Challenges for Testing Autonomous Driving Systems:

Tackling Uncertainty in the Open Physical World and Machine Learning

Fuyuki Ishikawa, National Institute of Informatics, Japan
with ERATO-MMSD Project / QAML Project

f-ishikawa@nii.ac.jp
About Speaker

- Was not really a testing researcher
  - Quality by optimization (web services, clouds)
  - Formal engineering methods (VDM/Event-B/Alloy)
  - A little on testing (previously a kind of adaptive testing)
- Current focus (1): dependability of (autonomous) driving systems

*Today’s core topic: testing is the key*

- Current focus (2): dependability of machine learning-based systems
  
  *(tomorrow at MLST)*
Smart CPS

- **Smart Cyber-Physical** Systems

- Continuous aspects (requirements, environment, and system behavior)
- Use of machine learning
Our ERATO-MMSD Project for Dependable CPS

**Group 0: Metamathematical Integration**

\[ T + e \rightarrow T(e) \]

**Category Theory**

**Group 1: Heterogenous Formal Methods**

\[ T_1 + e_1 \rightarrow T_1(e_1) \]
\[ T_2 + e_2 \rightarrow T_2(e_2) \]

**Computer Science**

**Control Theory**

**Testing is here!**

**Group 2: Formal Methods in Industry**

**Advanced setting in autonomous driving**

**Group 3: Formal Methods and Intelligence**

**Heuristics, Evolutionary, Search-based approaches**

**Optimization**

**Automotive Industry**

**Practical setting to improve present practices**

---

**led by Ichiro Hasuo (NII)**

(2016-2022)

---

Apr 22 2019  
f-ishikawa@A-MOST'19
Physical World, Modeling?
Smart CPS: Physical

Smart Cyber-Physical Systems

◆ Continuous dynamics
◆ Differential/Difference equations
◆ Quantitative goals (distance, energy, probability, etc.)

Rigorous Modeling for CPS

- Scope beyond the classical software science
  - Hybrid models with not only discrete dynamics but also continuous dynamics for speed, energy, electricity, etc.

- Heterogenization or quantification of “models” in software science and engineering (e.g., formal modeling and verification)
  - Modeling and verification of energy consumption
    - [Nakajima, FM’15]
  - SMT Solver for nonlinear real formulas
    - [Gao, CADE’13]
  - Hybrid “program” logic and theorem proving
    - [Platzer, IJCAR’08]
  - Hybrid automata, co-modeling/simulation, …
Example: Differential Dynamic Logic

Differential dynamic logic

Precondition on position $x$ and velocity $v$

Deaccelerated by braking $-b$

Not go beyond the position $m$

C ≡ $v^2 \leq 2b(m-x)$ by discriminant

“Program” including state changes by differential equations

$x \leq m \land C(x, v) \quad [a := -b; \quad (x' = v; \quad v' = a)] \quad x \leq m$

Pragmatic Solution?

- Heterogenized verification is sometimes infeasible (as happened even for the discrete settings)
- Not scalable, or just undecidable
- Mathematically rigorous models often unavailable (e.g., Simulink models without clear semantics or just too complex controller simulators)

► We know testing as a pragmatic solution!

But what from the formal community?

# I’m not saying the communities are disjoint
Typical Setting

Simulink model of car behavior with automated braking

Inputs

User Behavior
- Throttle signals
- Brake signals

Environment Condition
- Initial velocity
- Location and movement of pedestrian
- Road conditions
- ...

Outputs

- Hit a pedestrian?
- Speed if hit

- We have executables (simulation models or code)
- At least we can execute it to evaluate the output for a certain input

→ Any “dangerous” case?

Apr 22 2019  

f-ishikawa@A-MOST'19
Physical World, “Testing”!
One Trend: Optimization-Driven Falsification

- Quantified properties

2 sample simulations

The velocity becomes \( \geq 80 \text{ km/h} \) within 10 seconds after B occurs?

