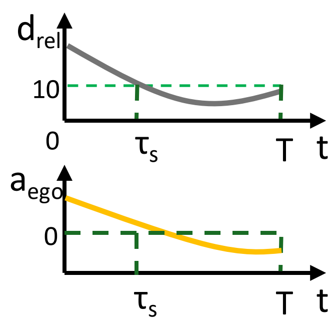


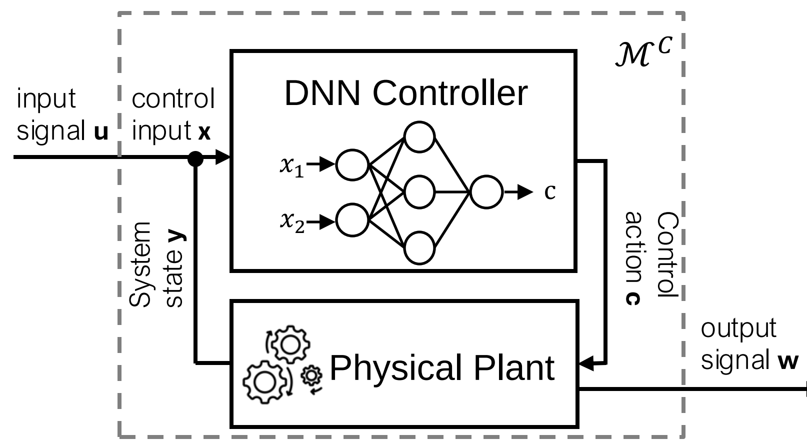
Image Classifier vs AI-CPS



A typical structure of image classifier







φ: Signal Temporal Logic (STL) specification

$$\varphi_{ACC} \equiv \square_{[0,50]} \left(d_{rel} \geq d_{safe} + 1.4 * v_{ego} \wedge v_{ego} \leq 30\right)$$

A typical structure of AI-enabled CPS

Adaptive Cruise Control (ACC) system

Background – STL and Its Robust Semantics

STL(Signal Temporal Logic) Syntax

Let $\vec{o} \in \mathbb{R}^d$ be a vector. In STL, an atomic proposition is represented as $\alpha :\equiv (f(\vec{o}) > 0)$, in which $f: \mathbb{R}^d \longrightarrow \mathbb{R}$ is a function that maps $\vec{o}$ to a real number. The syntax of an STL formula φ is defined as follows:

$$\varphi :\equiv \alpha \mid \perp \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \Box_I \varphi \mid \Diamond_I \varphi \mid \varphi_1 \mathcal{U}_I \varphi_2$$

STL Robust Semantics

The STL robust semantics tells how robust the output signal o satisfies/violates φ. The formal definition of STL semantics is as below:

$$\llbracket \mathbf{o}, \alpha \rrbracket := f\big(\mathbf{o}(0)\big) \qquad \llbracket \mathbf{o}, \neg\varphi \rrbracket := -\llbracket \mathbf{o}, \varphi \rrbracket$$

$$\llbracket \mathbf{o}, \varphi_1 \wedge \varphi_2 \rrbracket := \min\Big(\llbracket \mathbf{o}, \varphi_1 \rrbracket, \llbracket \mathbf{o}, \varphi_2 \rrbracket\Big)$$

$$\llbracket \mathbf{o}, \Box_I \varphi \rrbracket := \inf_{t \in I}\Big(\llbracket \mathbf{o}^t, \varphi \rrbracket\Big)$$

$$\llbracket \mathbf{o}, \varphi_1 \,\mathcal{U}_I\, \varphi_2 \rrbracket := \sup_{t \in I}\Big(\min\Big(\llbracket \mathbf{o}^t, \varphi_2 \rrbracket, \inf_{t' \in [0,t)} \llbracket \mathbf{o}^{t'}, \varphi_1 \rrbracket\Big)\Big)$$

