Safety under Statistical and Environmental Uncertainties

Challenges in Cyber-Physical Systems with Machine-Learning Components

Ichiro Hasuo

National Institute of Informatics & SOKENDAI

Research Director, ERATO MMSD

Outline

* Cyber-physical systems under uncertainties
* Formal methods and testing for CPS
  * Hybrid system falsification
    * Logical connectives [CAV'19]
    * Causality in time [EMSOFT'18]
    * … as demonstration of combining logical and statistical
  * Other topics

Uncertainties in Cyber-Physical Systems 2020

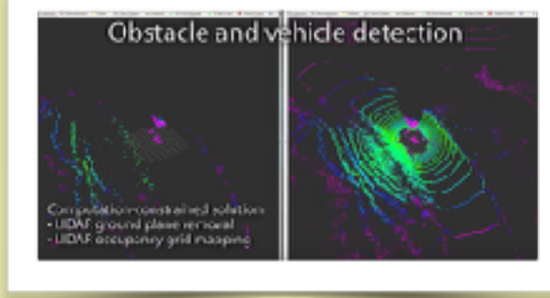
* Take automated driving…

Internal uncertainties:
never trust neural nets!

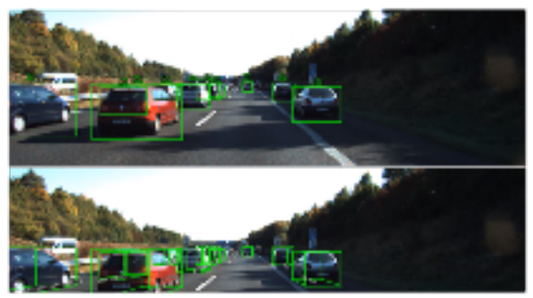
physical reality

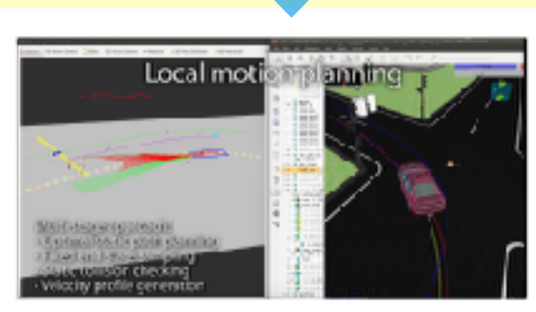
sensing

perception



object recognition





behavior planning
path planning

local planning
path tracking

actuation

External uncertainties:
other traffic participants, weather, sunlight, comets, earthquakes, …

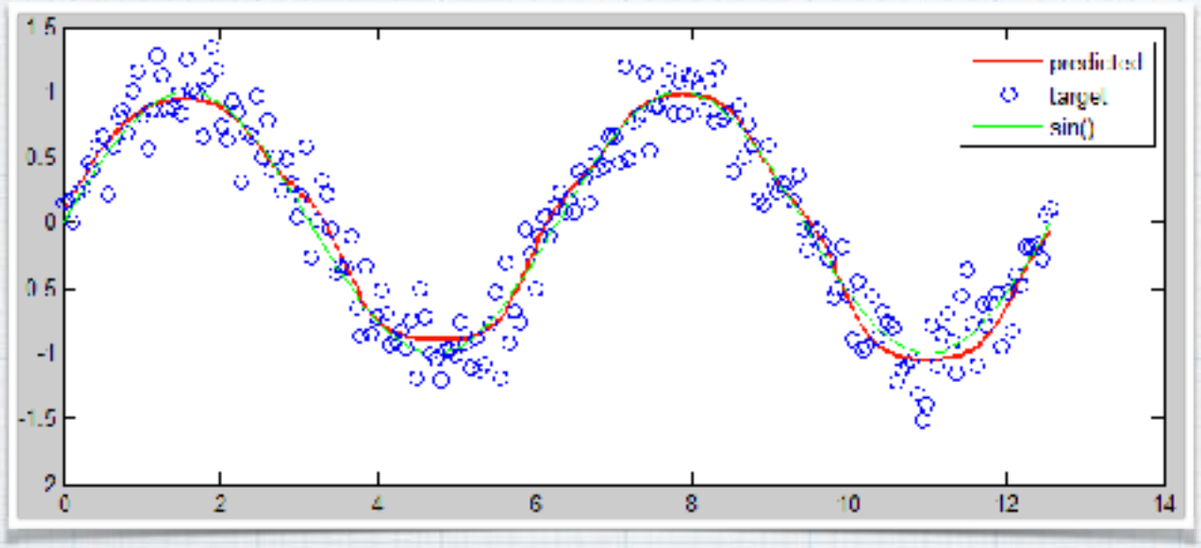
You walk in the dark,
guided by an apparatus

… which itself is a black box!

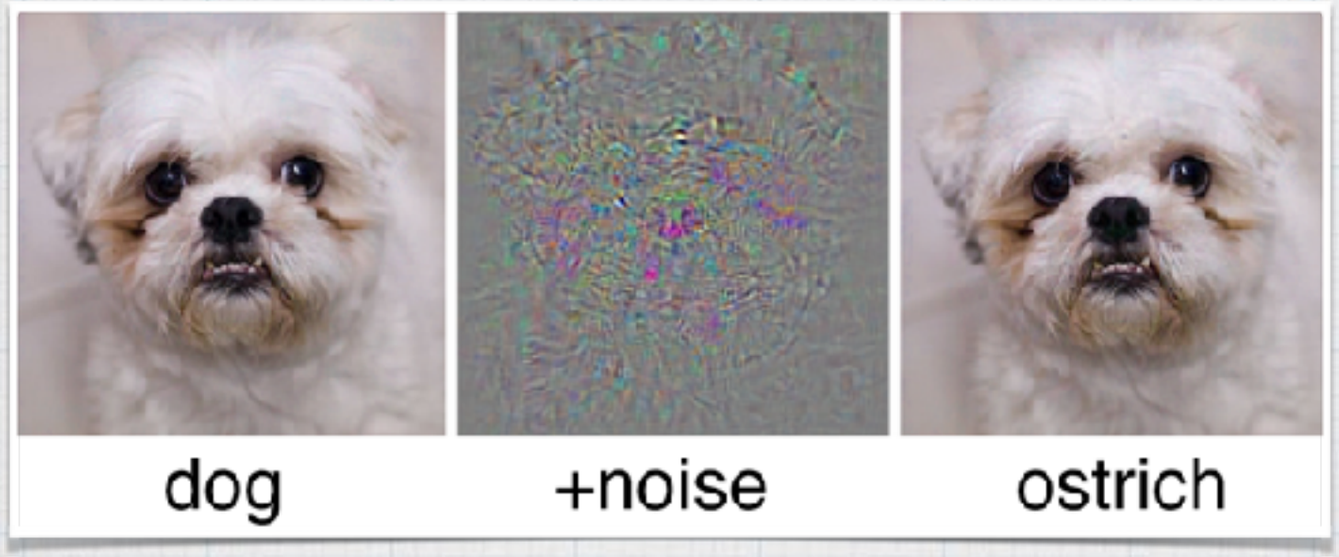
Never Trust Statistical AI

* They approximate.
  You never know when they are completely precise

Ground truth
➜ noisy data
➜ approximation by regression



Approximating the sine curve by NN
https://i.stack.imgur.com/l8qjP.png
https://stackoverflow.com/questions/1565115/unable-to-approximate-the-sine-function-using-a-neural-network



Adversarial example
https://i.stack.imgur.com/l8qjP.png
https://stackoverflow.com/questions/1565115/unable-to-approximate-the-sine-function-using-a-neural-network

Never Trust Statistical AI

* Question your “ground truth”

Ground truth
➜ noisy data
➜ approximation
by regression

(a) Husky classified as wolf (b) Explanation

Figure 11: Raw data and explanation of a bad model's prediction in the “Husky vs Wolf” task.

	Before	After
Trusted the bad model	10 out of 27	3 out of 27
Snow as a potential feature	12 out of 27	25 out of 27

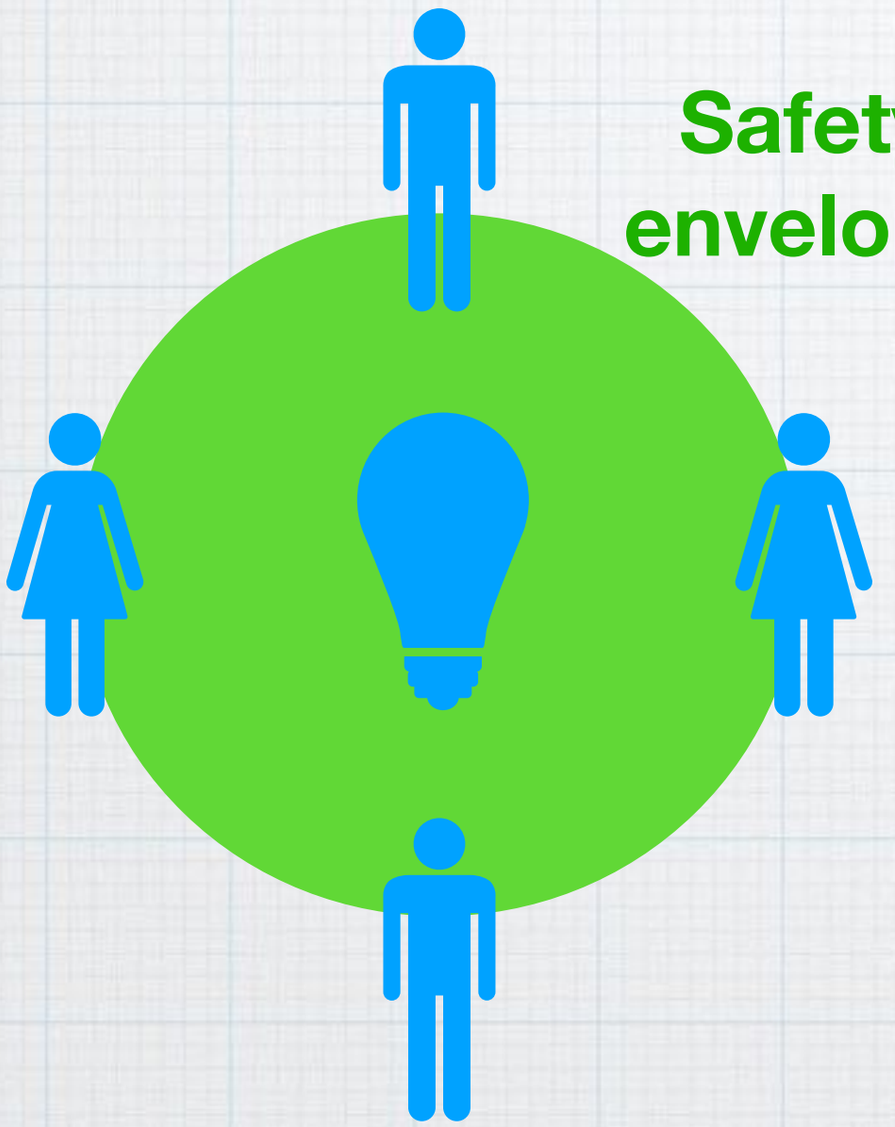
Table 2: “Husky vs Wolf” experiment results.

Husky vs Wolf? Snow vs No Snow?
https://arxiv.org/pdf/1602.04938.pdf

ML Component as "Super-Sharp but Not-Totally-Reliable Colleague"

* Often super sharp
  "Wonderful idea! How did you get it?"
* … for no clear reason
  "It just came down to me…"
* … and not always so
  "How come you made this stupid mistake?"
* Great to have one in a team; you don't want all to be like that

ML Component in Safety-Critical Cyber-Physical Systems

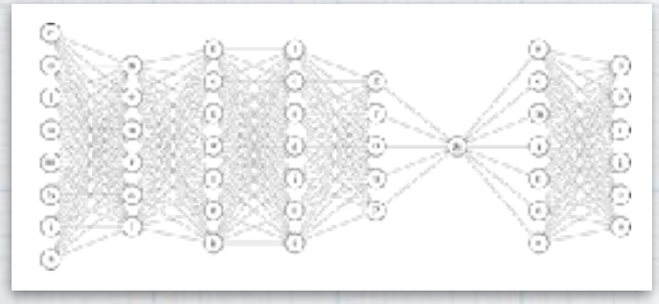


* Take its opinion with a pinch of salt
* **Suggestion**, instead of **decision**

Extreme example: proof check

* System-level assurance

Statistical AI vs Formal Deduction



$$\frac{A \quad A \supset B}{B}$$

Statistical AI		Formal Deduction
Allow noisy data	Errors in input	Axioms are absolute
No guarantee	Correctness of concl.	Logical guarantee (mathematical proofs)
High Automatic pattern discovery from data	Scalability	Low Manual preparation of axioms
Low Decision making by “weights”	Explainability	High Explicit deduction processes as proofs

Questions

* **Quality assurance** of CPS
  under **internal** (ML) and **external** (env.) **uncertainties?**
* How to combine the two worlds?

statistical	logical
continuous	discrete
noisy	rigorous
data-based	rule-based
bottom-up	top-down

Outline

* Cyber-physical systems under uncertainties
* Formal methods and testing for CPS
* Hybrid system falsification
  * Logical connectives [CAV'19]
  * Causality in time [EMSOFT'18]
  * … as demonstration of combining logical and statistical
* Other topics

Formal Methods

* The general body of mathematical techniques, originally in **software science**
* Goals:

Successes in software/ ICT

* IC design (Intel, …)
* device drivers (Microsoft)
* light-weight FM (Facebook)
* …

Verification

* **Input**:
  * a system model $\mathcal{M}$
  * a specification φ
* **Output**: if $\mathcal{M} \models \varphi$ or not
  * w/ a proof, if yes
  * w/ a counterexample,

Synthesis

* **Input**:
  * a specification φ
* **Output**: a system $\mathcal{M}$ such that $\mathcal{M} \models \varphi$
  * or: a parameter of a given (partial)

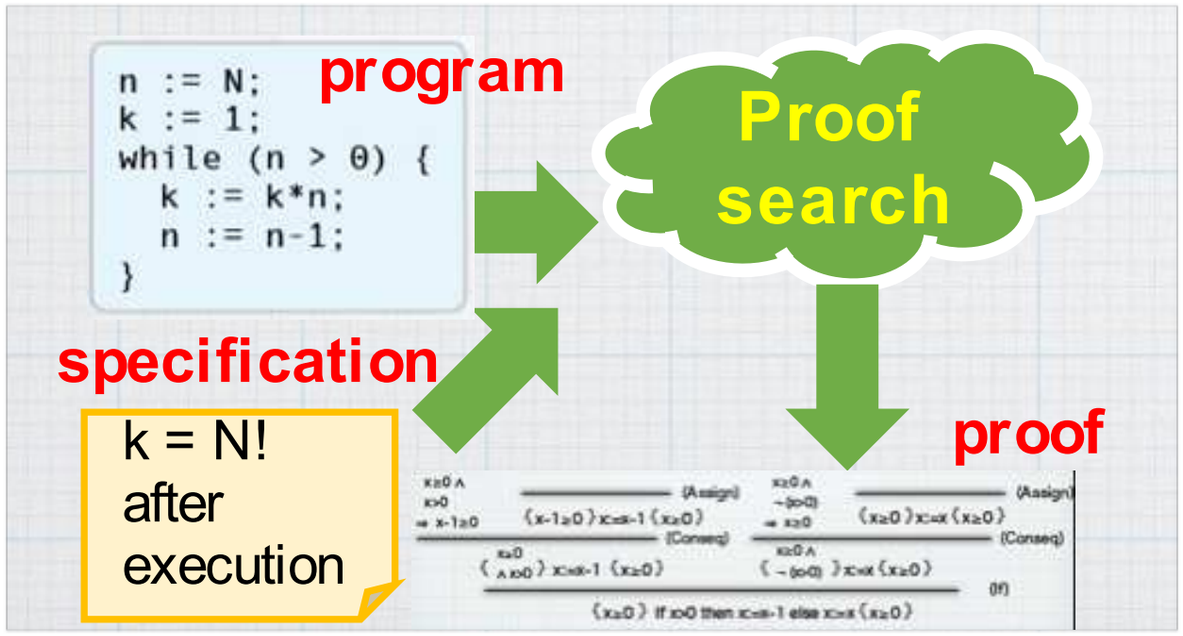
Specification

Expressing a property desired in a formal language

* machine-representable
* basis for verif. & synthesis

Verification by Theorem Proving

- Example: program verification by Hoare logic
  - Proof search can be automatic or interactive (proof assistants like Coq, Agda, PVS, Isabelle, …)
  - A mathematical proof guarantees correctness
    - Unlike testing (empirical guarantee)
  - Can cover infinitely many inputs/environments (in principle)
- Write “let ***i*** be the input” in the proof!