[ Fainekos et al., Robustness of temporal logic specifications for continuous-time signals, TheoCompSci’09 ]
Figure from [ Akazaki et al, Time Robustness in MTL and Expressivity in Hybrid System Falsification, CAV’15 ]
One Trend: Optimization-Driven Falsification

Quantified properties quantitatively evaluated

2 sample simulations

The velocity becomes \( \geq 80 \text{ km/h} \) within 10 seconds after B occurs?

YES, after 10 seconds +20 km/h than required!

YES, after 10 seconds exactly the required value!

Fragile Satisfaction (close to Violation)

Robust Satisfaction

(similar evaluation can be done for time, too)

[ Fainekos et al., Robustness of temporal logic specifications for continuous-time signals, TheoCompSci’09 ]
[ Figure from [ Akazaki et al, Time Robustness in MTL and Expressivity in Hybrid System Falsification, CAV’1 ] ]

Apr 22 2019
f-ishikawa@A-MOST'19
One Trend: Optimization-Driven Falsification

- Falsification of quantified properties

2 sample simulations

The velocity becomes $\geq 80$ km/h within 10 seconds after B occurs?

YES, after 10 seconds +20 km/h than required!

YES, after 10 seconds exactly the required value!

Robust Satisfaction

Fragile Satisfaction (close to Violation)

(similar evaluation can be done for time, too)

- Optimization of “robustness score”

- e.g., solved by “trials-evaluation” cycles

[ Fainekos et al., Robustness of temporal logic specifications for continuous-time signals, TheoCompSci’09 ]
[ Figure from [ Akazaki et al, Time Robustness in MTL and Expressivity in Hybrid System Falsification, CAV’15 ]

Apr 22 2019  f-ishikawa@A-MOST'19
We Know This Kind: Search-based Testing

- Use of metaheuristic for test generation
- Again, “trials-evaluation” cycles
- Used also for generating a “good” test list-suite/vector

Test Suite Candidates

Selection & Evolution (mutation, crossover)

Utility Score Evaluation

(n+1)-th generation

e.g., 10 min. to produce the “same-level” test suite as human (in terms of code coverage and mutation score)

e.g., use in Facebook (Sapienz)

[Molina et al, Java Unit Testing Tool Competition - Sixth Round]
[https://code.fb.com/developer-tools/sapienz-intelligent-automated-software-testing-at-scale/]
Another Trend: Statistical Model Checking

- Computational models with probabilities
- “Probability calculation” of (un)desirable situations is too heavy (e.g., on Markov models)

- Hypothesis testing or probability estimation by a lot of trials (executions/simulations)
  - Capability (and demand) to tailor to the problem by using prior knowledge on the domain/problem

[ Jha et al., A Bayesian Approach to Model Checking Biological Systems, CMSB’11 ]
[ Zuliani et al, Bayesian statistical model checking with application to Stateflow/Simulink verification, FMSD’13 ]
A Spectrum for Control Systems

[ Simulation-Based Approaches for Verification of Embedded Control Systems, IEEE Control Mag.'16 ]
Example Topics in SBT

- Targeting Simulink models
  - Construct a “heat map” of the global space for focused search and also learn the scoring function to avoid many costly simulation runs


- Use formal techniques to divide the search space

[ S. Kobuna, et al., Validation of Control Software by Search-Based Testing Using Formal Methods, 2016 ]
Our Example: More Exploratory Falsification

- Apply a systematic exploration method
  - Avoid local optima (too much exploitation)
  - Also obtain informative data on the search space

**Monte-Carlo Tree Search**

- Common for bandit problems, AI Go game player, etc.
- Used to record the “smells” over the search space (time-staged) in the “trials-evaluation” cycles

What about making full throttle in the first time slot, …

[Zhang et al., Two-Layered Falsification of Hybrid Systems Guided by Monte Carlo Tree Search, EMSOFT/TCAD’18]