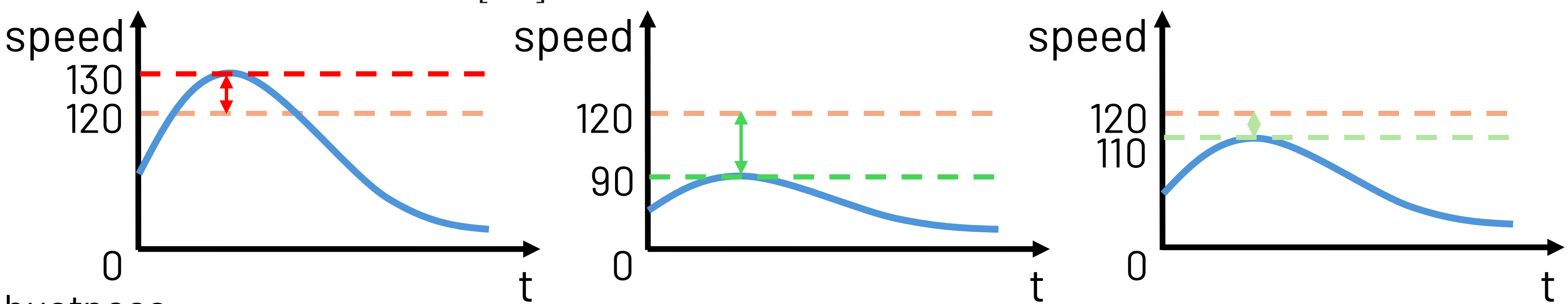
Background – STL and Its Robust Semantics

TL;DR

It can tell us how robust the output signal **o** satisfies/violates φ by robustness value of a given **o**.

System-Level Indicator
- Satisfaction Rob(**o**, φ) > 0
- Violation Rob(**o**, φ) < 0

Example: (φ: $\text{always}_{[0,T]}(\text{speed} < 120)$)



Robustness Value	-10	+30	+10
Boolean Satisfaction	Violation	Satisfaction	Satisfaction

Motivation

- **DNN can be faulty, even deadly.**

BBC

Home News Sport Business Innovation Culture Travel Earth Video Live

Man crushed to death by robot in South Korea

8 November 2023

Share

Emily Atkinson
BBC News

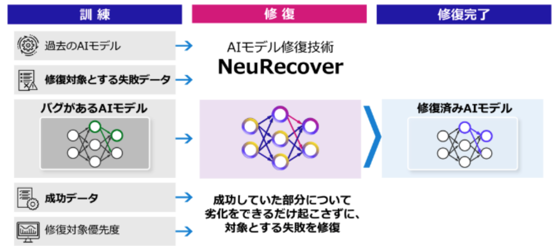
Tesla Autopilot feature was involved in 13 fatal crashes, US regulator says

Federal transportation agency finds Tesla's claims about feature don't match their findings and opens second investigation

- Emerging DNN quality assurance Technologies, e.g., **DNN repair.**

深層学習モデル修正技術に関する研究成果を SANER 2022 で発表します

0 B! Bookmark 0 シェアする



- **Lacks oracle for DNN components**, but has system-level specification.