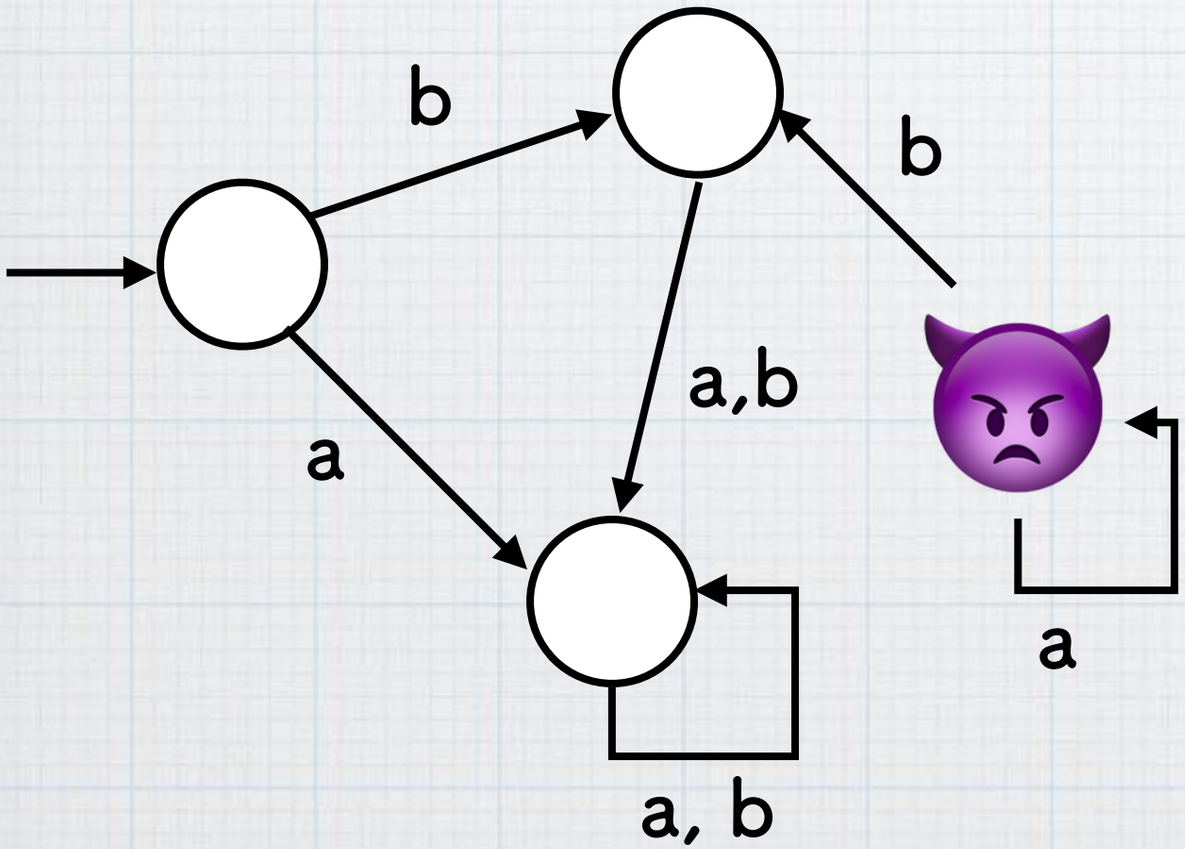

```
Theorem forall_exists : (forall P : Set->Prop,
          (forall x, ~(P x)) -> ~(exists x, P x)).
Proof.
  intros P.
  intros forall_x_not_Px.
  unfold not.
  intros exists_x_Px.
  destruct exists_x_Px as [ witness proof_of_Pwitness].
  pose (not_Pwitness := forall_x_not_Px witness).
  unfold not in not_Pwitness.
  pose (proof_of_False := not_Pwitness proof_of_Pwitness).
  case proof_of_False.
Qed.
```

A Coq proof of ∀x. ¬P(x) → ¬ ∃x. P(x)

Verification by Model Checking

* "Automated proof" by algorithms for automata
  * Proofs reduced to graph reachability
* An automaton is finite ➜ "automated proofs" by enumeration

* Example: Check if
  we never reach 😈 under any input word (*)



* (Wrong)
  Check the spec (*) for **all** input words
  (How many words? Infinite!!)
* (Correct)
  Compute the reachable set
  (graph exploration, saturates within finite time), and check if
  😈 is not there

History of Formal Methods

Software gets bigger

network, distributed, concurrent

meet physical dynamics

rise of statistical ML

- **1970's** Initial studies on software quality assurance by formal methods (**software verification**).
  Theorem proving, model checking

- **1990's** **Software verification** put to use, realistic tools
  Theorem proving: Isabelle, Coq, PVS, …
  model checking: SPIN, SMV/NuSMV, mCRL2, PRISM, Uppaal, …

- **2006** **Cyber-physical systems** as a new application domain.
  Software verification + control theory

- **2016** **Machine learning systems** as a new application domain.
  Formal/logical deduction + statistical inference

CPS Research, So Far (the V&V Aspect)

CPS
(esp. hybrid systems)

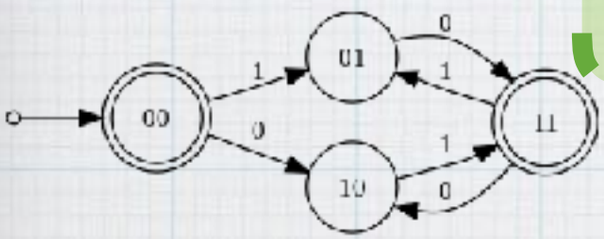
Analysis

Formal Methods

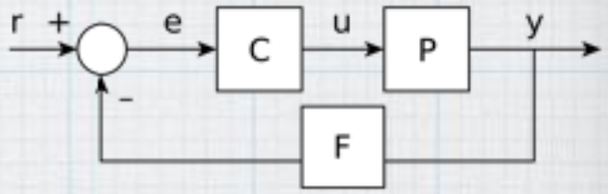
Control Theory

Collaboration

$\Box(p \Rightarrow \Diamond q)$



$x' = f(x, u)$



- US and Europe have had systematic efforts
  - US: CPS Initiative of NSF (2006-)
  - Europe: Quantitative extension of formal methods (probability, time, …)
- Need to catch up…?
  ➜ They haven't necessarily been so successful in real-world applications

CPS Research, So Far (the V&V Aspect)

CPS
(esp. hybrid systems)

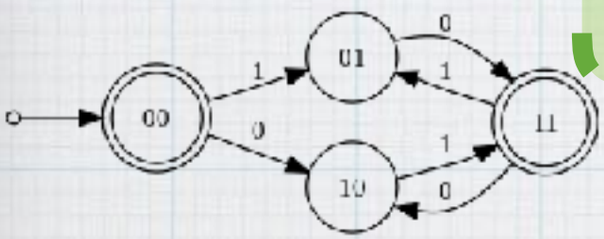
Analysis

Formal Methods

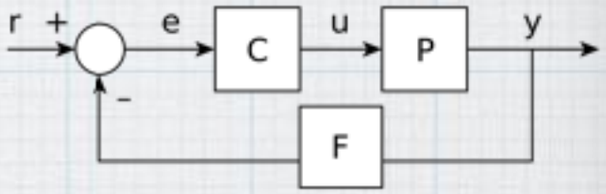
Control Theory

Collaboration

$\Box(p \Rightarrow \Diamond q)$



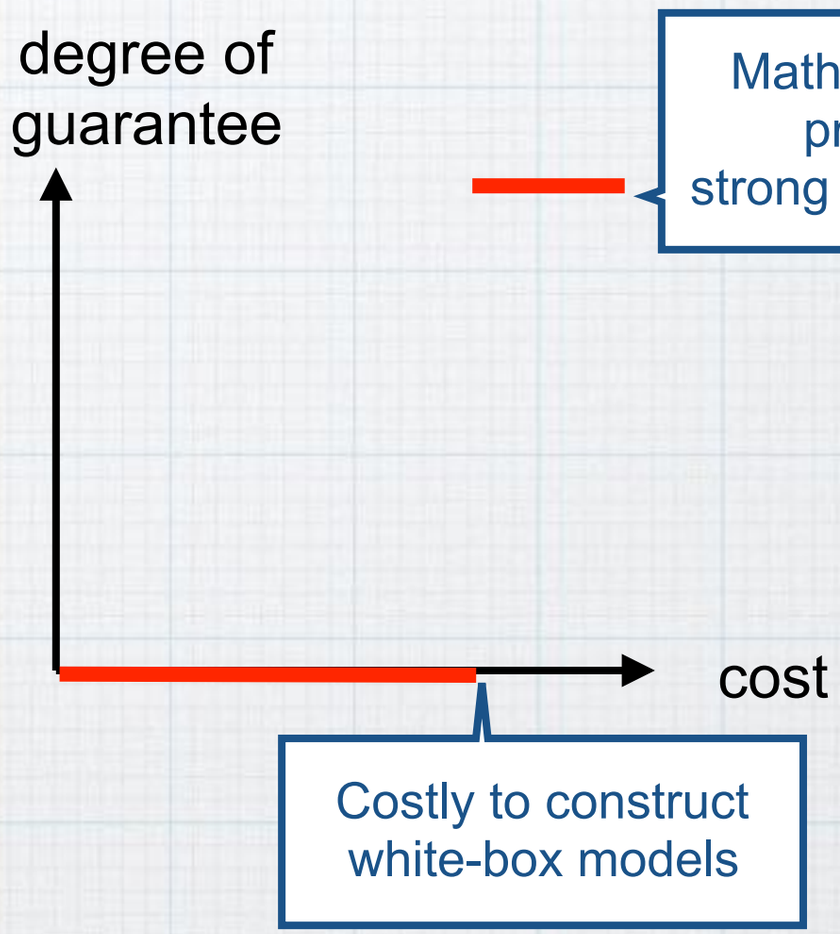
$x' = f(x, u)$



- Problem: scalability, esp. for real-world CPSs
  - Require complete understanding of a white-box model
  - Insist on being absolutely sound and correct
  - Little tolerance to uncertainty and noise
    - Big problem too in presence of statistical machine learning

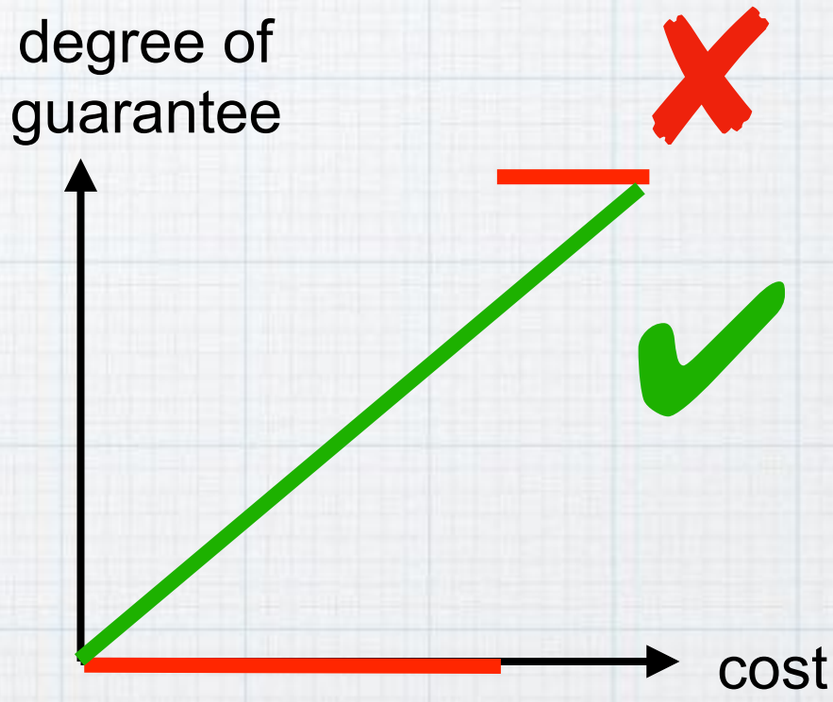
Towards Formal Methods That Are Down-Scalable

* Many FM methods (esp. for verification) are not down-scalable
  * They demand white-box models. Need formal models of all system components
  * They also require totally new design processes
  * ➜ huge cost before getting non-zero benefit (quality assurance)
* Example applications to CPS: airplanes (Airbus), space (NASA), …
  * Failure is so expensive
    ➜ worth verification effots
  * Still takes decades
  * Integrating continuous dynamics is hard
    ➜ They focus on software



Towards Formal Methods That Are Down-Scalable

* For real-world applications (esp. automotive domain), we need down-scalability
  * Even if we can only afford half the cost,
  * degree of guarantee does not become zero, but half



Formal Methods: Spectrum

Formal methods

testing | Model Checking | Automated Theorem Proving | Interactive Theorem Proving

low — Cost — high

Work with black-box models | Mandates white-box models

high — Automation — low

Automated | (Expert) Labor-intensive

low — Guarantees comprehensive? — high

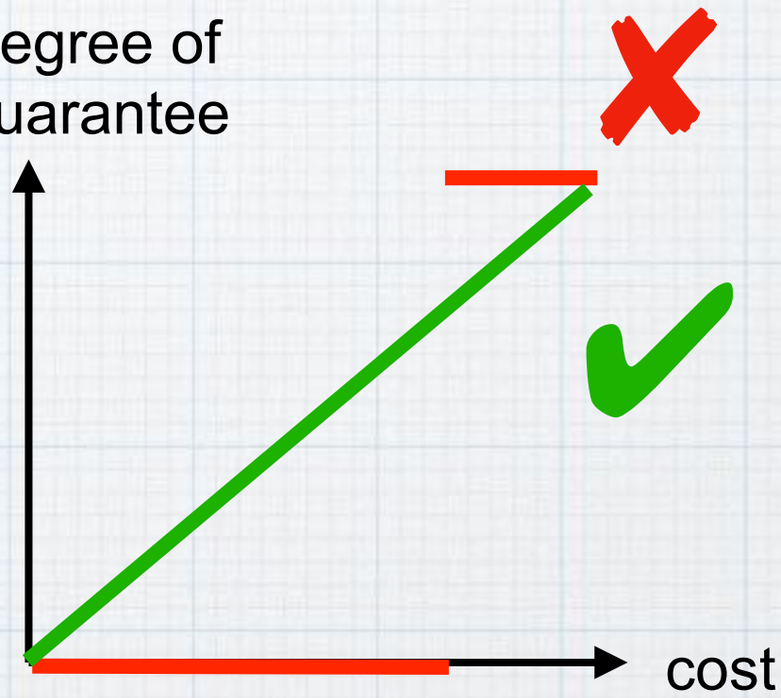
low — Guarantees mathematically rigorous? — high

empirical guarantees | mathematical proofs ➜ “formal verification”

Towards Formal Methods That Are Down-Scalable

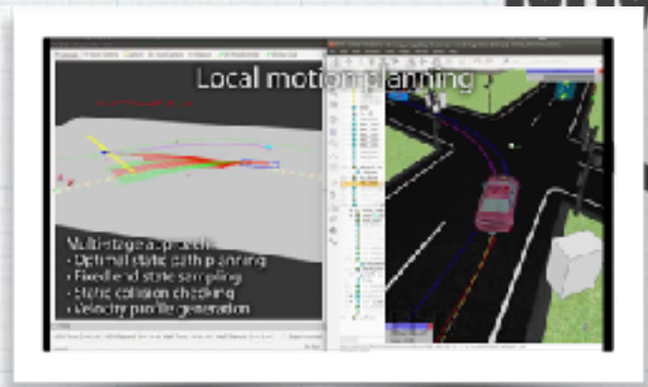
* Real-world applications (esp. automotive domain), need down-scalability
  * Even if we can only afford half the cost,
  * degree of guarantee does not become zero, but half

* Our efforts at ERATO MMSD
  * Combine testing and formal verification.
    * Search-based testing
    * monitoring
    * contract-based verification



ERATO 蓮尾メタ数理システムデザインプロジェクト
ERATO Metamathematics for Systems Design Project
国立情報学研究所 & 科学技術振興機構 National Institute of Informatics & Japan Science and Technology Agency

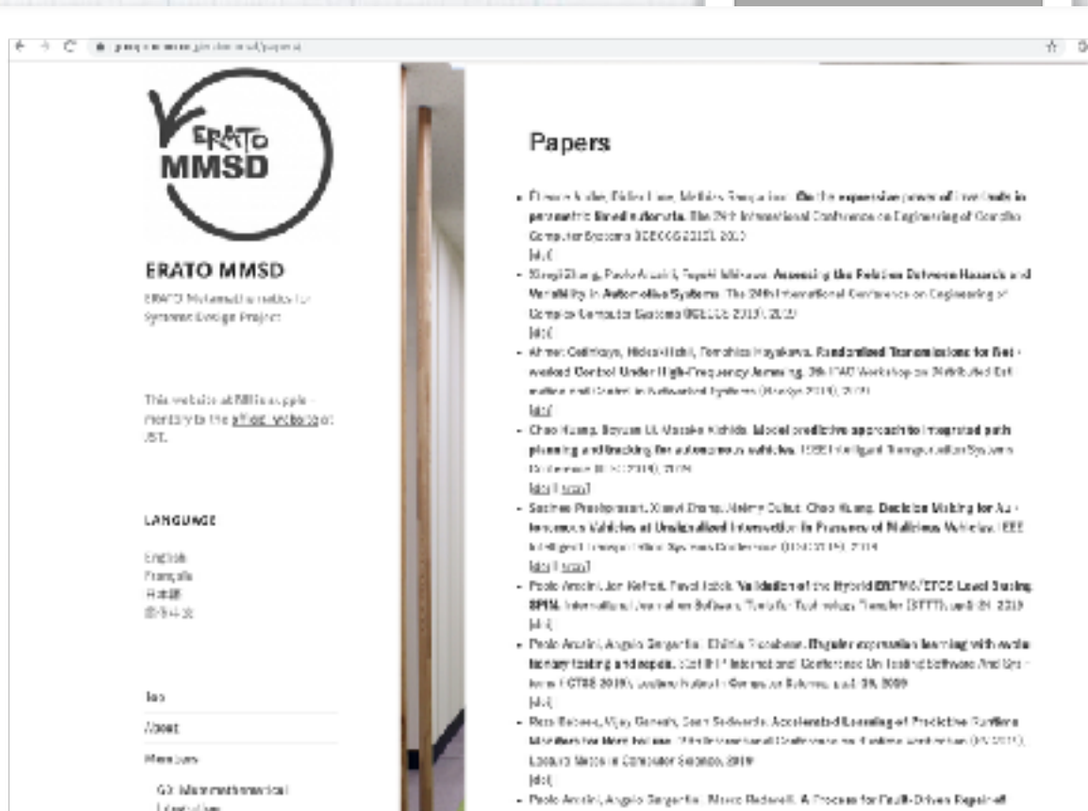
On ERATO MMSD



* **JST ERATO Project, 2016/10-2022/03**
  https://group-mmm.org/eratommsd

* **Our goal:**
  formal methods for cyber-physical systems (CPS)
  * **Extend formal methods, from software to CPS**
  * **Safety, reliability, V&V (Verification & Validation).**
    "Check if a system behaves as expected"
  * **Emphasizing industry collaboration, also for scientific inspirations**
  * **Automated driving as a strategic target domain.**
    Collaboration with U Waterloo:
    www.autonomoose.net

* **Our team:**
  * **28 researchers, > 20 students (as of 2019/09)**
  * **International and scientifically diverse**



Our Organization

International and multi-disciplinary. "creative chaos"

ERATO MMSD

Kyoto U IS Site:
Advanced Deductive Verification
Leader:
Kohei Suenaga

Kyoto U RIMS Site:
Categorical Infrastructure
Leader:
Masahito Hasegawa

Kyushu U Site:
Optimization for CPS V&V
Leader:
Hayato Waki

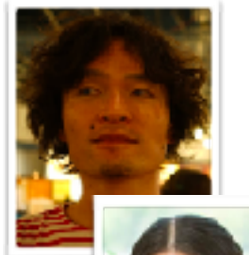
Osaka U Site:
Control Theory for CPS
Leader:
Toshimitsu Ushio

Group 0 @ NII:
Metatheoretical Integration
Leader: Shin-ya Katsumata

Topics:
Programming Languages, Formal Semantics, Categorical Models, Mathematical Logic, …

Group 3 @ NII:
Formal Methods and Intelligence
Leader: Fuyuki Ishikawa
Subleader: Paolo Arcaini
Topics:
Software Engineering, Formal Modeling, Testing, Safe & Explainable AI