Apr 22 2019  f-ishikawa@A-MOST'19
Our Example: Falsification by Training

- Learn of how to falsify
  - Avoid search from scratch for every check
- Target a product family
  - Quick check during trial-and-error or for updates
  - Not only design parameters but also environmental parameters

[ Kato et al., Falsification of Cyber-Physical Systems with Reinforcement Learning, MT-CPS'18 ]

Quick falsification given a specific configuration (possibly a little out of the learned space)
Summary at This Point

More and more focus on “testing” or simulation-based approaches to CPS dependability

Your contributions with testing expertise will be really appreciated!
Typical Setting: Revised

Simulink model of car behavior with automated braking

Inputs

User Behavior
- Throttle signals
- Brake signals

Environment Condition
- Initial velocity
- Location and movement of pedestrian
- Road conditions
- ...

Outputs

- Hit a pedestrian?
- Speed if hit

We have executables (simulation models or code)
At least we can execute it to evaluate the output for a certain input

Any useful problem settings & solution techniques?
Generate a test suite that includes “diverse” shapes of output signals for human investigation

Specification mining (pareto-front search)

Search for formulae (in a given form) not satisfied

\[ \neg (\text{within time } t_0 (v \geq 100) \land \text{always } (w \leq w_0)) \]


[ B. Hoxha, et al., Mining parametric temporal logic properties in model-based design for cyber-physical systems, 2018 ]
Our Example: Stability Analysis

- Search for “unstable” areas of parameter values
  
  "Small changes of parameter values affect the safety score a lot"

- About design parameters (avoid or test intensively) and about environmental parameters (test intensively)

0.12 sec. difference in the initial kick-off changes non-crash to a 20km/h crash

[ Lee et al., Stability Analysis for Safety of Automotive Multi-Product Lines: A Search-Based Approach, GECCO’19 ]
Summary at This Point

- More and more focus on “testing” or simulation-based approaches to CPS dependability
- Complexity and uncertainty of various kinds
  - Approaches not limited to find “faulty” cases

Your contributions with testing expertise will be really appreciated!
Open World, System-Level
System-Level Example: Path Planner

- A component to decide the path of the ego-car

[Figure from McNaughton, Parallel Algorithms for Real-time Motion Planning]

- For the short term or the long term, anyway deciding the immediate control operations
- Imagine a simple one by optimizing the cost function: including distance with obstacles, lane conformance, limitation in possible behavior, comfort, etc.
Let’s Generate Test Cases!

- Space of parameters for a test case
  - Defined naively or at least bounded by possible configurations of the simulator
- Design parameters (fixed this time)
  - Weights in the cost function
- Environmental parameters
  - Road structure and behavior of signals
  - Weights, wheel size, etc. of the ego-car
  - Position and movement of other cars and pedestrians
Demo: First Example

- A possible configuration and a possible car behavior

(demo omitted in this online version: the ego-car just goes straight, almost but not crashing with a lot of other cars that go orthogonally to the ego-car)
What’s “Strange”?

“Cars crossing the street”?

- But this can reasonably occur, e.g., a car moving from a shopping mall to a restaurant at the opposite side
- Then, “too many”, which is too rare (almost impossible)

So, what is the (model of) assumptions?

Was the test “pass” or “fail” (if it is valid)?

- No crash, very efficient, and (physically) comfortable
- Not so safe (too close to other cars)

Naturally considered by the quantified verification

- Not appropriate in terms of the driving convention

So, what is the (model of) oracle?
More Focus on “Scenarios”

Think of “scenarios” or “situations” rather than the whole possibilities

- e.g., Ego-car stops properly for a Jaywalking pedestrian
- At least necessary for manual test design
- e.g., in Pegasus project

  [https://www.pegasusprojekt.de/en/about-PEGASUS]

- e.g., in Open Autonomous Safety

  [https://oas.voyage.auto/]