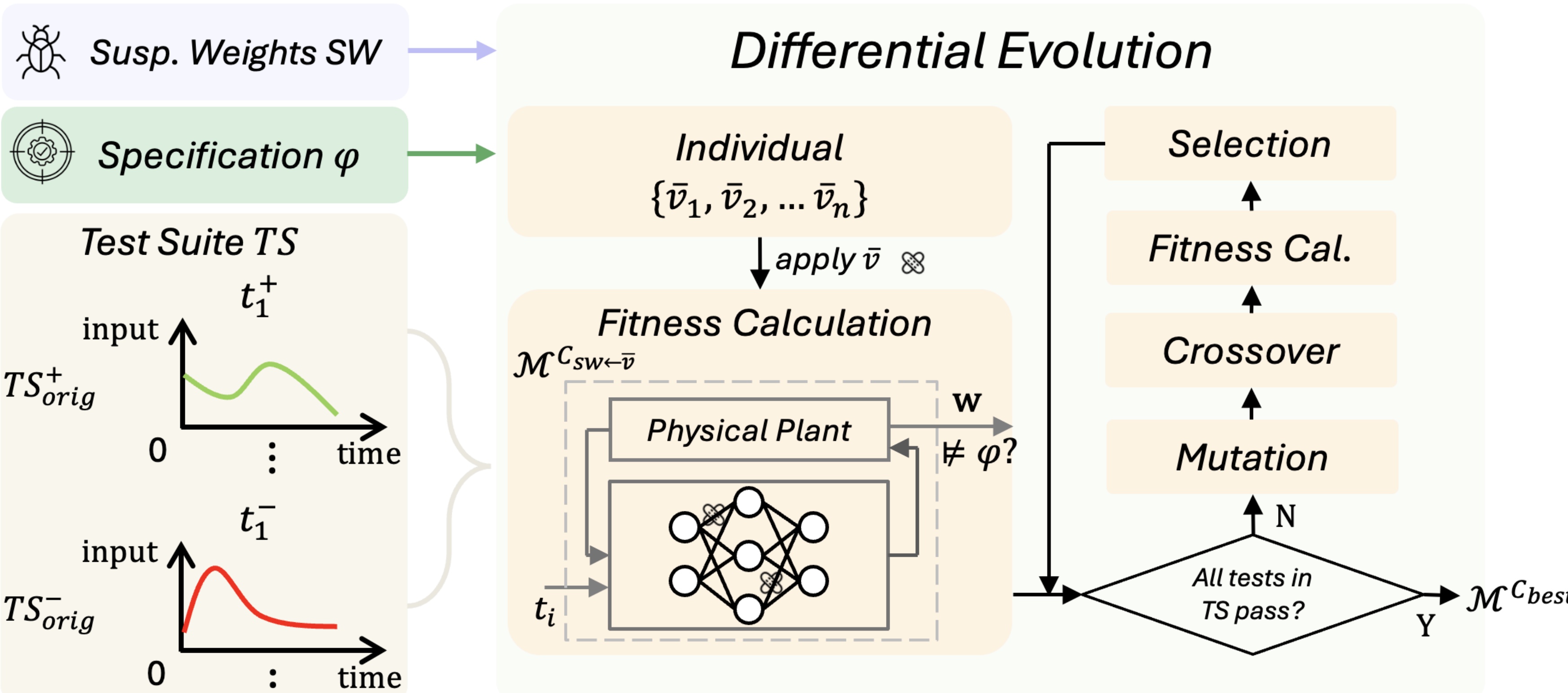
Self-driving cars safety project launched

15 April 2024

Share

Christian Fuller
BBC News, South East

Research Workflow of ContrRep



Our Approach - ContrRep

Input:

- A problematic AI-CPS $\mathcal{M}^{C_{orig}}$
- A system specification φ
- A test suite *TS* of input sequences for $\mathcal{M}^{C_{orig}}$

$$TS_{orig}^{+} = \{t \in TS \mid \mathcal{M}^{C_{orig}}(t) \vDash \varphi\}$$

$$TS_{orig}^{-} = \{t \in TS \mid \mathcal{M}^{C_{orig}}(t) \nvDash \varphi\}$$

- Correctness measure *CM*: the ratio of positive tests to all tests in *TS*.

$$CM\left(\mathcal{M}^{C_{orig}}, \varphi, TS\right) = \frac{\left|TS_{orig}^{+}\right|}{|TS|}$$

- A set of suspicious weights *SW*

Search-Based Repair of the DNN Controller

ContrRep casts the repair problem as a search problem, with an aim to find feasible values to replace the suspicious weights *SW*.

The **search variables** $\bar{x}$ of ContrRep are the possible alternative values for *SW*:

$$\bar{x} = [x_1, \ldots, x_{|SW|}]$$

The **search space** (the range of $\bar{x}$) is defined as follows:

$$\left[v_{orig} \cdot \ \delta^{-sign(v_{orig})}, v_{orig} \cdot \ \delta^{sign(v_{orig})}\right]$$

where $sign(v_{orig}) \in \{-1, 1\}$ identifies the sign of v_{orig}. Here we set $\delta = 2$.