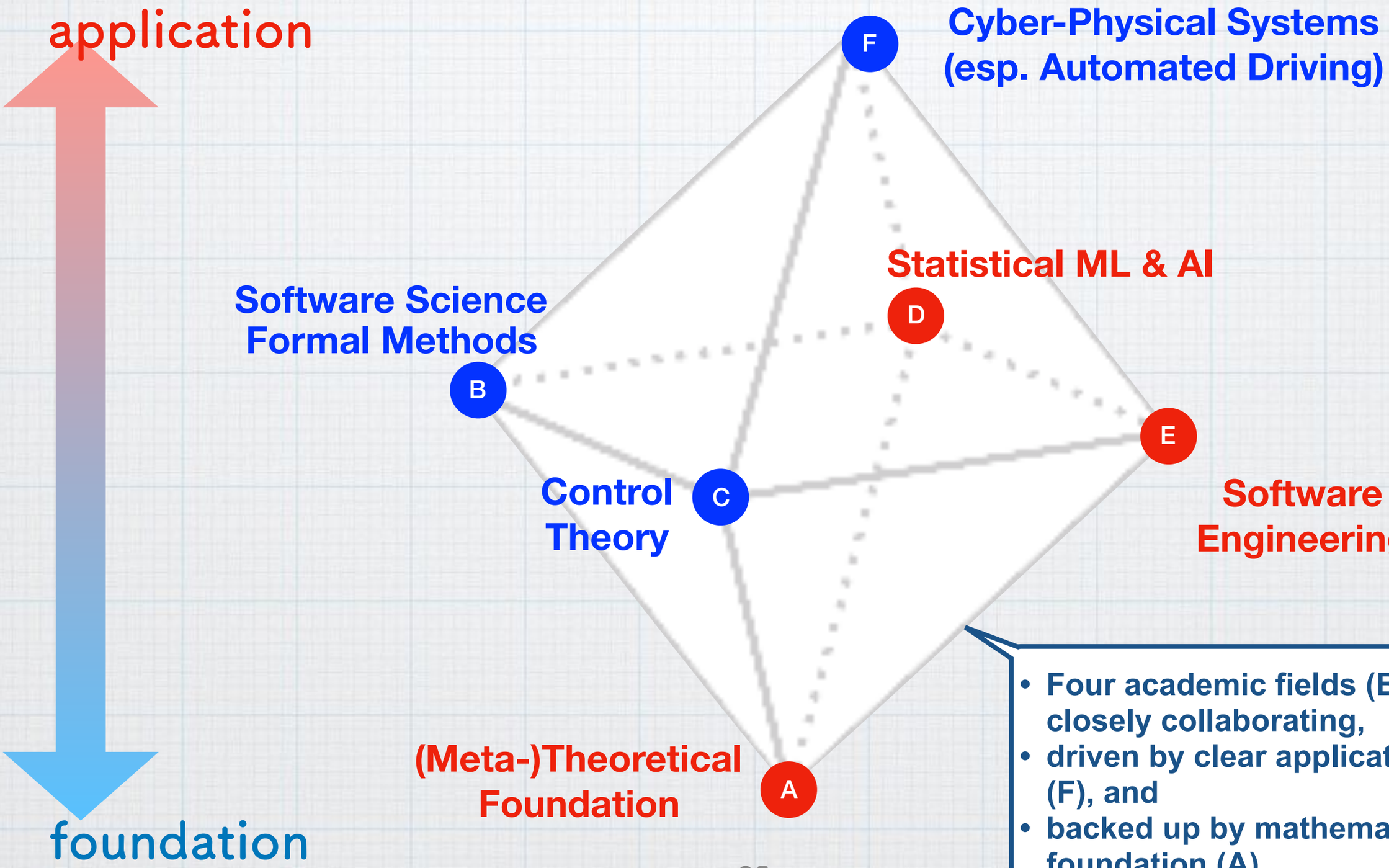
Group 1 @ NII:
Heterogeneous Formal Methods
Leader: Ichiro Hasuo
Subleader: Masako Kishida
Topics:
Automata Theory, Control Theory, Formal Verification, Proof Assistants, Automated Deduction, Runtime Verification



Group 2 @ U Waterloo:
Formal Methods in Industry
Leader: Krzysztof Czarnecki

Topics:
Automated Driving, Software Engineering, Machine Learning

Interdisciplinary Efforts towards CPS: Six Scientific Fields



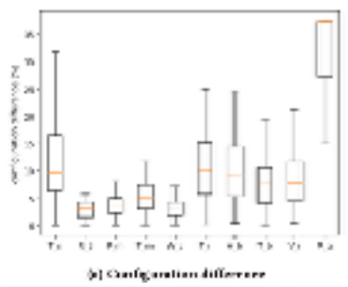
- Four academic fields (B,C,D,E) closely collaborating,
- driven by clear application goals (F), and
- backed up by mathematical foundation (A)

Our Techniques in a Spectrum

testing

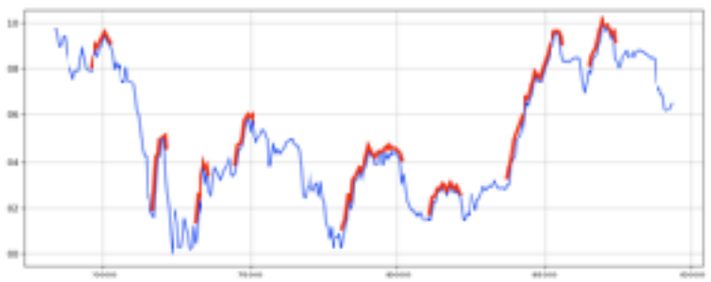
verification

Fault localization, stability analysis
- "Which part of the system is responsible for this failure?"
- [Zhang+, ICECCS'19] [Lee+, GECCO'19]

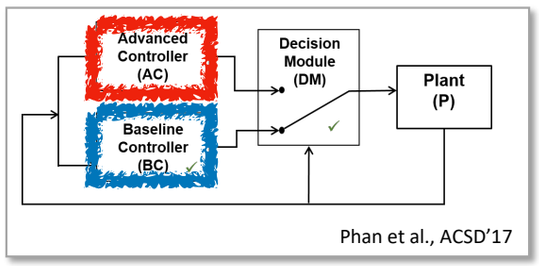


Search-based testing
(aka hybrid system falsification)
- Active search for counterexamples
- Incompletely but efficiently, by **reinforcement learning**
- [Zhang+, CAV'19] [Zhang+, EMSOFT'18]

Monitoring, runtime verification
- "Warn me when something is wrong"
- "Extract interesting parts out of this 10 GB log"
- Pattern matching via **timed automata**
- [Waga+, CAV'19] [Waga+, EMSOFT'19]



Contract-based verification, safety architecture
- Formal verification in presence of black-box models
- Failover strategy + switching, monitoring
- Incremental modeling by Event-B, fuzzy logic, …

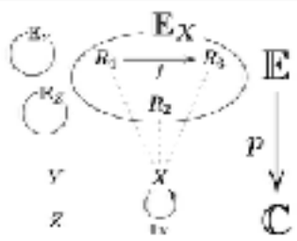


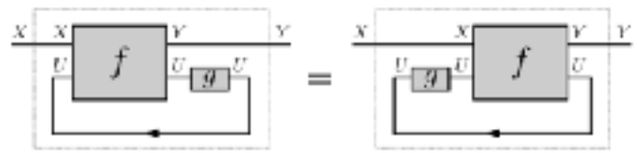
Phan et al., ACSD'17

Quantitative verification
- "Termination probability for this probabilistic program?" "How long will it take?"
- Automated analysis by **martingale synthesis**
- [Takisaka+, ATVA'18] [Kura+, TACAS'19]

$$V(x) = \max_{a \in \Gamma(x)} \{F(x,a) + \beta V(T(x,a))\}$$

Categorical foundations
- "Uniform framework for bisimilarity and bisimulation distance?"
- "Structural justification of backpropagation for RNN?"
- [Komorida+, LICS'19] [Sprunger+, LICS'19] …





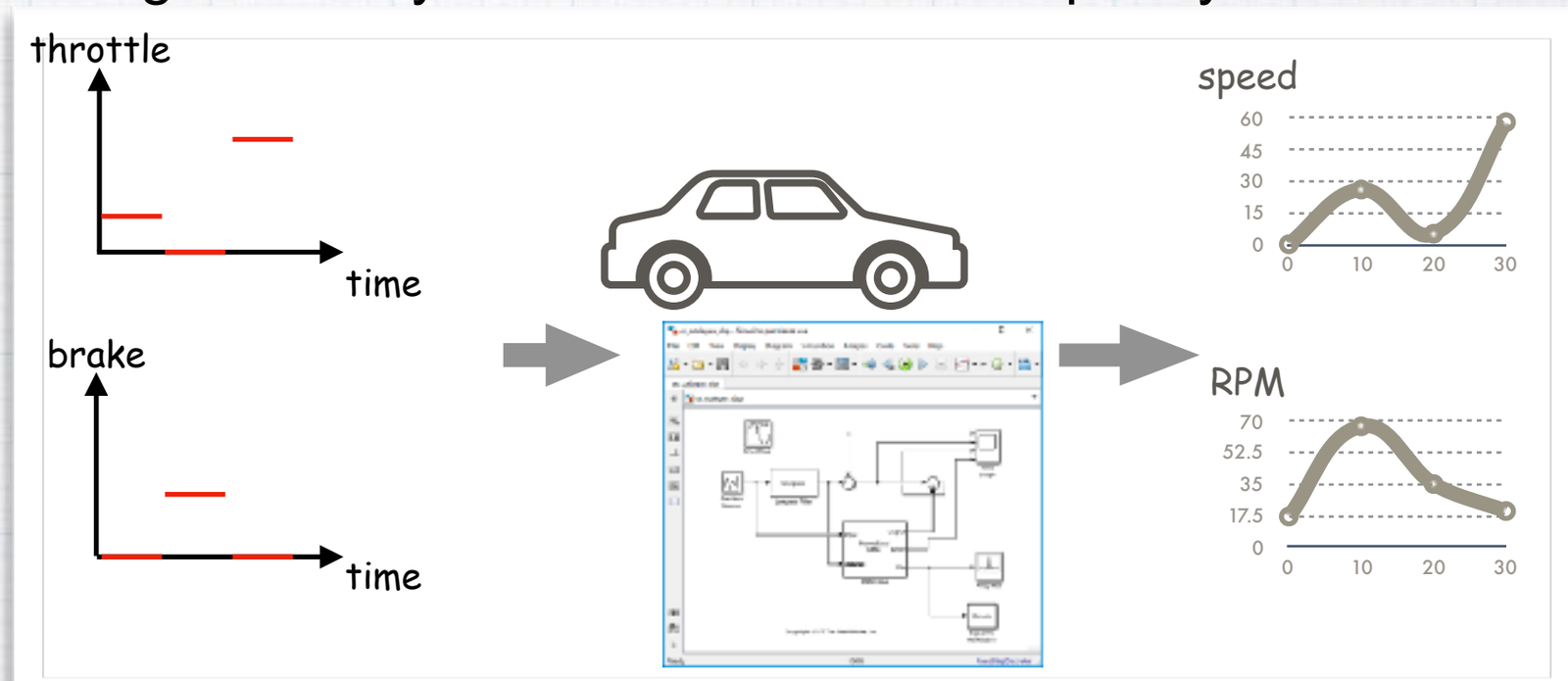
Outline

* Cyber-physical systems under uncertainties
* Formal methods and testing for CPS
  * Hybrid system falsification
    * Logical connectives [CAV'19]
    * Causality in time [EMSOFT'18]
    * … as demonstration of combining logical and statistical
  * Other topics

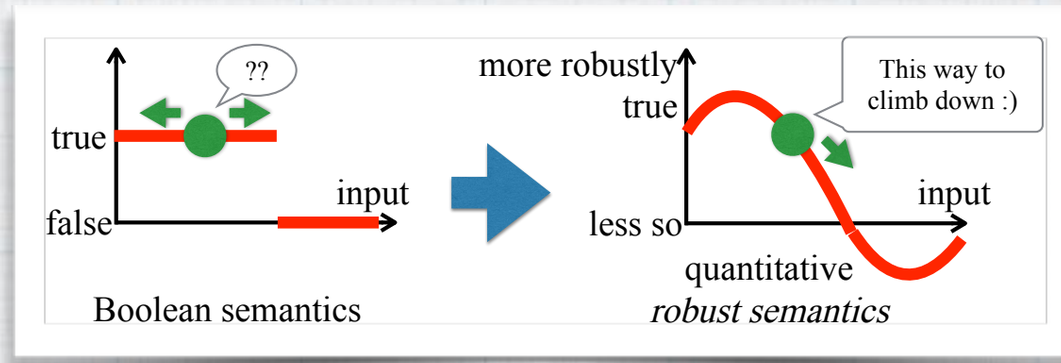
Hybrid System Falsification

[Fainekos & Pappas, TCS'09]

* Black-box, search-based testing. Actively search for erroneous input by:
  * Try an input signal
  * Observe the system's behavior
  * Choose the next input that is likely to be erroneous



* [Fainekos & Pappas, TCS'09]
  A reinforcement learning problem, by moving
  * from the Boolean semantics (erroneous or not)
  * to quantitative "robust semantics" (how far from being erroneous)



[Zhang, Ernst, Sedwards, Arcaini & Hasuo, EMSOFT'18]
[Zhang, Hasuo & Arcaini, CAV'19]

Discrete Structures in Continuous Learning & Optimization

- In hybrid system falsification (and statistical machine learning in general), efficiency comes mainly from **gradient descent** (which is continuous)
- **Question:** can we also exploit **discrete structures** for efficiency and explainability?

Search-based testing
(aka hybrid system falsification)
- Active search for counterexamples
- Incompletely but efficiently, by **reinforcement learning**
- [Zhang+, CAV'19] [Zhang+, EMSOFT'18]

- Example: hierarchical optimization scheme [Zhang+, CAV'19]
  - Given: a black-box model $\boldsymbol{M}$ and a specification $\Box_I(\varphi_1 \vee \varphi_2)$
  - Goal: input $\boldsymbol{i}$ s.t. $\boldsymbol{M}(\boldsymbol{i})$ violates the spec

"made this much progress for φ_i"

Feed search results back

Discrete Optim.

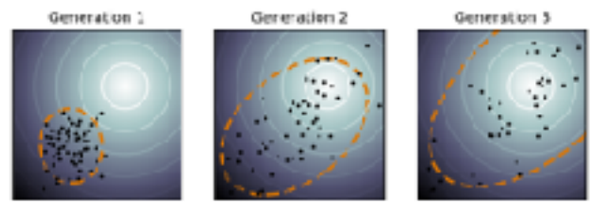
Restrict search domain

Continuous Optim.

Multi-armed bandit
over subformulas φ_1 and φ_2

Which formula (φ_1 or φ_2) to try to falsify

Stochastic gradient descent (such as CMA-ES) for input that violates φ_i



Multi-armed Bandits for Boolean Connectives in Hybrid System Falsification

Zhenya Zhang, Ichiro Hasuo, Paolo Arcaini
National Institute of Informatics, Tokyo, Japan
The Graduate University for Advanced Studies (SOKENDAI), Hayama, Japan

31st International Conference on Computer-Aided Verification
July 16, 2019

Falsification

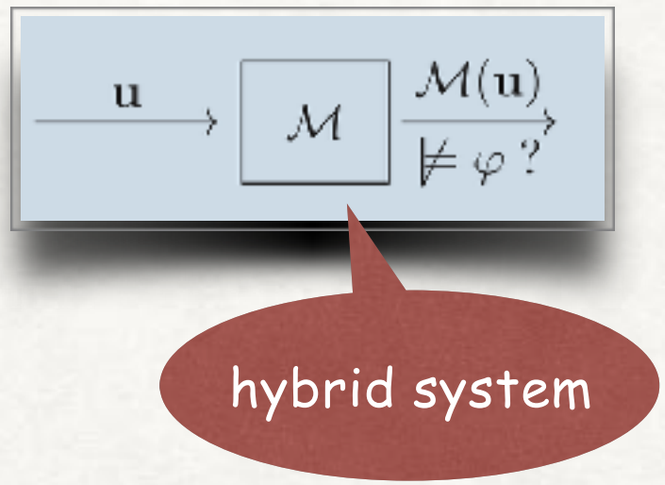
- Falsification problem:

$\mathcal{M}$: black-box model

$\mathbf{u}$: input signal (piecewise constant)

$\mathcal{M}(\mathbf{u})$: output signal

φ : specification in Temporal Logic

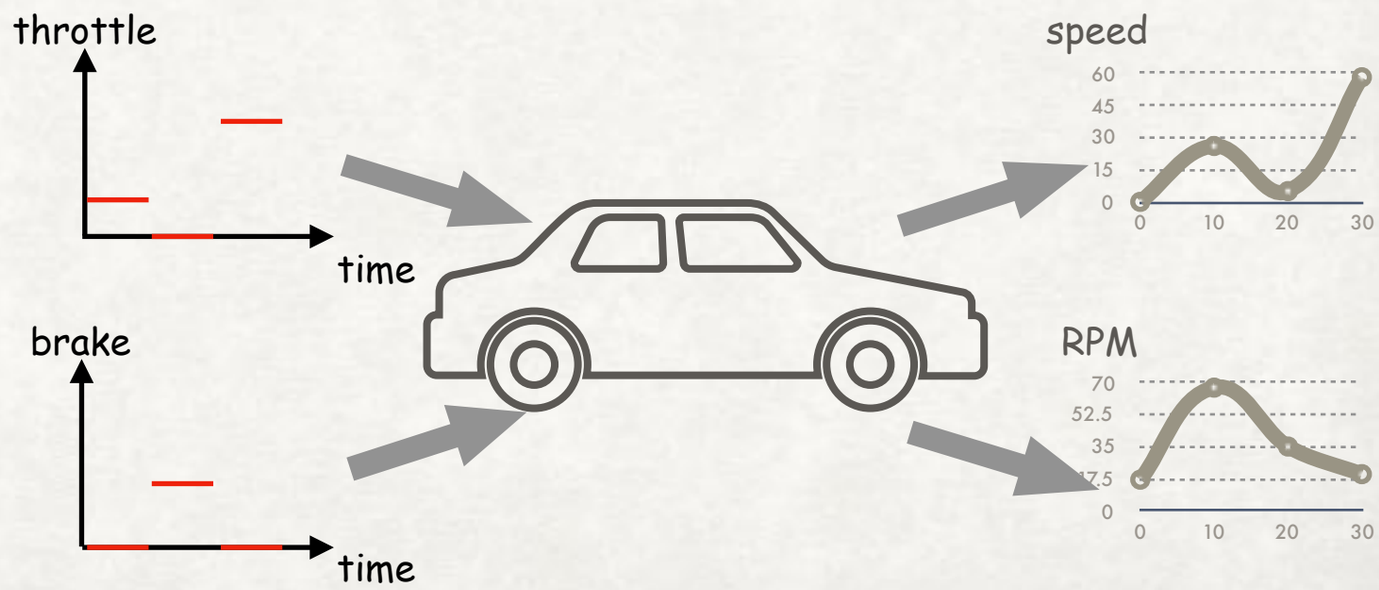


- Goal: find an input signal $\mathbf{u}$ s.t. $\mathcal{M}(\mathbf{u}) \not\models \varphi$

