FalCAuN − CPS Testing with Automata Learning

Masaki Waga · Kyoto University

Q. How to enhance system testing? e.g. Reusability, Explanation, Theoretical gurantee, …

Learning of formal model + Verification **Our Approach** Testing black-box CPS with

Robustness of \mathscr{M}' is always positive (Guaranteed by model checking)

Heuristic: Find *u* s.t. $\mathscr{M}(u) \neq \mathscr{M}(u)$ focusing on u making \mathscr{M} less robust

• Also the case for the formulas "related" to φ

Our Toolkit: FalCAuN (on Jupyter with Kotlin Kernel)

Robustness-guided equivalence test [Waga, HSCC'20]

Counterexample synthesis via Model Checking of strengthened formulas

<u>Idea:</u> Find evidence of $M \neq M'$ using inputs w/ low robustness i.e. use inputs leading "near dangerous" status

Assumption: $M \not\models \varphi$
 A $x \rho(M(u), \varphi) + \rho(M'(u), \varphi)$ Robustness of $\mathcal M$ can get negative for some inputs **Assumption:** $\mathcal{M} \not\models \varphi$

- ⁸ Definition 8 (Strengthening relation of LTL formulas). *For LTL for-*• Model checking is typically faster than equivalence testing
- σ obtained by model checking is often useful

<u>ldea:</u> Model checking of "related" specification can find useful evidence of $M \neq M'$

 α α α α α α β β β β α β α β α β α β α β β α \tilde{g} receive four ling in the decomplete from an φ where φ **Fact**: Counterexample σ of model checking progresses learning if \mathscr{M} does not violate φ with σ

1. For any $\mu, \nu \in \mathbf{LTL}$, we have $(\mu \vee \nu) \rightarrow (\mu \wedge \nu)$. 2. For any $\mu \in \mathbf{LTL}$, we have $\Diamond \mu \rightarrow \Box \Diamond \mu$. 3. For any $\mu \in \mathbf{LTL}$, we have $\Box \Diamond \mu \rightarrow \Diamond \Box \mu$.
4. For any $\mu \in \mathbf{LTL}$, we have $\Diamond \Box \mu \rightarrow \Box \mu$. \mathcal{L} 4. For any $\mu \in \mathbf{LTL}$, we have $\Diamond \Box \mu \rightarrow \Box \mu$. $\mathbf{15}$ *for any* $\mu \in \mathbf{LTL}$ *and for any indices* $i, j \in \mathbb{N} \cup \mathbb{N}$ $have \ \Diamond_{[i,j)} \mu \rightarrow \Box_{[i,j)} \mu.$ 6. For any $\mu, \nu \in \mathbf{LTL}$, we have $(\mu \mathcal{U} \nu) \rightarrow (\Box \mu \wedge \Box)$ *7. For any µ* 2 LTL *and for any indices i, j, i*⁰ ¹⁷ 2 N [*{*1*} satisfying* [*i, j*))

[*i* 0 *, j*⁰

)*, we have* ⌃[*i,j*)*µ* ⇢ ⌃[*i*0*,j*⁰ ¹⁸)*µ.*

¹⁹ *8. For any µ,* ⌫ 2 LTL*, if we have* ⌫ ⇢ *µ, we have ¬µ* ⇢ *¬*⌫*. 9. For any µ, µ*⁰ ²⁰ _ ⌫)*.*

- 22 *For any Particularly (R)NNs 12. For any µ,* ⌫*,* ⌫⁰ 2 LTL *satisfying* ⌫ ⇢ ⌫⁰ ²³ *and for any indices i, j* 2 N [*{*1*}* • Testing of Python classes, particularly (R)NNs
- Better illustration of falsifying executions
- Support of hyperproperties, e.g., to test robustness and fairness

This work is partially supported by JST ACT-X Grant No. JPMJAX200U, JST PRESTO Grant No. JPMJPR22CA, JST ERATO HASUO Metamathematics for
Systems Design Project (No. JPMJER1603). JST CREST Grant Number JPMJCR2012, and JSPS 29 dition 2, 10922470, and 39 of Definition and Definition 8, 19022470, processes, and 8, p. applying condition $\frac{1}{2}$ $\frac{1}{2}$ Systems Design Project (No. JPMJER1603), JST CREST Grant Number JPMJCR2012, and JSPS KAKENHI Grant Number 15KT0012, 18J22498, 19H04084, and 22K17873.

10. For any µ, ⌫*,* ⌫⁰ 2 LTL *satisfying* ⌫ ⇢ ⌫⁰

- Support numeric inputs (w/ symbolic automata)
- Extension for stochastic systems

, we have (*µ* _ ⌫) ⇢ (*µ* _ ⌫⁰

Experience (Shijubo, Waga, Suenaga, RV'21]

²¹)*.*

…

Spec. ¹ Spec. ²

Observations:

for learning since it is related to *φ*

Approach: Syntactically strengthen LTL formulas and conduct model checking with it

Eventually μ \longrightarrow \ldots \longrightarrow Always μ

Notes on formal guarantee

Can testing be faster?

Why tests passed?

Is it reliable?

println("maximum value of velocity: \${velocityValues.max()}") println("minimum value of velocity: \${velocityValues.min()}") bunch.show() maximum value of velocity: 120.21325006771389 minimum value of velocity: 0.0 Input to falsify: $[] (output(3) < 120.000000)$ 100 $\frac{1}{2}$ 50 10 12 14 16 time Output to falsify[] ($output(3) < 120.000000$) 100 output 50 10 12 14 16 18 time $[18]$: s0 2 aa / aaaaab / aaaa bb / aaaa Һа / яяяя /ba/aaac \aa/aaad \ab/aaad

Assumption: Target system can be modeled with a Mealy machine ℳ

- If ∀ input, eq. test eventually try it, then we can find any counterexample in the limit
- If we know the number of states of $\mathscr{M},$ we can stop eq. test with correctness guarantee (based on conformance testing, such as W-method [Chow, TSE' 78])
- Alternatively, we can stop with probably approximately correct (PAC) guarantee [Angluin, '87]