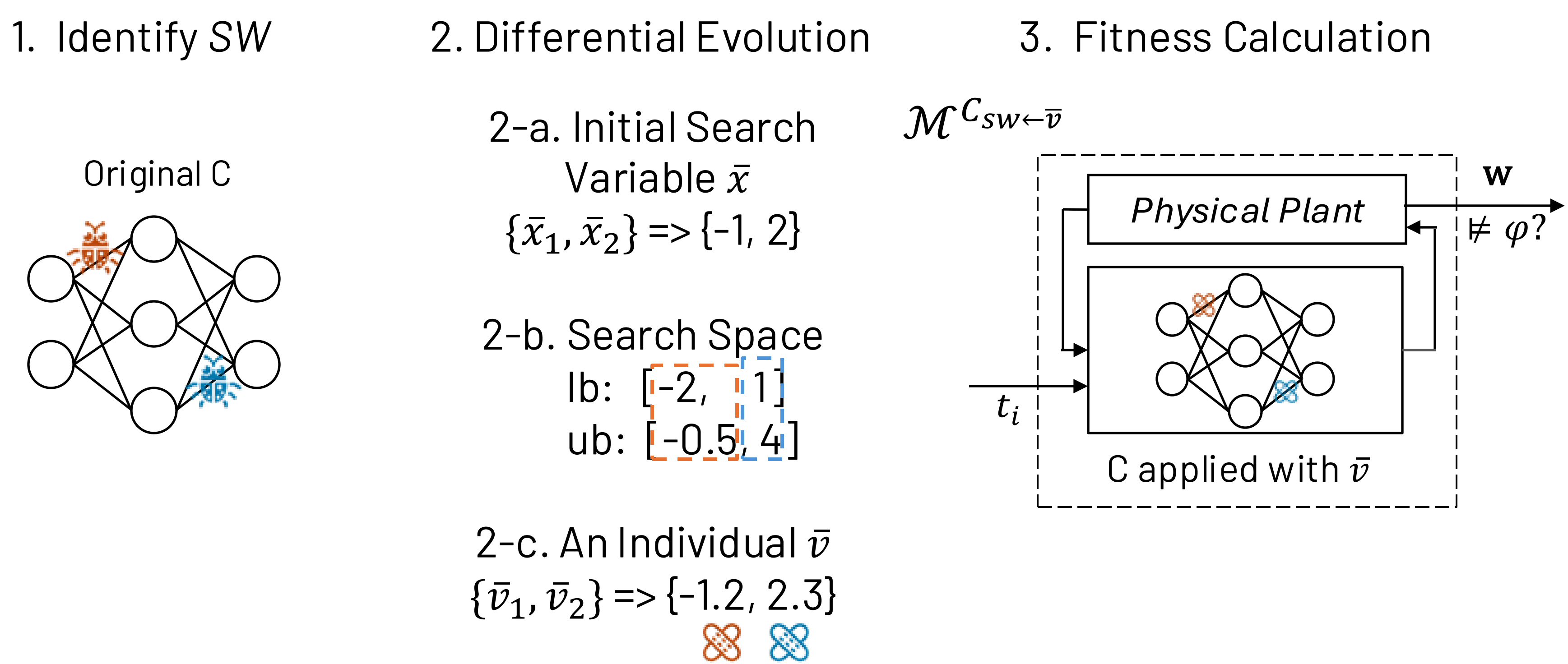
Given an individual $\bar{v}$, the **fitness function** that must be maximized is:

$$\text{fit}(\bar{v}) = CM(\mathcal{M}^{C_{sw \leftarrow \bar{v}}}, \varphi, TS)$$

Here, we adopt **Differential Evolution** (DE) algorithm as the underlying search algorithm.

Search-Based Repair of the DNN Controller

ContrRep casts the repair problem as a search problem, with an aim to find better values for the suspicious weights *SW*.



A schematic diagram of a ContrRep

Search-Based Repair of the DNN Controller

Four possible states of Rob($\mathbf{o}$, φ)

TS	Rob of orignial M	Rob of an individual
t_1^+	10	7
t_2^+	2	3
t_3^+	5	2
t_4^+	8	-4
t_1^-	-7	1
t_2^-	-3	4
t_3^-	-1	-3
t_4^-	-4	-2
fitness	4	5

A single fitness calculation

preserved: t_1^+, t_2^+, t_3^+

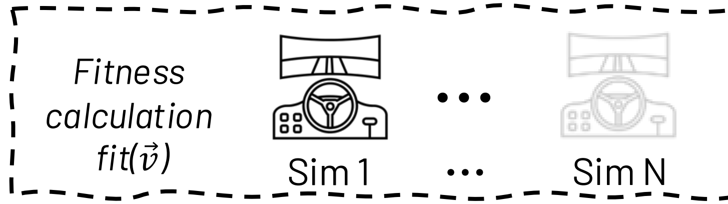
broken: t_4^+

repaired: t_1^-, t_2^-

not repaired: t_3^-, t_4^-

Speed Up the Fitness Computation – ContrRep$_{fast}$

How to speed up ContrRep?



numerous times of simulation needed

The heuristic behind ContrRep$_{fast}$

The **difficulty** of repairing negative tests or maintaining positive tests is related to the value of robustness.