Example:

- $\Box_{[0,30]}(speed < 120)$

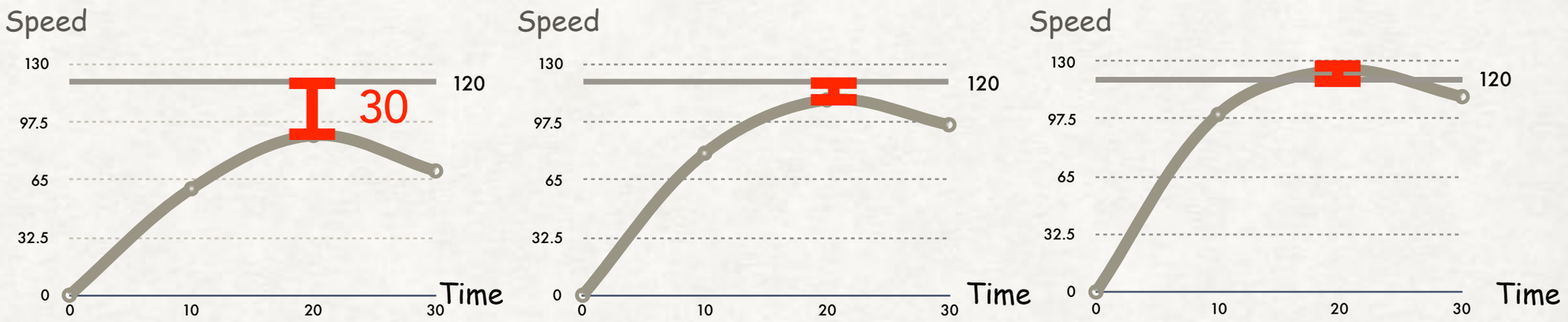
Always | Bounded-time | Proposition



STL robustness

- Quantitative robustness semantics of STL [Donze, Maler FORMATS'10]

e.g., $\varphi = \Box_{[0,30]}(speed < 120)$

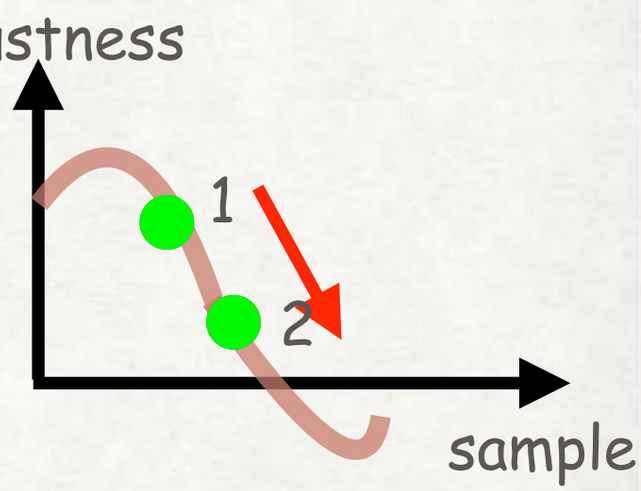


Output Signals			
Boolean Satisfaction	True	True	False
Quantitative Robustness	30	10	-5

Boolean → Real number

Hill-climbing optimization

- falsification => optimization
  - Objective function: $[\![\mathcal{M}(\mathbf{u}), \varphi]\!]$ (robustness)
  - Goal: min $[\![\mathcal{M}(\mathbf{u}), \varphi]\!]$
  - $[\![\mathcal{M}(\mathbf{u}), \varphi]\!] < 0$ → done!



- "Hill-climbing" optimization algorithms
  - Decide the next sample based on history
  - More intelligent than random sampling
  - Tools: Breach [Donze CAV'10] (CMA-ES, NM, etc.)

    S-TaLiRo [Annapureddy et al. TACAS'11] (SA etc.)

Motivating example

- Scale problem
  - e.g., $\phi \equiv \Box_{[0,30]}(gear = 3 \rightarrow speed > 20)$

- $\phi \equiv \Box_I(\varphi_1 \vee \varphi_2)$ where $\varphi_1 \equiv \neg(gear = 3)$ and $\varphi_2 \equiv (speed > 20)$

- STL robustness semantics [Donze, Maler FORMATS'10]

$$[\![\mathbf{w}, f(x_1, \cdots, x_n) > 0]\!] := f\big(\mathbf{w}(0)(x_1), \cdots, \mathbf{w}(0)(x_n)\big)$$
$$[\![\mathbf{w}, \bot]\!] := -\infty \qquad [\![\mathbf{w}, \neg\varphi]\!] := -[\![\mathbf{w}, \varphi]\!]$$
$$[\![\mathbf{w}, \varphi_1 \wedge \varphi_2]\!] := [\![\mathbf{w}, \varphi_1]\!] \sqcap [\![\mathbf{w}, \varphi_2]\!] \qquad [\![\mathbf{w}, \varphi_1 \vee \varphi_2]\!] := [\![\mathbf{w}, \varphi_1]\!] \sqcup [\![\mathbf{w}, \varphi_2]\!]$$
$$[\![\mathbf{w}, \varphi_1 \,\mathcal{U}_I\, \varphi_2]\!] := \bigsqcup_{t \in I \cap [0,T]} \big([\![\mathbf{w}^t, \varphi_2]\!] \sqcap \bigsqcap_{t' \in [0,t)} [\![\mathbf{w}^{t'}, \varphi_1]\!] \big)$$

$gear \in \{1, 2, 3, 4\}$

$speed \in [0, 150]$

$[\![\mathcal{M}(\mathbf{u}), \varphi_1]\!] \overset{?}{\sqcup} [\![\mathcal{M}(\mathbf{u}), \varphi_2]\!]$

Motivating example

$$\phi \equiv \square_{[0,30]}(gear = 3 \rightarrow speed > 20)$$

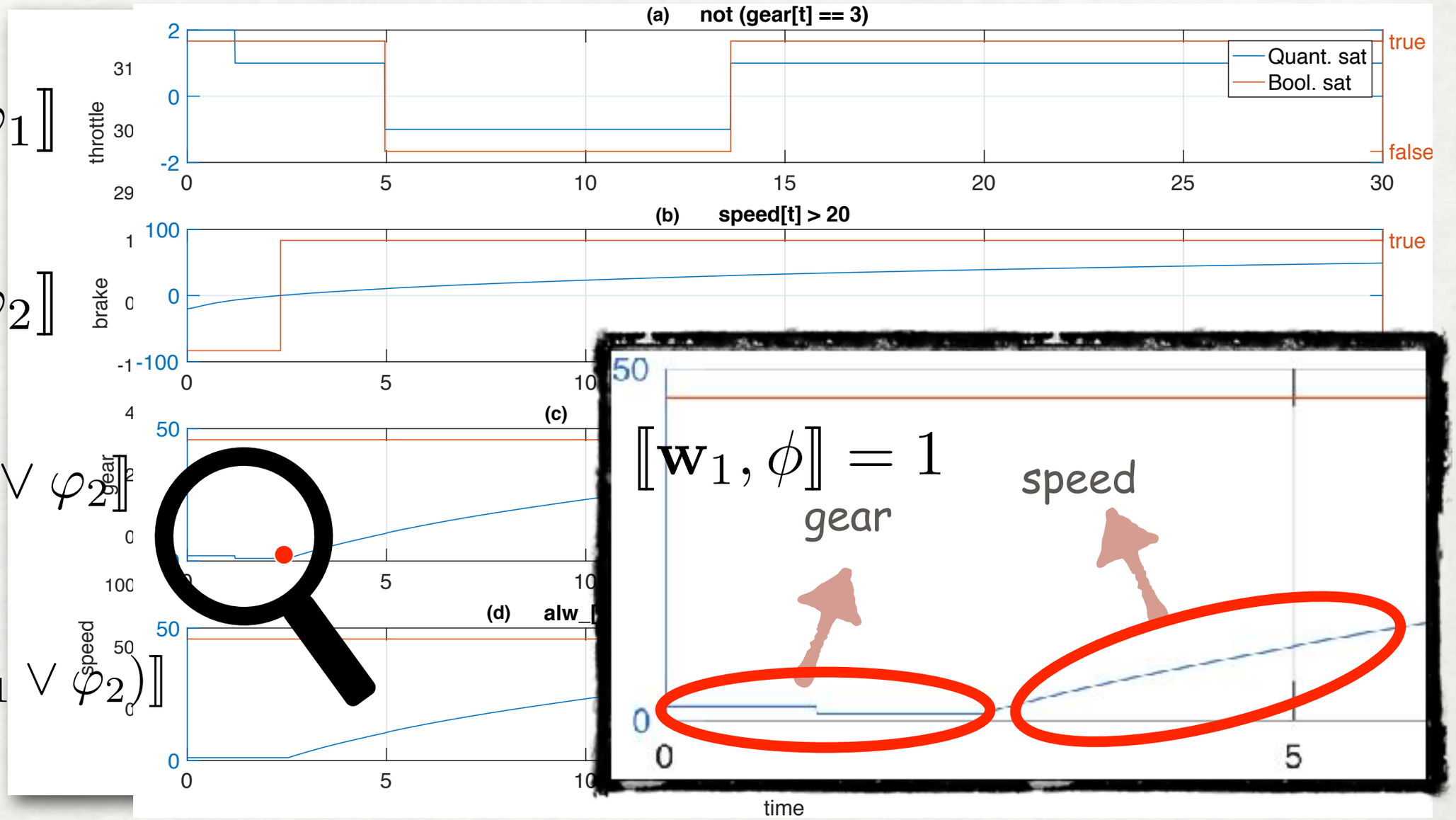
$$\phi \equiv \square_I(\varphi_1 \vee \varphi_2) \qquad \varphi_1 \equiv \neg(gear = 3) \qquad \varphi_2 \equiv (speed > 20)$$

$[\![\mathcal{M}(\mathbf{u}), \varphi_1]\!]$

$[\![\mathcal{M}(\mathbf{u}), \varphi_2]\!]$

$[\![\mathcal{M}(\mathbf{u}), \varphi_1 \vee \varphi_2]\!]$

$[\![\mathcal{M}(\mathbf{u}), \square_I(\varphi_1 \vee \varphi_2)]\!]$



Motivating example

$$\phi \equiv \Box_{[0,30]}(gear = 3 \rightarrow speed > 20)$$

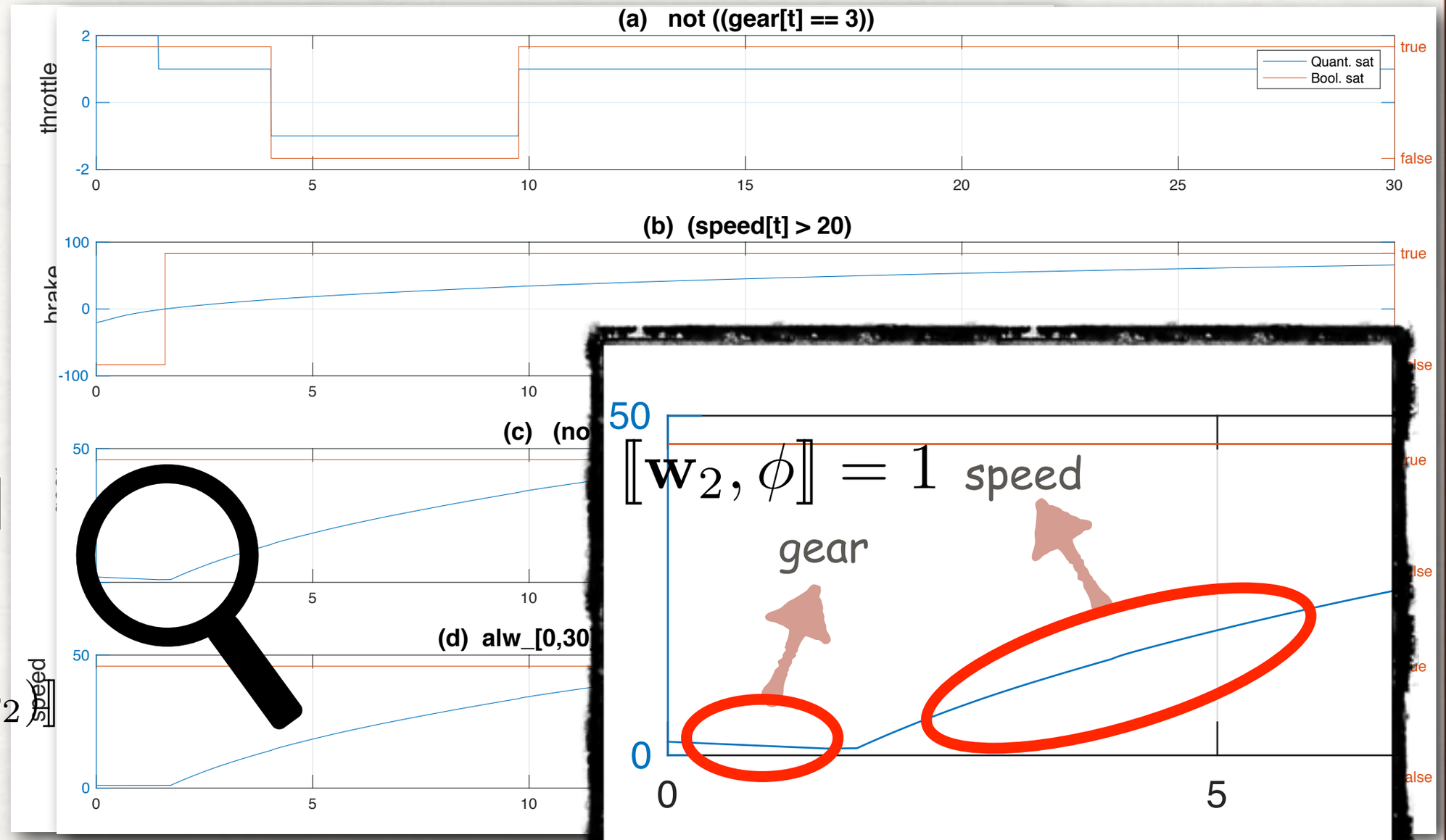
$$\phi \equiv \Box_I(\varphi_1 \vee \varphi_2) \qquad \varphi_1 \equiv \neg(gear = 3) \qquad \varphi_2 \equiv (speed > 20)$$

$[\![\mathcal{M}(\mathbf{u}), \varphi_1]\!]$

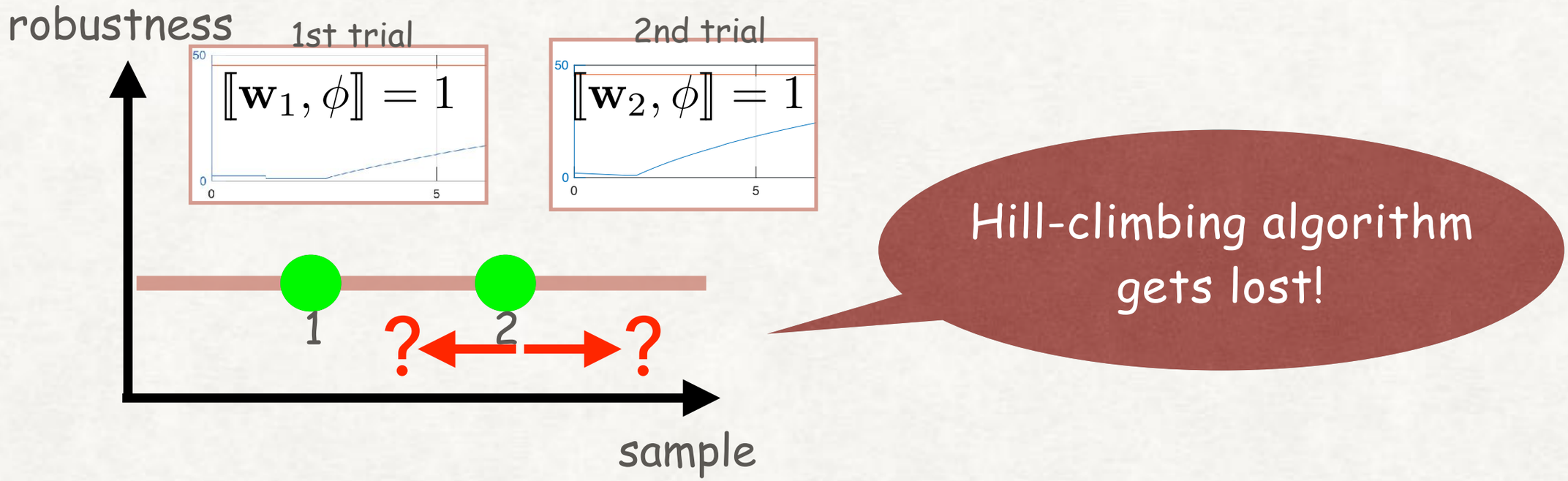
$[\![\mathcal{M}(\mathbf{u}), \varphi_2]\!]$

$[\![\mathcal{M}(\mathbf{u}), \varphi_1 \vee \varphi_2]\!]$

$[\![\mathcal{M}(\mathbf{u}), \Box_I(\varphi_1 \vee \varphi_2)]\!]$



Problem



- Falsification of safety properties with boolean connectives
  - $\square_I(\varphi_1 \wedge \varphi_2)$, conjunctive
  - $\square_I(\varphi_1 \vee \varphi_2)$, disjunctive

Multi-armed bandit

- Multi-armed Bandit problem
  - Gambler vs a row of slot machines
  - Goal: maximize the reward
  - exploitation: focus on a "good" one
  - exploration: also consider others
- Algorithms based on "Reward"
  - n machines, k is index of rounds
  - ϵ-greedy: relies on a scalar ϵ

$$
\begin{aligned}
&j_{\text{emp-opt}} \leftarrow \arg\max_{j \in [1,n]} R(j, k-1) \\
&\text{Sample } i_k \in [1, n] \text{ from the distribution} \\
&\quad \left[\begin{array}{rl} j_{\text{emp-opt}} & \longmapsto (1-\varepsilon) + \frac{\varepsilon}{n} \\ j & \longmapsto \frac{\varepsilon}{n} \end{array} \right. \quad \text{for each } j \in [1, n] \setminus \{j_{\text{emp opt}}\}
\end{aligned}
$$