- Difficulties remain in defining a valid parameter space that matches with the intention of the scenario and also (often implicit) environmental assumptions
## From Open Autonomous Safety

### Ego approaches jaywalking pedestrian on undivided road

<table>
<thead>
<tr>
<th>Case</th>
<th>Scenario</th>
<th>Expected Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Ego approaches pedestrian. Pedestrian stands on side of the road.</td>
<td>Ego reduces speed.</td>
</tr>
<tr>
<td>B</td>
<td>Pedestrian enters roadway.</td>
<td>Ego comes to a stop in front of pedestrian.</td>
</tr>
<tr>
<td>C</td>
<td>Pedestrian exits the road.</td>
<td>Ego proceeds.</td>
</tr>
</tbody>
</table>

[https://oas.voyage.auto/scenarios/pedestrians-in-road.html](https://oas.voyage.auto/scenarios/pedestrians-in-road.html)
From Pegasus

Cut-in

- Start situation
  - Ego car (E) drives on highway lane
  - Other vehicle (C) on adjacent lane
  - Potentially further vehicles involved

- Evolution
  - C moves into E-lane in front of E

- Criticalities
  - C cuts in with little distance to E
  - C brakes after cutting in
  - Low TTC(E,C)

TTC: Time to collision

Cut-in (left, from behind) (regular traffic situation)

- Step 1:
  - Velocity [m/sec]: E, L: [22-36]; E-L: [-4,4]; C: [23-67]; C-E: [1,45];
  - Position [m]: L-E: [33,100]; E-C: [0,30];
  - Distributions: may be multivariate binomial (nontrivial correlations), or multivariate gamma-distributions
  - ...

- Step 2: Cut-in starts (C crosses lane marking) \( \Delta t: [2,20] \)
  - Velocity [\( \Delta m/sec \)]: L: [-7,+7]; C: [-50,+5]; C-E: [-5,40]; C-L: [-12,50]
  - Position [m]: L-E: [25,110]; C-E: [1,60]; L-E: [5,100]
  - ...

- Step 3: Cut-in completed (C has crossed lane marking halfway) \( \Delta t: [0.5,4] \)
  - Velocity [\( \Delta m/sec \)]: ...
  - ...

Demo: Second Example

How about this?

- Is this a “strange” case or not?
- If not,
- How to comprehensively include this kind of scenario?
- How to reach this specific crash case in the kind of scenario?

*(demo omitted in this online version: two cars goes orthogonally to the ego-car, the ego-car succeeds to avoid the first one by changing the path but crashes with the second one)*
Application of Existing Methods?

- White-box testing
  - What are coverage criteria to define “important” factors in the implementation?

- Model-based testing
  - For example, what is a model to serve as the foundation for (requirements-based) coverage criteria?

- Mutation analysis
  - What are “bugs” that should be detected in the implementation?

- We need more focus and insights about models on assumptions, “good” tests, bugs, etc.
Emerging Efforts

- Metamorphic testing
  - Define the expected output via a metamorphic relation: “changing an input this way will change the output that way”
  - (Applied to the sensing part)

  [ Zhou et. al., Metamorphic Testing of Driverless Cars, 2019 ]

- Formal analysis
  - “Responsibility-Sensitive Safety”
  - e.g., exclude “strange cases”

  [ Shwartz et. al., On a Formal Model of Safe and Scalable Self-driving Cars, 2017 ]
Summary at This Point

- More and more focus on “testing” or simulation-based approaches to CPS dependability
- Complexity and uncertainty of various kinds
  - Approaches not limited to find “faulty” cases
  - Intrinsic high uncertainty in systems that work in the open world with no clear boundaries/criteria

Your contributions with modeling and testing expertise will be really appreciated!
A Little on Machine Learning
Smart CPS: Smart

**Smart** Cyber-Physical Systems

Use of machine learning
- Data-driven development
- Black-box behavior
- Imperfect functionality
AI and ML