Given a sorted test suite TS, for t_i^-, we start from t_1^-. If an individual can successfully repair t_1^-, we continue until this individual can no longer repair the next test or has repaired all the negative tests.

$$\text{fit}_{\text{fast}}(\bar{v}) = ApproxFit(\mathcal{M}^{C_{sw \leftarrow \bar{v}}}, \varphi, TS, TS_{orig}^{+}, TS_{orig}^{-})$$

sorted *TS*	Rob of orignial M	Rob of an individual
t_4^+	12	-
t_3^+	9	-
t_2^+	4	3
t_1^+	2	-1
t_1^-	-1	3
t_2^-	-3	2
t_3^-	-5	-1
t_4^-	-10	-

Experimental Design - Research Questions

RQ1. How is the repair performance of ContrRep? Is it affected by the quality of the fault localization results?

- In this RQ, we want to assess if ContrRep can actually repair DNN controllers. Moreover, we want to assess whether having imprecise fault localization results (i.e., containing weights that are not faulty) affects the repair effectiveness.

RQ2. Does $\text{ContrRep}_{\text{fast}}$ reduce the execution time of the repair approach? Does it affect the repair performance?

- In this RQ, we want to assess whether the heuristic implemented by $\text{ContrRep}_{\text{fast}}$ is indeed effective in reducing the repair time, and which is the reduction that must be paid in terms of repair performance.

RQ3. To what extent does ContrRep fix failing tests and break passing tests?

- In this RQ, we want to assess whether, while trying to repair the failing tests, ContrRep also breaks some passing tests.

Experimental Design - Benchmarks

ACC: ACC controls the acceleration of the ego car to keep a safe distance of it from a lead car.

$$\varphi_{ACC} \equiv \square_{[0,50]}(d_{rel} \geq d_{safe} + 1.4 \cdot v_{ego} \wedge v_{ego} \leq 30)$$

AFC: AFC is a powertrain control system developed by Toyota. It takes two signals, pedal angle, and engine speed, as the external inputs and produces two output signals AF that indicate the air-to-fuel ratio and AF_{ref} that is the reference value of AF.

$$\varphi_{AFC} \equiv \square_{[0,30]}\left(\left|AF - AF_{ref}\right| \leq 0.2 \cdot AF_{ref}\right)$$

Table 1: AI-enabled CPSs $\mathcal{M}^C$ and correctness measure of its faulty versions $\mathcal{M}^{C_{fault}}$ (CM ($\mathcal{M}^{C_{fault}}$, φ, TS) (%))

$\mathcal{M}^C$	ACC#1	ACC#2	AFC#1	AFC#2
Structure of C	[15 15 15]	[30 30 30]	[15 15 15]	[15 15 15 15]
#weights of C	450	1800	450	675
#blocks of $\mathcal{M}$	49	49	153	153
$CM(\mathcal{M}^{C_{fault}}, \varphi, TS)$	20	24	52	35

Xiaoqing Jin, Jyotirmoy V. Deshmukh, James Kapinski, Koichi Ueda, and Ken Butts. 2014. Powertrain Control Verification Benchmark. In Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control (Berlin, Germany) (HSCC '14). Association for Computing Machinery, New York, NY, USA, 253–262.

Experimental Design – Approaches, Inputs

Approaches

- ContrRep
- $\text{ContrRep}_{\text{fast}}$

Different Fault Localization Results

We built three sets of *SW*, for assessing a repair approach in settings of different complexity:

- $SW_{noNoise}$: the FL results are precise, i.e., $SW_{noNoise} = \mathcal{W}^{fault}$.
- SW_2: the FL results are not precise, i.e., $SW_2 = \mathcal{W}^{fault} \cup \{w_1, w_2\}$, *with* $w_1, w_2 \in \mathcal{W} \backslash \mathcal{W}^{fault}$.
- SW_4: similar to SW_2, i.e., $SW_4 = \mathcal{W}^{fault} \cup \{w_1, w_2, w_3, w_4\}$, *with* $w_1, w_2, w_3, w_4 \in \mathcal{W} \backslash \mathcal{W}^{fault}$.