High probability

Low probability

  - UCB1: balance of exploration and exploitation

$$
i_k \leftarrow \arg\max_{j \in [1,n]} \left(R(j, k-1) + c\sqrt{\frac{2\ln(k-1)}{N(j,k-1)}} \right)
$$

The number of times visiting j

Case 1. $\square_I(\varphi_1 \wedge \varphi_2)$

- Intuition: it suffices that either $[\![\mathcal{M}(\mathbf{u}), \varphi_1]\!] < 0$ or $[\![\mathcal{M}(\mathbf{u}), \varphi_2]\!] < 0$

- Run hill-climbing optimization on two machines
- Schedule them with MAB solvers
  - ϵ-greedy, UCB1, etc.
- Reward: hill-climbing gain

k: rounds played

$$\mathcal{R}(\varphi_i) = \begin{cases} \dfrac{\text{max-rb}(i, k-1) - \text{last-rb}(i, k-1)}{\text{max-rb}(i, k-1)} & \text{if } \varphi_i \text{ has been played before} \\ 0 & \text{otherwise} \end{cases}$$

Intuition: the more robustness decrease, the more promising it is

Case 2. $\Box_I(\varphi_1 \vee \varphi_2)$

- Not as straightforward as Case 1
- Assumption: it is trivial to falsify only φ_1 or φ_2
- Robustness restricted to S : $[\![\mathcal{M}(\mathbf{u}), \varphi]\!]_S \quad S \subseteq [0, T]$
- Key insight:

  Let $S = \{t \subseteq [0,T] | [\![\mathcal{M}(\mathbf{u})^t, \varphi_1]\!] < 0\}$

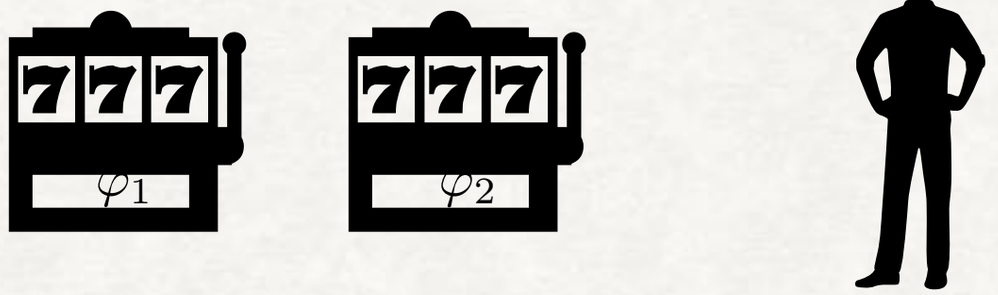
  $[\![\mathcal{M}(\mathbf{u}), \varphi_2]\!]_S < 0$ implies $[\![\mathcal{M}(\mathbf{u}), \Box_I(\varphi_1 \vee \varphi_2)]\!] < 0$

- Intuition: it suffices that

  either $[\![\mathcal{M}(\mathbf{u}), \varphi_1]\!]_{S_1} < 0$

  or $[\![\mathcal{M}(\mathbf{u}), \varphi_2]\!]_{S_2} < 0$

  $S_i = \{t \in [0,T] | [\![\mathcal{M}(\mathbf{u}^t), \varphi_{\bar{i}}]\!] < 0\}$



when $i = 1, \bar{i} = 2$

when $i = 2, \bar{i} = 1$

Experimental evaluation

- Benchmark set 1: efficacy and efficiency
- Simulink models & specifications:
  - Automatic transmission [Hoxha et al. ARCH'15]
  - Abstract fuel control [Jin et al. HSCC'14]
  - NARMA-L2 neurocontroller [Beale et al. '92]
- Algorithms:
  - Breach (with CMA-ES)
  - MAB-ϵ-greedy
  - MAB-UCB
- Misc:
  - 30 trials for each
  - Timeout: 600s
  - HC: CMA-ES

General boolean connectives, not necessarily with scale problem

(a) Bbench (here $\delta_{t'}(\mathbf{w})$ represents $\mathbf{w}'(t') - \mathbf{w}'(0)$).

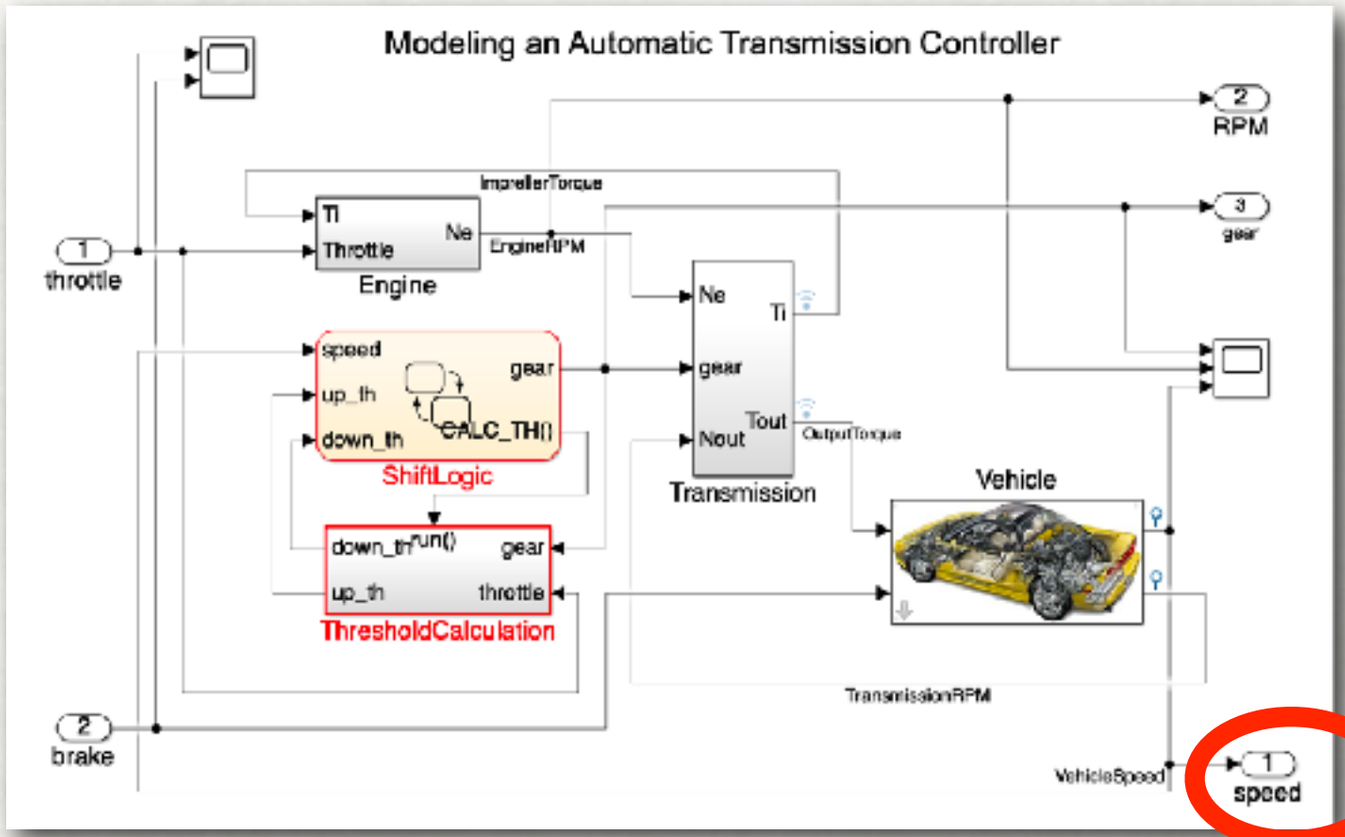
Bench	Specification ID	Specification Formula	Parameter
AT	AT1	$\square_{[0,30]}((gear = 3) \rightarrow (speed > \rho))$	$\rho \in \{20.6, 20.4, 20.2, 20, 19.8\}$
	AT2	$\square_{[0,30]}((gear = 4) \rightarrow (speed > \rho))$	$\rho \in \{43, 41, 39, 37, 35\}$
	AT3	$\square_{[0,30]}((gear = 4) \rightarrow (rpm > \rho))$	$\rho \in \{700, 800, 900, 1000, 1100\}$
	AT4	$\square_{[0,30-\tau]}((\delta_{10}(rpm) > 2000) \rightarrow (\delta_{\tau}(gear) > 0))$	$\tau \in \{15, 16, 17, 18, 19\}$
	AT5	$\square_{[0,30]}((speed < \rho) \wedge (RPM < 4780))$	$\rho \in \{130, 131, 132, 133, 134, 135, 136, 137\}$
	AT6	$\square_{[0,26]}((\delta_4(speed) > \rho) \rightarrow (\delta_4(gear) > 0))$	$\rho \in \{20, 25, 30, 35, 40\}$
	AT7	$\square_{[0,30-\tau]}((\delta_{\tau}(speed) > 30) \rightarrow (\delta_{\tau}(gear) > 0))$	$\tau \in \{2, 3, 4, 5, 6, 7, 8\}$
AFC	AFC1	$\square_{[11,50]}((controller\_mode = 0) \rightarrow (mu < \rho))$	$\rho \in \{0.16, 0.17, 0.18, 0.19, 0.2\}$
	AFC2	$\square_{[11,50]}((controller\_mode = 1) \rightarrow (mu < \rho))$	$\rho \in \{0.222, 0.224, 0.226, 0.228, 0.23\}$
NN		$close \equiv \lvert Pos - Ref \rvert <= \rho + \alpha * \lvert Ref \rvert$ $reach \equiv \Diamond_{[0,2]}(\square_{[0,1]}(close))$	
	NN1	$\square_{[0,18]}(\neg close \rightarrow reach), \alpha = 0.04$	$\rho \in \{0.001, 0.002, 0.003, 0.004, 0.005\}$
	NN1	$\square_{[0,18]}(\neg close \rightarrow reach), \alpha = 0.03$	$\rho \in \{0.001, 0.002, 0.003, 0.004, 0.005\}$

Spec. ID	Breach SR (/30) success rate			Breach (sec.) time			MAB-ε-greedy SR (/30) success rate			Δ	MAB-ε-greedy (sec.) time			Δ	MAB-UCB SR (/30) success rate			Δ	MAB-UCB (sec.) time			Δ
				Mi		Avg					Mi		Avg						Mi		Avg	
AT1	14	25	20.2	125	361.2	223.1	24	30	28.6	35.7	62.7	213.4	106.4	−73.4	28	30	29.2	37.8	45.1	146.8	77.4	−97.1
AT2	11	30	20.2	14	390.6	209.8	30	30	30	43.9	11.9	126.3	54.5	−96.9	27	30	29.4	42.2	17.7	92.5	36.8	−112.1
AT3	29	30	29.4	2.3	22.2	14.2	30	30	30	2	2.5	7	3.5	−82.9	30	30	30	2	2.5	3.6	3	−88.6
AT4	18	30	25.8	19.5	265.3	109.6	29	30	29.8	16	7.8	45.1	24.4	−105	30	30	30	16.6	6.2	36.2	22.2	−113.5
AT5	6	23	14.1	203.1	525.9	366.2	26	30	28.5	72.1	35.2	149	93.7	−120.6	26	30	28.2	71.4	37.7	154.1	99.2	−116.8
AT6	5	29	22.8	30.1	509.5	157	21	30	27	28	2.3	300	95.1	−98.3	22	30	27	27.7	2.9	247.3	86.1	−99.4
AT7	15	30	26.6	12.2	314	81.5	20	30	28.6	8.4	2.9	283.9	49.9	−92	23	30	29	10.3	5.5	223.3	42.9	−88.3
AFC1	6	30	14.4	124.8	565.6	413.5	4	28	12	−28.4	171	568.4	446	10.8	5	30	16.4	9.7	98.7	559.8	389.9	−9.3
AFC2	2	30	18	80.7	582.3	343.4	5	30	20	23.8	43.2	547.8	301.9	−23.8	5	30	20	22.9	59.4	568.4	320.5	−11.1
NN1	17	25	20.8	212.9	384.7	292.9	14	27	20.2	−4.5	189.5	422.8	320.3	6.2	17	28	22.6	7.3	148.2	403.3	272.3	−11.8
NN2	27	28	27.2	55.5	93.4	73.1	30	30	30	9.8	11	39.3	26.3	−97.8	30	30	30	9.8	14.6	38.2	27.4	−92.3
$AT1^{-2}$	30	30	30	42.5	97.4	56.9	28	30	29	−3.4	75.6	178.3	118.7	68.7	28	30	29.4	−2.1	54.3	136.3	80.3	33.3
$AT1^{0}$	14	25	20.2	125	361.2	223.1	24	30	28.6	35.7	62.7	213.4	106.4	−73.4	28	30	29.2	37.8	45.1	146.8	77.4	−97.1
$AT1^{1}$	4	21	15.4	204.5	527.6	310.2	25	30	29	68.4	49	234.7	102.1	−108	27	29	28.2	64.5	77.5	128.7	105.1	−93
$AT1^{3}$	8	24	19.8	164	471.7	240.1	29	30	29.8	44.6	67.5	170.6	101.9	−77.3	29	30	29.4	43.4	55.4	104.8	80.6	−93.6
$AT5^{-2}$	29	30	29.6	61.1	163.7	102	25	30	27.8	−6.4	76.9	139.5	111.9	12.6	28	30	29.4	−0.7	48.5	131.9	85.7	−17
$AT5^{0}$	6	18	11.2	291.1	525.9	423.1	28	30	28.4	90.5	80.2	151.3	107.4	−117.7	26	30	28	89.4	68.3	154.1	114.9	−114.5
$AT5^{1}$	0	2	0.4	566.4	600	593.3	27	30	28.4	194.8	70.7	184.5	110.3	−138.5	25	30	27.6	194.1	83.1	150	123.7	−131.2
$AT5^{3}$	0	1	0.2	586.4	600	597.3	27	30	28.6	197.2	66.8	163.3	102.5	−142.3	27	29	28	197.2	80.4	160.9	111.9	−137.4
$AFC1^{0}$	6	30	14.4	124.8	565.6	413.5	4	29	16.4	8.5	115.1	559.9	411.1	−2.8	5	30	16.4	9.7	98.7	559.8	389.9	−9.3
$AFC1^{1}$	7	30	16.6	99	548.2	393.3	3	29	10.8	−60.9	198.1	587.6	465.8	24.6	7	29	17.8	10.3	105.7	527.3	354.3	−10.3
$AFC1^{2}$	0	12	5.2	434.4	600	535.8	3	28	11.6	96.2	180.8	577.6	463	−20.7	4	30	17	127	73.7	556.3	374.5	−47.3
$AFC1^{3}$	1	12	4.8	425.7	587.4	532.6	3	30	14.4	109	138	585.5	436.5	−28	7	30	15	113	77.1	553.4	403.7	−39.9

- RQ1: Which is the best MAB algorithm for our purpose?
  - MAB-UCB
- RQ2: How is the performance of the proposed process
  - Generally more effective and efficient compared to Breach
  - Works very well even on examples free of scale problem

Experimental evaluation

- Benchmark set 2: scale problem
- Experiment setting
- e.g., Automatic transmission [Hoxha, ARCH'15]



Spec ID	scaled output	factor 10^k
$AT1_1$		
$AT1_2$		
$AT1_3$	*speed*	$k \in \{-2,0,1,3\}$
$AT1_4$		
$AT1_5$		
$AT5_4$		
$AT5_5$		
$AT5_6$	*speed*	$k \in \{-2,0,1,3\}$
$AT5_7$		
$AT5_8$		
$AFC1_1$		
$AFC1_2$		
$AFC1_3$	*mu*	$k \in \{0,1,2,3\}$
$AFC1_4$		
$AFC1_5$		

e.g., when scaling speed and k = 3,

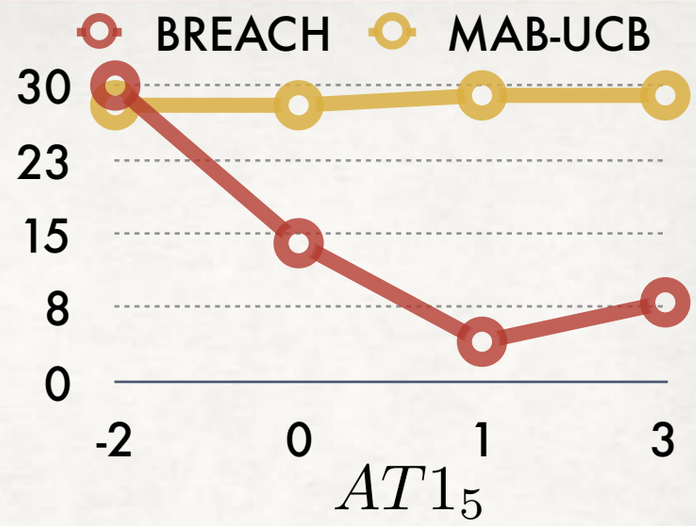
$$\square_{[0,30]}(gear = 3 \rightarrow speed > 20000)$$

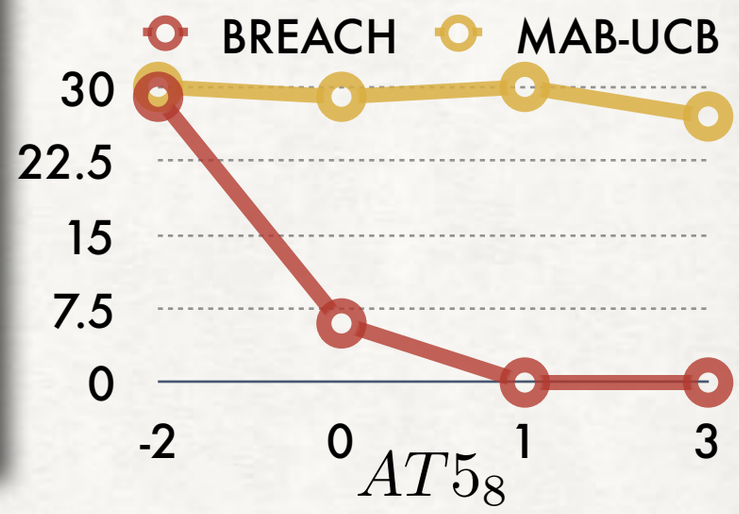
speed*10^k

Spec. ID	Breach SR (/30)	Breach time (sec.)	MAB-UCB SR (/30)	MAB-UCB time (sec.)
$AT1_1^{-2}$	30	51.3	30	54.3
$AT1_1^{0}$	25	125	29	75
$AT1_1^{1}$	20	221.1	28	107.9
$AT1_1^{3}$	23	170	29	55.4
$AT1_5^{-2}$	30	49	29	67.5
$AT1_5^{0}$	22	187.5	30	45.1
$AT1_5^{1}$	21	204.5	29	77.5
$AT1_5^{3}$	24	164	30	61
$AT1_6^{-2}$	30	42.5	30	62.4
$AT1_6^{0}$	19	239.5	29	62.5
$AT1_6^{1}$	16	296.2	27	128.7
$AT1_6^{3}$	21	209.8	30	93.4
$AT1_7^{-2}$	30	44.5	30	80.8
$AT1_7^{0}$	21	202.2	30	57.4
$AT1_7^{1}$	16	301.7	28	119.5
$AT1_7^{3}$	23	185.1	29	88.3
$AT1_8^{-2}$	30	97.4	28	136.3
$AT1_8^{0}$	14	361.2	28	146.8
$AT1_8^{1}$	4	527.6	29	91.9
$AT1_8^{3}$	8	471.7	29	104.8