Everyone is talking about AI and ML, recently more about risks and concerns in terms of dependability

Accidents of autonomous cars

[http://www.dailymail.co.uk/news/article-3677101/Tesla-told-regulators-fatal-Autopilot-crash-nine-days-happened.html]

Technically unsolved problems at Google Photo

Improper online learning


[https://www.theguardian.com/technology/2015/jul/01/google-sorry-racist-auto-tag-photo-app]

[https://www.theguardian.com/technology/2018/jan/12/google-racism-ban-gorilla-black-people]
What will You Test? (1)

- When Honda sees ramen shop sign
  - First buzz in Dec 2017
    - [https://twitter.com/_gyochan_/status/938240168078622720]
    - [https://twitter.com/Bleu_kakeru727/status/937680760491753473]
  - Now a caution on the web site
    - http://www.honda.co.jp/hondasensing/feature/srf/
  - Second buzz in Sep 2018

Can you find beforehand or prevent adverse (?) news??
What will You Test? (2)

- From DeNA (May 2018)
  - Generate an image of a certain pose
  - Generate a movie given a pose sequence while changing the character

[https://dena.com/intl/anime-generation/]

What do you ensure to sell this to anime companies?
With ML, we obtain the behavior of a component (e.g., a neural net) \textit{inductively} from training data. Black-box, imperfect, non-testable (no or costly oracle), unexplainable, has adversarial examples, …

Existing principles do not work

Especially when arguing the product and its quality

Impacts on Testing (Not Comprehensive)

- Large black-box to handle various situations
  - Principle of unit testing (and debugging) invalidated
- Errors as normal behavior (i.e., not 100% accuracy)
  - Test fail does not mean existence of a bug
- Often non-testable [Weyuker, On Testing Non-Testable Programs, 1982]
  - No oracle or oracle too costly to build
- Unknown boundaries in the implementation, adversarial examples
  - Little confidence on how it works in “similar” cases
    (no “equivalence classes”)

[Goodfellow et al., Explaining and Harnessing Adversarial Examples, 2015]
V&V Research Emerging in SE Community

- SMT-based verification [CAV’17]
- Search for non-robust cases with “neuron coverage” [SOSP’17]
  [ICSE’18]
- System-level falsification [NFM’17]
- Safe reinforcement learning [AAAI’18]
- Verification by stochastic game [TACAS’18]
- Metamorphic testing case studies [ISSTA’18]
- Empirical study on bug statistics [ISSTA’18]
- Mutation Analysis [ISSRE’18]
- Updated coverage criteria [ASE’18]
- Fairness testing [ASE’18]
- Concolic testing [ASE’18]
- Surprise-driven testing [ICSE’19]
- Demonstration: structural coverage is misleading [ICSE’19]
  (and more) e.g., [https://github.com/TrustAI/Literature-on-DNN-Verification-and-Testing]
Example: White-box Structural Coverage

“How diverse output values of each neuron have been observed by a test suite?”

- Originally used for increasing diversity in the output “bad” (non-robust) cases
  

- Used as criteria to judge a test suite (dataset)

  [ Ma et al., DeepTest: DeepGauge: multi-granularity testing criteria for deep learning systems, 2018 ]

- Now said “could be misleading”
  
  - e.g., dependency rather on the search methods
  - e.g., natural inputs vs. adversarial inputs

  [ Li et al., Structural Coverage Criteria for Neural Networks Could Be Misleading, 2019 ]
Summary of This Talk

- More and more focus on “testing” or simulation-based approaches to Smart CPS dependability
- Complexity and uncertainty of various kinds
  - Approaches not limited to find “faulty” cases
  - Intrinsic high uncertainty in systems that work in the open world with no clear boundaries/criteria
  - ML is adding too high complexity and uncertainty even into the implementations we produce

*Your contributions with modeling and testing expertise will be really appreciated!*

*Investigation for essential question: what are “good tests”?*