Experimental Design – Evaluation Metrics 1

Evaluation Metrics

for RQ1 and RQ2:

1. **Effectiveness:** Correctness Measure ***CM***: the ratio of positive tests to all tests in *TS*.

$$CM\left(\mathcal{M}^{C_{orig}}, \varphi, TS\right) = \frac{\left|TS^{+}_{orig}\right|}{|TS|}$$

Note:

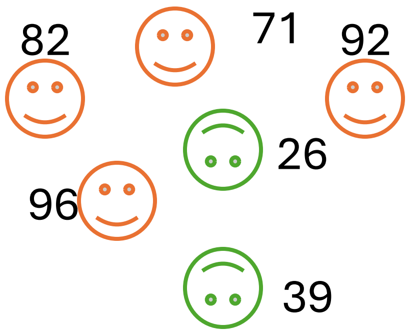
- we consider the value of the correctness measure ***CM*** obtained by the best-repaired model.

2. **Efficiency:** Total number of test executions ***ExecTests :*** used to assess the computational cost of a repair approach.

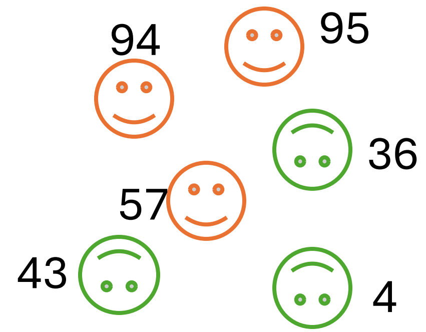
Experimental Design – Evaluation Metrics 2

1. Mann-Whitney U Test $(\alpha = 0.05)$

Is there any difference between two independent samples (APP1 vs APP2 over a given **EM**)?



A given EM by APP1 for ($\mathcal{M}^{C_{fault}}$, *SW*)



A given EM by APP2 for ($\mathcal{M}^{C_{fault}}$, *SW*)

2. Vargha and Delaney's $\hat{A}_{12}$ effect size

It can assess the strength of the significance.

APP1 is significantly **Better** than APP2 **(0.5, +⋈) :**

negligible: (0.5, 0.556) ✓ small: [0.556, 0.638) ✓✓

medium: [0.638, 0.714) ✓✓✓ large: [0.714, +⋈) ✓✓✓✓

András Vargha and Harold D. Delaney. A Critique and Improvement of the "CL" Common Language Effect Size Statistics of McGraw and Wong. Journal of Educational and Behavioral Statistics, Vol. 25, No. 2 (Summer, 2000), pp.101-132

Experimental Results – RQ1

RQ1. How is the repair performance of ContrRep? Is it affected by the quality of the fault localization results?

Table 2: Average $CM(\mathcal{M}^{C_{best}}, \varphi, TS)$ (%) of ContrRep and $ContrRep_{fast}$ across 10 runs.

		CONTRREP	$CONTRREP_{FAST}$
ACC#1	$SW_{noNoise}$	80.6	77.3
	SW_2	83.8	77.9
	SW_4	91.5	81.5
ACC#2	$SW_{noNoise}$	80.5	83.3
	SW_2	81.4	79.7
	SW_4	73.2	73.0
AFC#1	$SW_{noNoise}$	71.6	71.3
	SW_2	78.0	77.1
	SW_4	82.6	66.6
AFC#2	$SW_{noNoise}$	72.9	50.7
	SW_2	43.8	43.0
	SW_4	59.6	57.4

1. ContrRep outperforms $ContrRep_{fast}$ in terms of final repair accuracy.

2. The difference is often not very noticeable in many cases.

Experimental Results – RQ2

RQ2. Does ContrRep$_{fast}$ reduce the execution time of the repair approach? Does it affect the repair performance?