Spec. ID	Breach SR (/30)	Breach time (sec.)	MAB-UCB SR (/30)	MAB-UCB time (sec.)
$AT5_4^{-2}$	30	61.1	30	48.5
$AT5_4^{0}$	18	291.1	28	94.5
$AT5_4^{1}$	2	566.4	25	150
$AT5_4^{3}$	1	586.4	28	96.2
$AT5_5^{-2}$	30	71.3	29	67.8
$AT5_5^{0}$	15	369.1	27	114
$AT5_5^{1}$	0	600	29	83.1
$AT5_5^{3}$	0	600	27	113.8
$AT5_6^{-2}$	29	110.2	28	103.3
$AT5_6^{0}$	10	438.2	30	68.3
$AT5_6^{1}$	0	600	27	126.7
$AT5_6^{3}$	0	600	29	80.4
$AT5_7^{-2}$	30	103.6	30	77.3
$AT5_7^{0}$	7	491.4	26	154.1
$AT5_7^{1}$	0	600	27	134.3
$AT5_7^{3}$	0	600	29	108
$AT5_8^{-2}$	29	163.7	30	131.9
$AT5_8^{0}$	6	525.9	29	143.6
$AT5_8^{1}$	0	600	30	124.2
$AT5_8^{3}$	0	600	27	160.9

Spec. ID	Breach SR (/30)	Breach time (sec.)	MAB-UCB SR (/30)	MAB-UCB time (sec.)
$AFC1_1^{0}$	30	124.8	30	98.7
$AFC1_1^{1}$	30	99	29	105.7
$AFC1_1^{2}$	12	434.4	30	73.7
$AFC1_1^{3}$	12	425.7	30	77.1
$AFC1_2^{0}$	16	421.5	23	346.8
$AFC1_2^{1}$	25	345.9	27	227.9
$AFC1_2^{2}$	8	497.2	25	320.5
$AFC1_2^{3}$	5	518.1	21	364
$AFC1_3^{0}$	11	457.7	15	442
$AFC1_3^{1}$	13	479.2	14	455.5
$AFC1_3^{2}$	2	590.7	15	453.2
$AFC1_3^{3}$	5	545.6	8	510.6
$AFC1_4^{0}$	9	498.2	9	502.1
$AFC1_4^{1}$	8	494	12	455
$AFC1_4^{2}$	4	556.8	11	468.7
$AFC1_4^{3}$	1	587.4	9	513.4
$AFC1_5^{0}$	6	565.6	5	559.8
$AFC1_5^{1}$	7	548.2	7	527.3
$AFC1_5^{2}$	0	600	4	556.3
$AFC1_5^{3}$	1	586	7	553.4





- RQ3: Does the proposed approach effectively solve the scale problem?
  - —— yes, ours is not subject to the change of the scale.
- RQ4: Is the proposed approach more effective than normalization?
  - ——yes, e.g., $AT5_5^1$ is normalized to the same scale, but Breach performs badly

Conclusion and future work

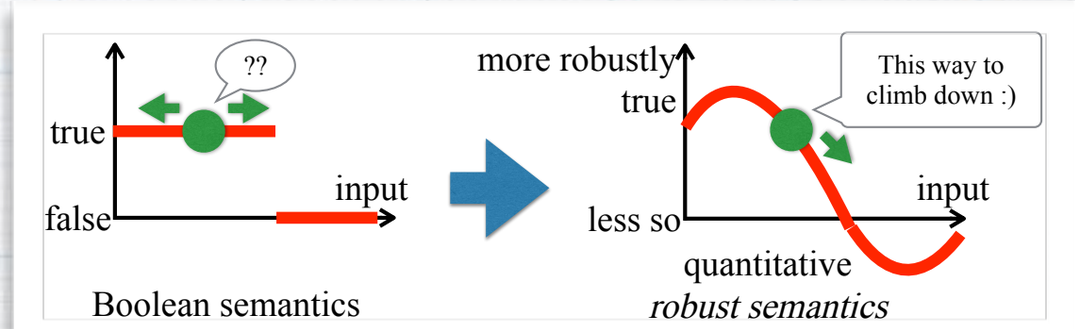
- Conclusion
  - We propose an approach for solving the scale problem
  - Multi-armed bandit solver + hill-climbing gain reward
  - Experiments show the effectiveness and efficiency

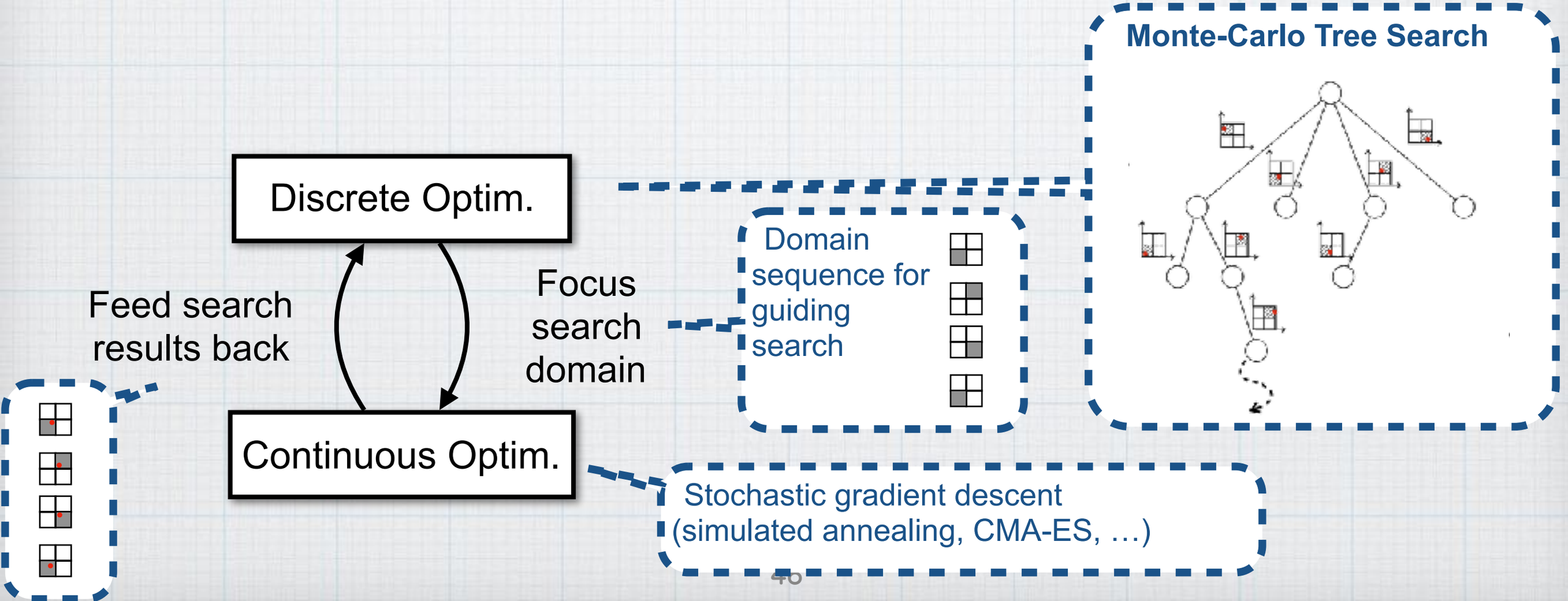
- Future work
  - Extension to a more general class of STL formula
  - Extension to robustness semantics other than the spatial one

Hierarchical Optimization in Hybrid System Falsification

[Zhang, Ernst, Sedwards, Arcaini & Hasuo, EMSOFT'18]
[Zhang, Hasuo & Arcaini, CAV'19] …

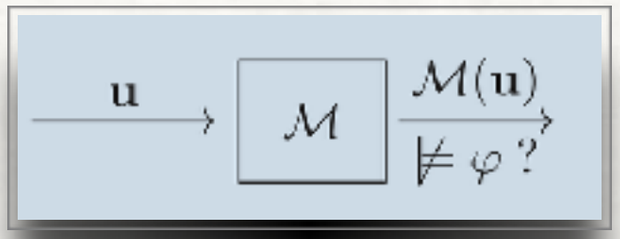
* Another example [EMSOFT'18]
  * utilizing time-causal structures
  * for search capabilities,
  * but also for coverage and exploration





Problem

- Falsification problem:



$\mathcal{M}$: black-box model

$\mathbf{u}$: input signal (piecewise constant)

$\mathcal{M}(\mathbf{u})$: output signal

φ : specification in Temporal Logic

- Goal: to find a $\mathbf{u}$ s.t. $\mathcal{M}(\mathbf{u}) \not\models \varphi$

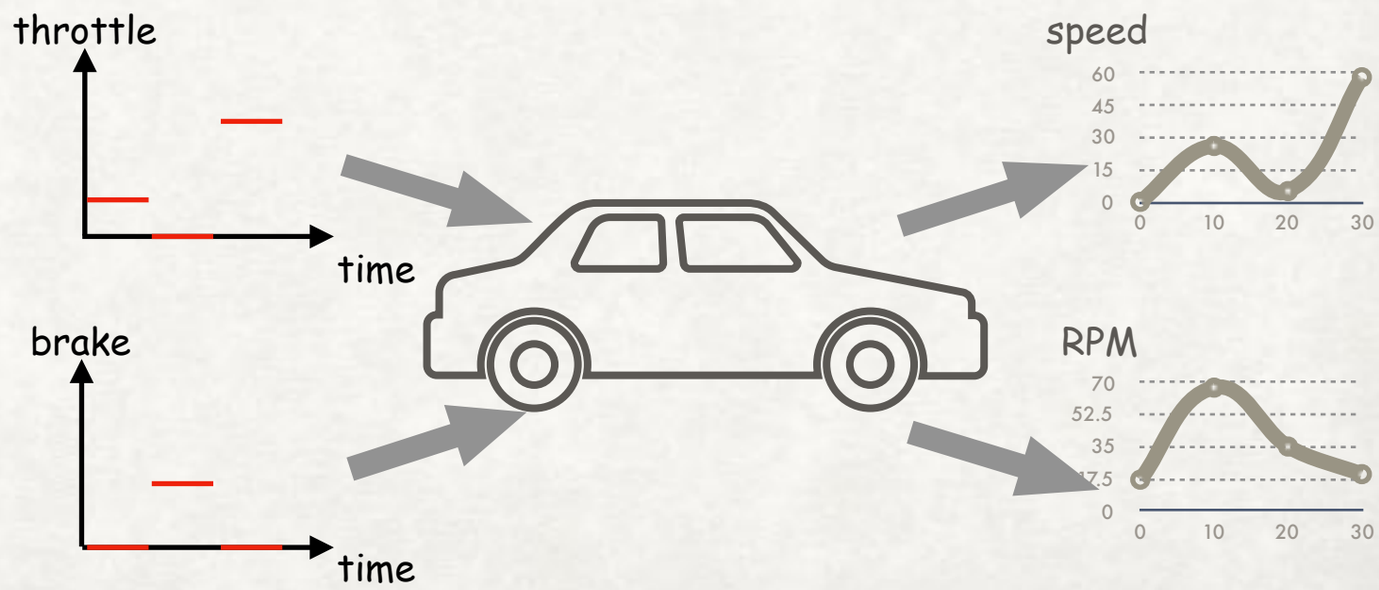
Example:

$$\Box_{[0,30]}(speed < 120)$$

Always

Bounded -time

Proposition

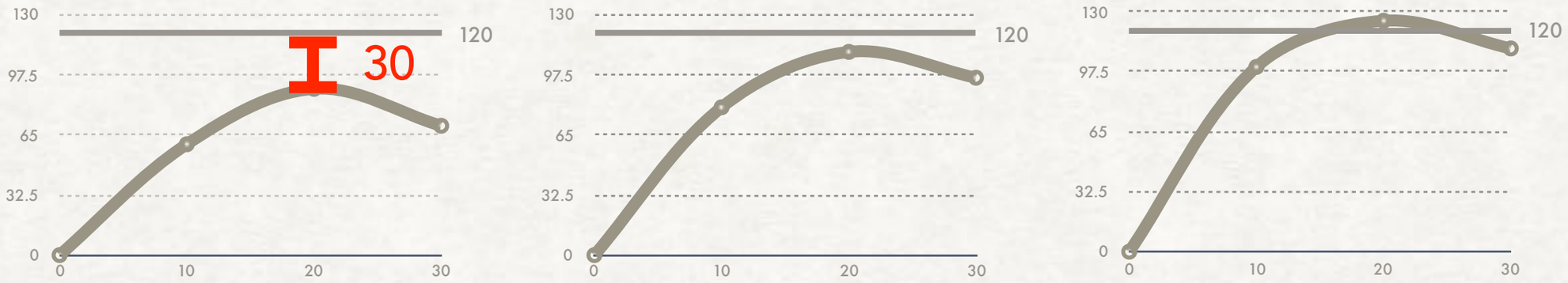


Existing solutions

- Quantitative robustness semantics of STL [Donze, Maler FORMATS'10]

e.g., $\varphi = \Box_{[0,30]}(speed < 120)$

Output Signals

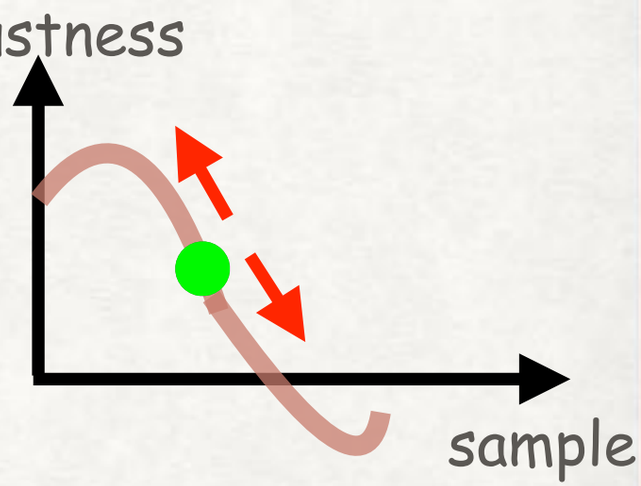


Boolean Satisfaction	True	True	False
Quantitative Robustness	30	10	-5

Boolean → Real number

Existing solutions

- falsification => optimization
  - Objective function: $[\![\mathcal{M}, \mathbf{u}]\!]$ robustness
  - Goal: min $([\![\mathcal{M}, \mathbf{u}]\!])$ to see if it can be negative

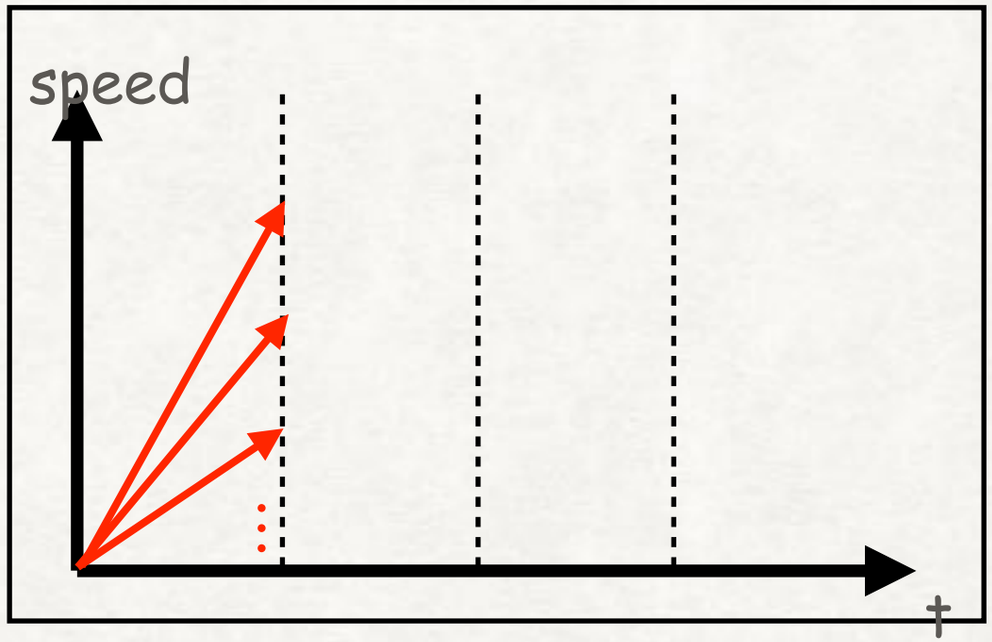


- "Hill-climbing" optimization algorithms
  - Decide next sample based on history
  - More intelligent than random sampling
  - Tools: Breach [Donze CAV'10] (CMA-ES, NM, etc.)

    S-TALIRO [Annapureddy et al. TACAS'11] (SA etc.)