Table 3: RQ2 - Comparison of ContrRep and ContrRep$_{fast}$ in terms of $\boldsymbol{CM}(\mathcal{M}^{C_{best}}, \varphi, TS)$

		ContrRep vs. ContrRep$_{FAST}$
ACC#1	$SW_{noNoise}$	✓✓
	SW_2	✓✓✓✓
	SW_4	✓✓✓✓
ACC#2	$SW_{noNoise}$	✗✗
	SW_2	✓✓✓
	SW_4	✓✓
AFC#1	$SW_{noNoise}$	✗
	SW_2	≡
	SW_4	✓✓✓✓
AFC#2	$SW_{noNoise}$	✓✓✓✓
	SW_2	✓✓✓✓
	SW_4	✓✓

Table 4: RQ2 - Comparison of ContrRep and ContrRep$_{fast}$ in terms of ***ExecTests***

		ContrRep$_{FAST}$ vs. ContrRep	*ExecTests* (avg. 10 runs)	
			ContrRep$_{FAST}$	ContrRep
ACC#1	$SW_{noNoise}$	✓✓✓✓	7.0628e+04	2.5e+05
	SW_2	✓✓✓✓	7.4871e+04	3.0e+05
	SW_4	✓✓✓✓	9.9049e+04	3.5e+05
ACC#2	$SW_{noNoise}$	✓✓✓✓	3.9681e+04	2.5e+05
	SW_2	✓✓✓✓	4.8032e+04	3.0e+05
	SW_4	✓✓✓✓	6.2070e+04	3.5e+05
AFC#1	$SW_{noNoise}$	✓✓✓✓	1.0905e+05	2.5e+05
	SW_2	✓✓✓✓	1.2923e+05	3.0e+05
	SW_4	✓	1.7531e+05	3.07e+05
AFC#2	$SW_{noNoise}$	✓✓✓✓	8.4564e+04	2.5e+05
	SW_2	✓✓✓✓	1.0180e+05	3.0e+05
	SW_4	✓✓✓✓	1.2021e+05	3.5e+05

ContrRep$_{fast}$ has a significant advantage in terms of time spent on repair!

Experimental Results – RQ3

RQ3. To what extent does ContrRep fix failing tests and break passing tests?

Table 5: RQ3 - Number of repaired and broken tests (average across 10 runs)

		ContrRep				ContrRep$_{FAST}$			
		TS^{-}_{orig}		TS^{+}_{orig}		TS^{-}_{orig}		TS^{+}_{orig}	
		repaired	not repaired	preserved	broken	repaired	not repaired	preserved	broken
ACC#1	$SW_{noNoise}$	75.89%	24.11%	99.29%	0.71%	71.61%	28.39%	100.00%	0.00%
	SW_2	79.64%	20.36%	100.00%	0.00%	72.32%	27.68%	100.00%	0.00%
	SW_4	89.38%	10.63%	100.00%	0.00%	76.88%	23.13%	100.00%	0.00%
ACC#2	$SW_{noNoise}$	74.34%	25.66%	100.00%	0.00%	78.03%	21.97%	100.00%	0.00%
	SW_2	75.79%	24.21%	100.00%	0.00%	73.29%	26.71%	100.00%	0.00%
	SW_4	65.13%	34.87%	100.00%	0.00%	64.80%	35.20%	100.00%	0.00%
AFC#1	$SW_{noNoise}$	47.02%	52.98%	94.23%	5.77%	43.75%	56.25%	96.70%	3.30%
	SW_2	62.50%	37.50%	92.31%	7.69%	64.84%	35.16%	88.46%	11.54%
	SW_4	67.50%	32.50%	97.69%	2.31%	48.75%	51.25%	79.62%	20.38%
AFC#2	$SW_{noNoise}$	66.84%	33.16%	84.13%	15.87%	29.23%	70.77%	90.48%	9.52%
	SW_2	32.31%	67.69%	65.14%	34.86%	18.62%	81.38%	88.29%	11.71%
	SW_4	36.92%	63.08%	64.00%	36.00%	29.85%	70.15%	85.71%	14.29%

ContrRep and ContrRep$_{fast}$ can repair the broken test, while keeping the originally correct tests still correct!

Discussion and Future Work

- This work makes an early attempt to repair DNN controller in AI-enabled CPS. **There is a need for more research efforts** on the repair of such systems.

- In this work, we assume that the plant is correct and that the DNN controller must be fixed. However, some plants have hyperparameters that can be tuned, which can affect the behavior of the AI-enabled CPS. In future work, we plan to **investigate the combined repair of the physical plant and the DNN controller**.

- Besides repair, **AI-CPS enhancement approaches** also should be developed to further improve the quality of such systems.