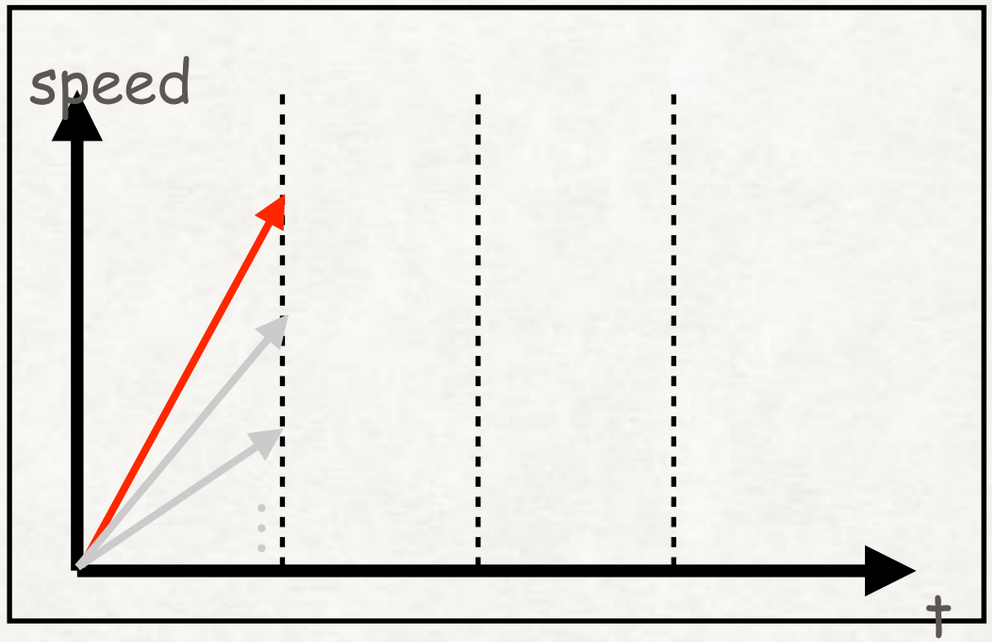
Existing solutions

- Time-staging heuristics [Ernst+SNR'18]
  - Divide time into many intervals
  - Take local optimum of each stage
  - Combine them as whole input



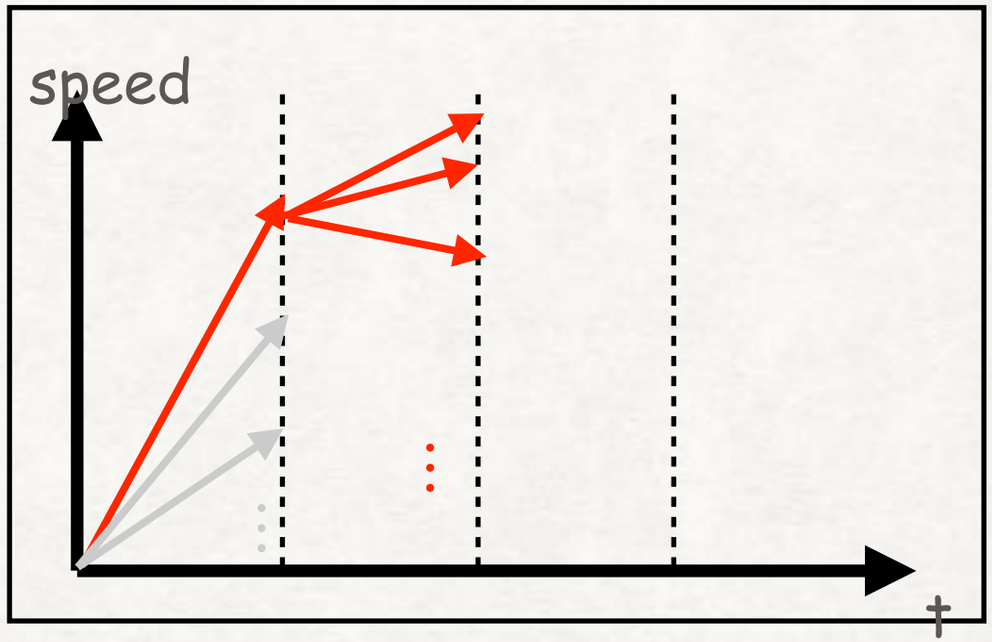
Existing solutions

- Time-staging heuristics [Ernst+SNR'18]
  - Divide time into many intervals
  - Take local optimum of each stage
  - Combine them as whole input



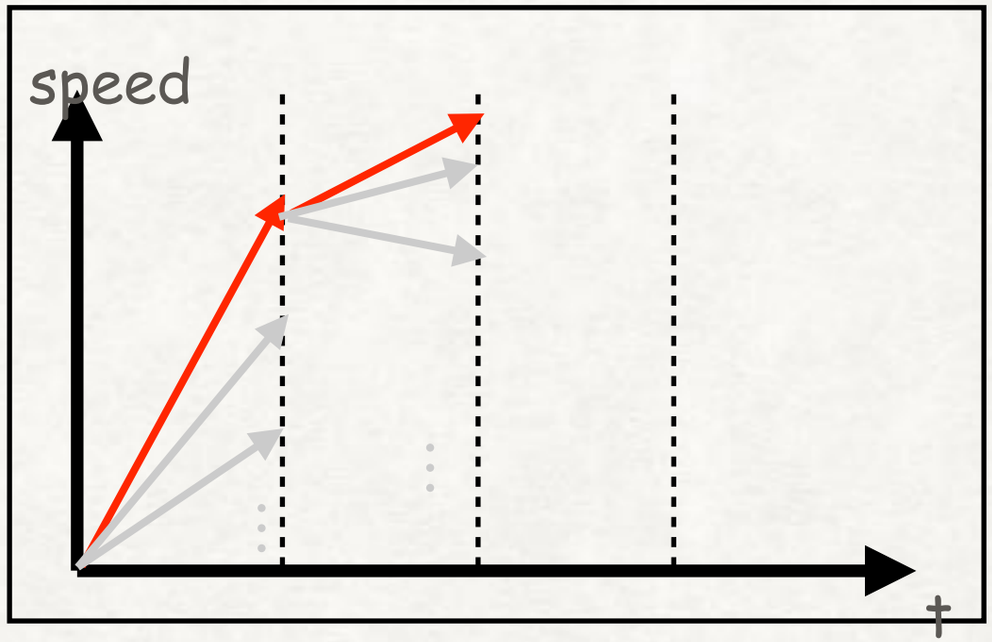
Existing solutions

- Time-staging heuristics [Ernst+SNR'18]
  - Divide time into many intervals
  - Take local optimum of each stage
  - Combine them as whole input



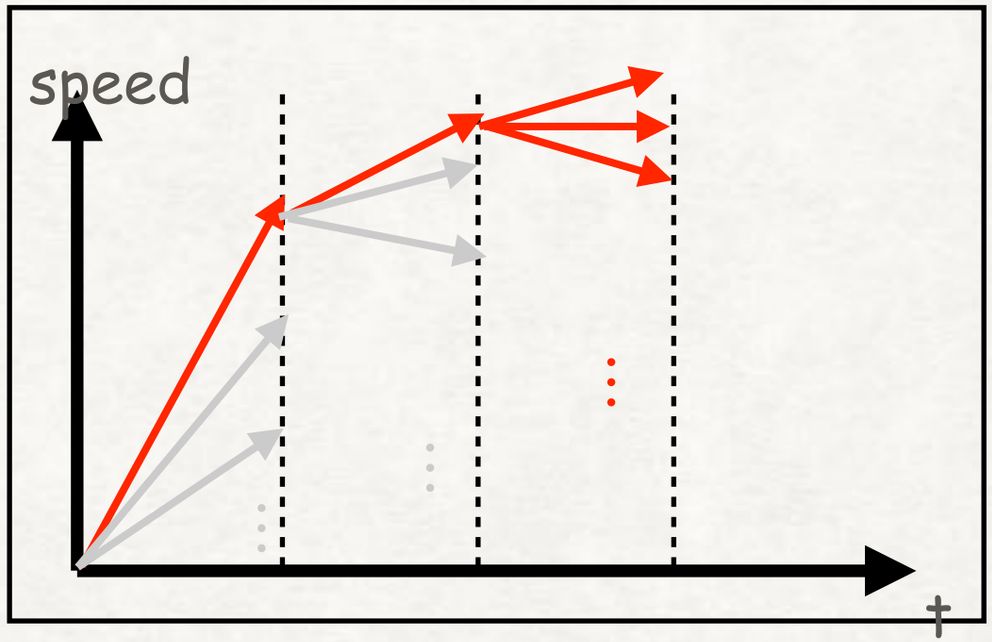
Existing solutions

- Time-staging heuristics [Ernst+SNR'18]
  - Divide time into many intervals
  - Take local optimum of each stage
  - Combine them as whole input



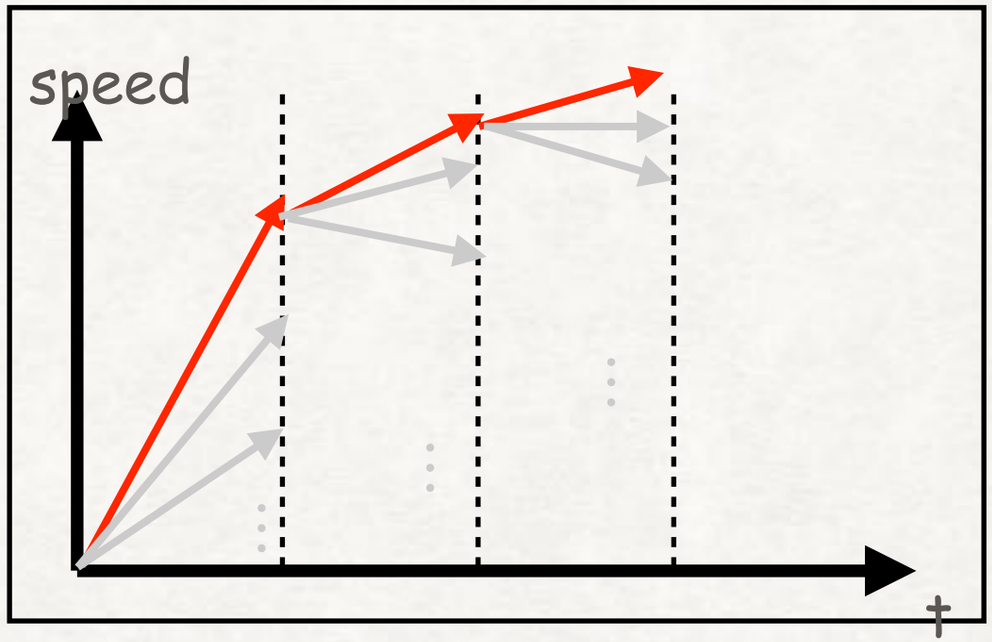
Existing solutions

- Time-staging heuristics [Ernst+SNR'18]
  - Divide time into many intervals
  - Take local optimum of each stage
  - Combine them as whole input



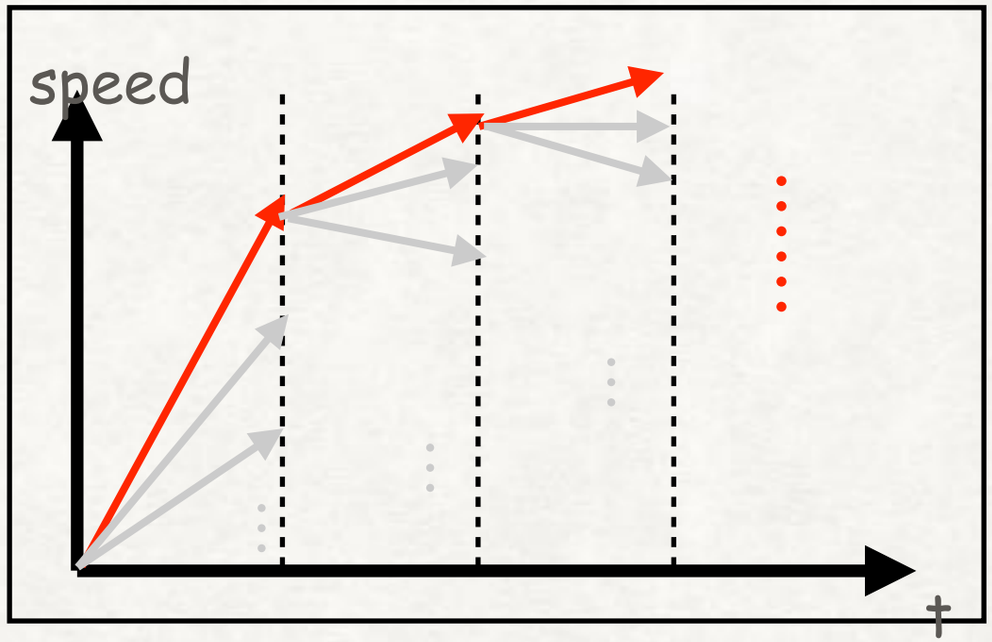
Existing solutions

- Time-staging heuristics [Ernst+SNR'18]
  - Divide time into many intervals
  - Take local optimum of each stage
  - Combine them as whole input



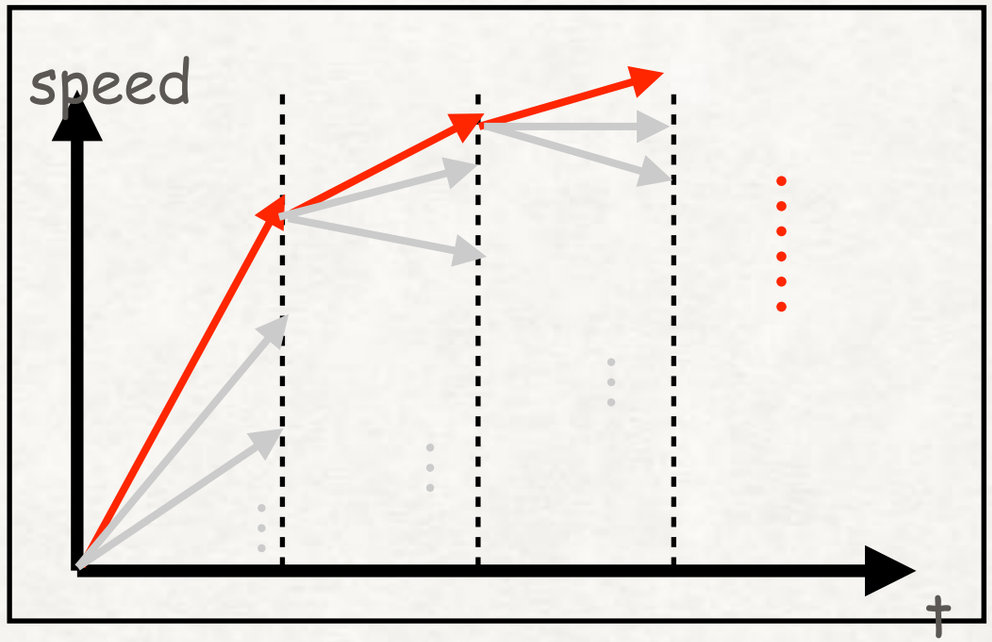
Existing solutions

- Time-staging heuristics [Ernst+SNR'18]
  - Divide time into many intervals
  - Take local optimum of each stage
  - Combine them as whole input

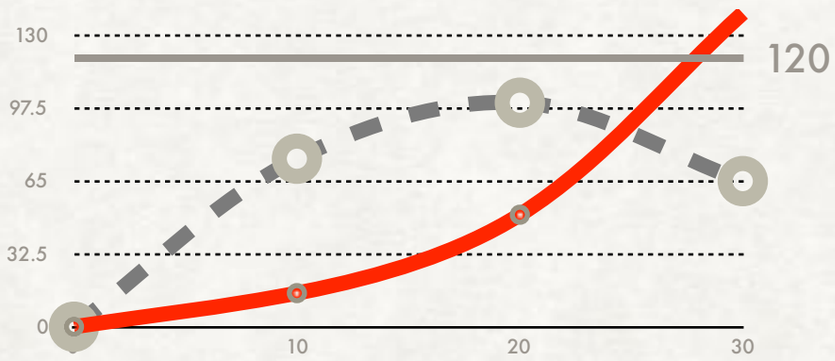


Existing solutions

- Time-staging heuristics [Ernst+SNR'18]
  - Divide time into many intervals
  - Take local optimum of each stage
  - Combine them as whole input



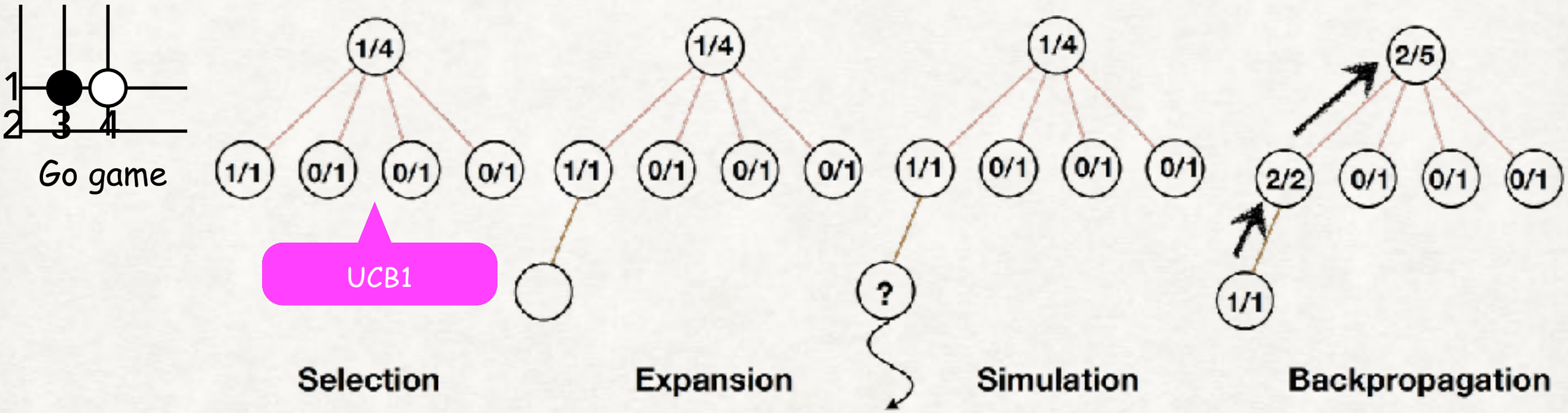
Weakness: How about this situation?



We need trackback mechanism,
we need a tree-search structure…

- But…
- Our search space is infinite…

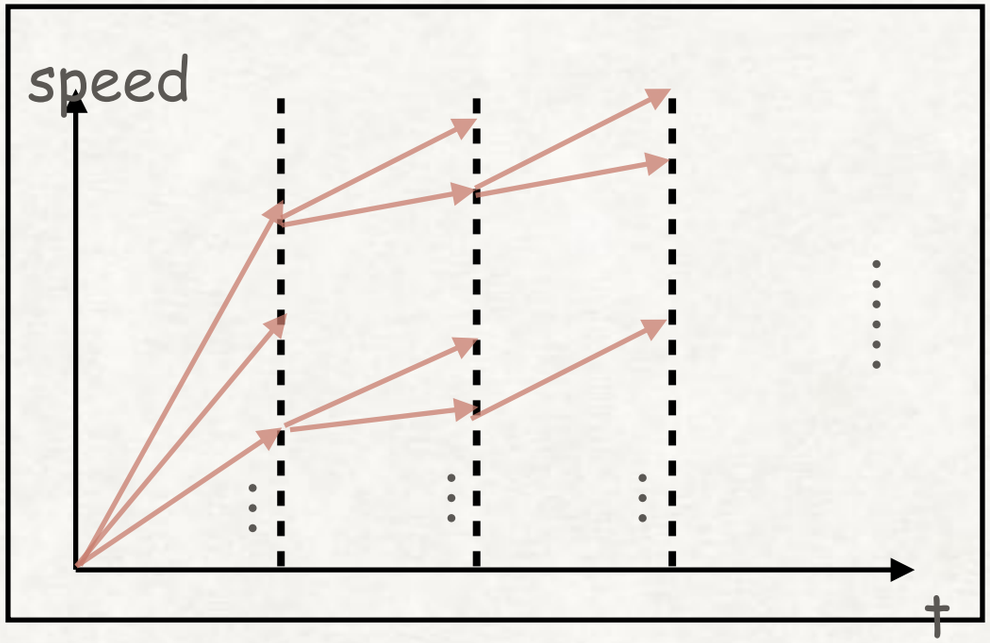
Monte Carlo Tree Search



- Pay much attention on "promising" branches
- Playout (simulation) to give each node a reward
- UCB1 [Kocsis+ECML'06] for exploration and exploitation

Application of MCTS

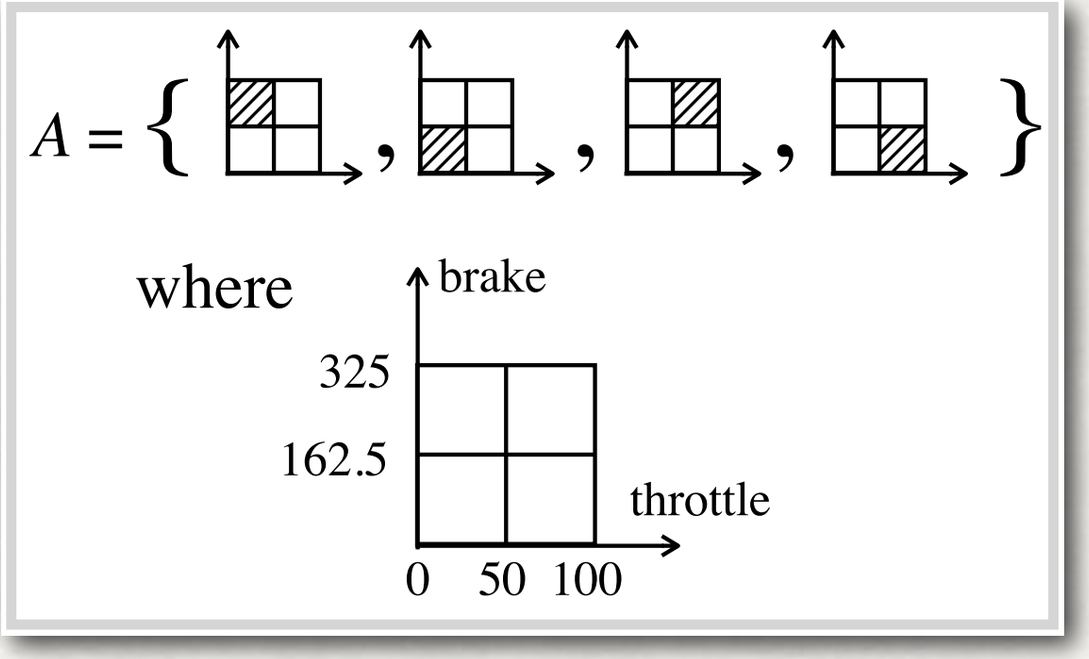
- Naive idea of applying MCTS to falsification

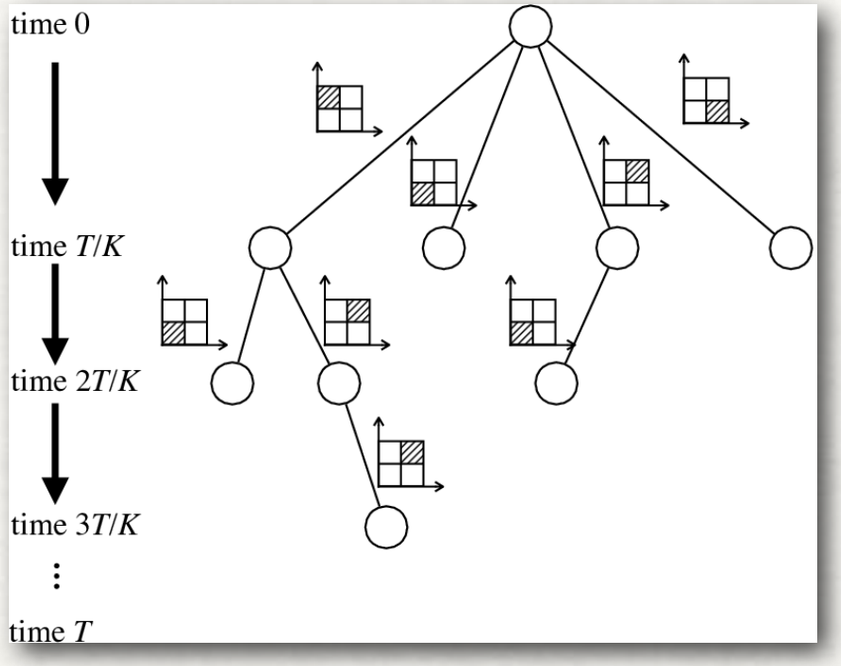


1. Divide time
2. Take a value as child
3. Randomly pick child to do playout …

- Challenges
  - Q1: How to select children from infinite input space
  - Q2: How to define reward
  - Q3: How does playout work

Q1: Infinite input space





- Partition input space at each stage
- Treat each sub-region as a child
- Reward evaluates each sub-region

Q2: reward

- Original MCTS:

function BESTCHILD(v, c)

return $\arg\max_{v' \in \text{children of } v} \frac{Q(v')}{N(v')} + c\sqrt{\frac{2\ln N(v)}{N(v')}}$

Reward: an estimated winning rate

v' : expanded child

$Q(v')$: winning times

$N(v')$: visiting times

$N(v)$: visiting times to the parent

- Our definition:

$$\left(1 - \frac{R(v')}{\max_{w \in \mathcal{T}} R(w)}\right)$$

function BESTCHILD(v, c)

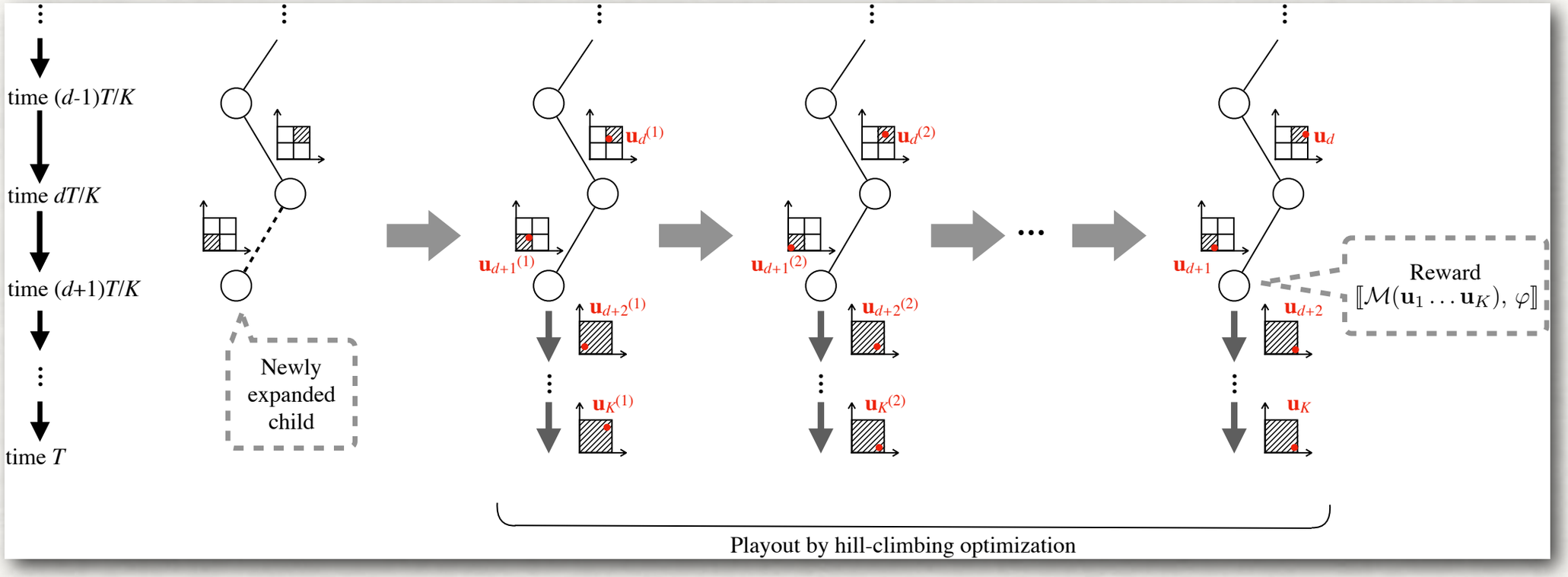
return $\arg\max_{v' \in \text{children of } v} \left(\left(1 - \frac{R(v')}{\max_{w \in \mathcal{T}} R(w)}\right) + c\sqrt{\frac{2\ln N(v)}{N(v')}} \right)$

Reward: negatively correlated to robustness

$R(v')$: robustness

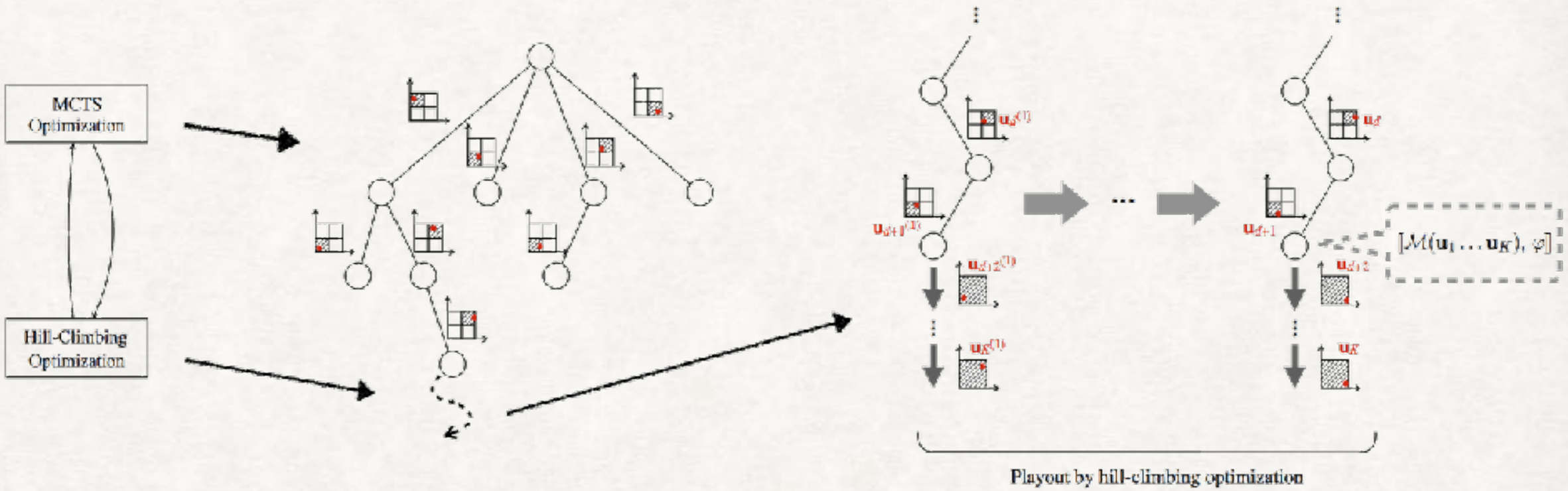
$\max_{w \in \mathcal{T}} R(w)$: the max (worst) robustness over the whole tree

Q3: playout



- Goal: evaluate the expanded child
- Steps:
  - Collect the sequence of sub-regions from the root to the expanded child
  - Include all the input region of the remaining stages
  - Run hill-climbing optimization within a timeout

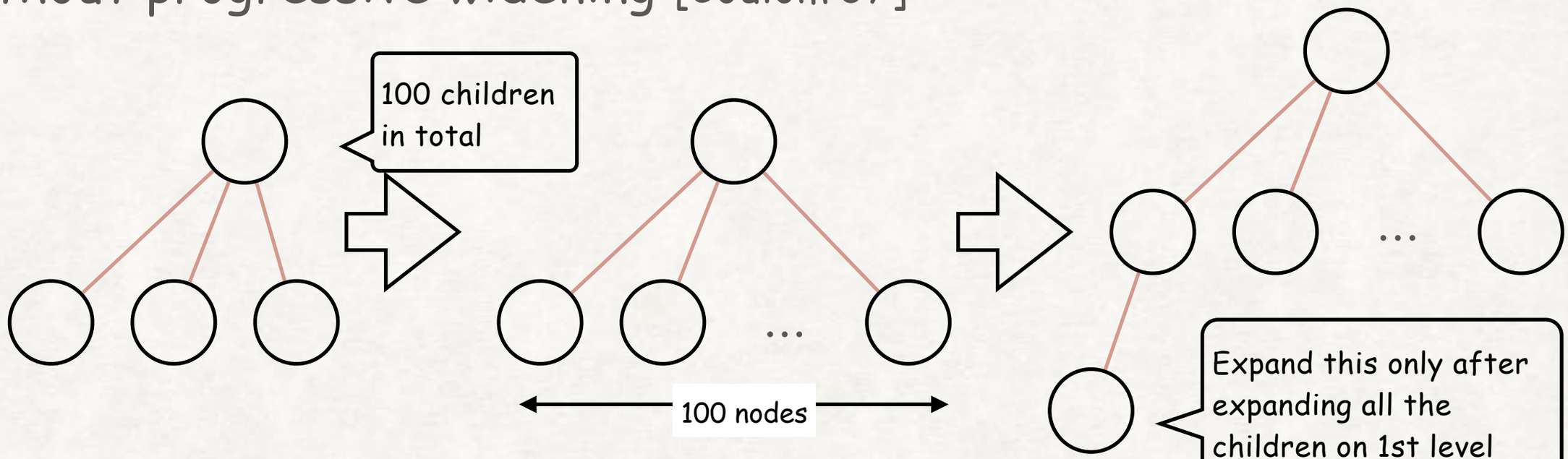
Basic Alg.: Two-layered optimization



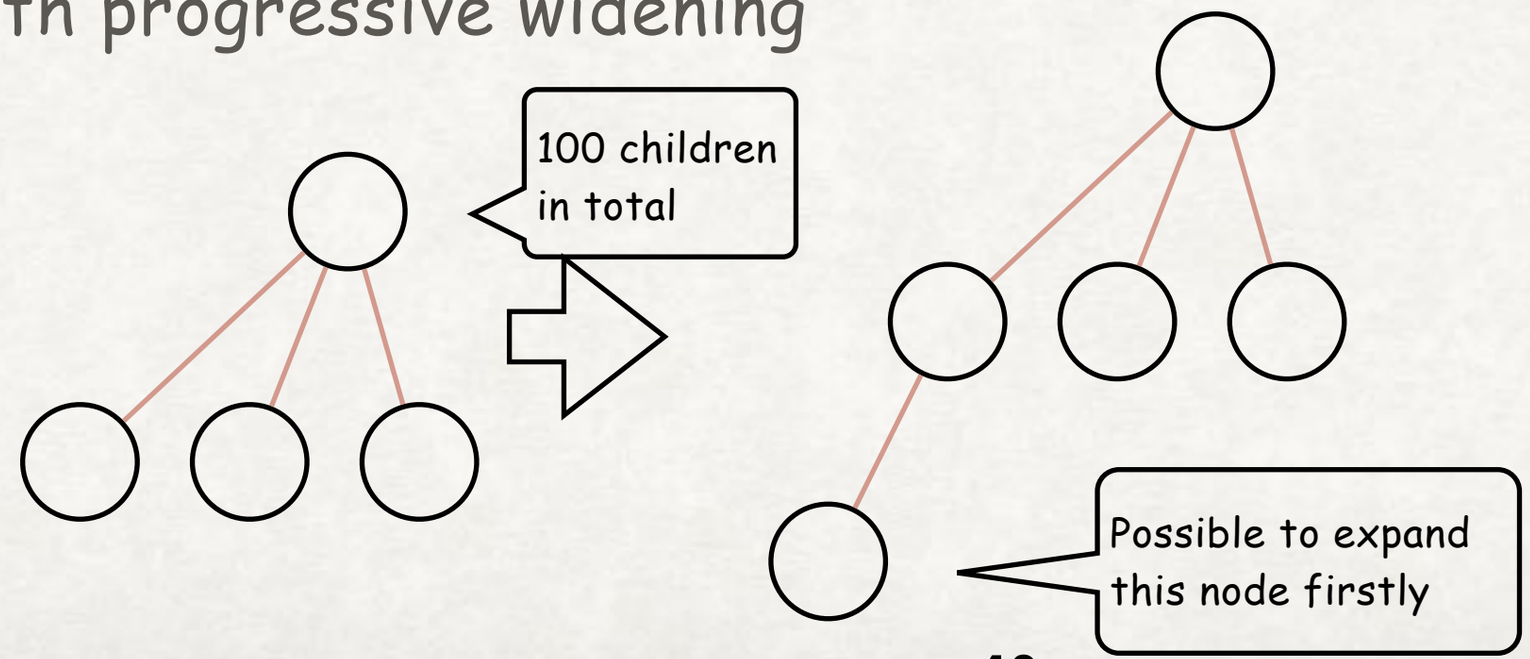
- Two-layered optimization framework:
  - MCTS: decide the directions of the search based on the reward given by hill-climbing
  - Hill-climbing: compute the concrete robustness values of the path assigned by MCTS

Biased Alg.: Progressive widening

- Without progressive widening [Coulom'07]



- With progressive widening



$$C \cdot N(v)^{\alpha}$$

Experimental evaluation

- Evaluated algorithms:
  - Breach (Hill-climbing)
  - Basic algorithm (MCTS + Hill-climbing)
  - Biased algorithm (MCTS + Hill-climbing + progressive widening)
- Benchmarks:
  - Automatic transmission
  - Abstract fuel control
  - Free floating robot

Experimental evaluation

	Algorithm	AT model S1 succ.	AT model S1 time	AT model S2 succ.	AT model S2 time	AT model S3 succ.	AT model S3 time	AT model S4 succ.	AT model S4 time	AT model S5 succ.	AT model S5 time	AFC model Sbasic succ.	AFC model Sbasic time	AFC model Sstable succ.	AFC model Sstable time	FFR model Strap succ.	FFR model Strap time
	Random	10/10	108.9	10/10	289.1	1/10	301.1	0/10	-	0/10	-	**6/10**	**278.7**	10/10	242.6	4/10	409.3
CMA-ES	Breach	10/10	21.9	6/10	30.3	**10/10**	**193.9**	4/10	208.8	3/10	75.5	**10/10**	**111.7**	3/10	256.3	**10/10**	**119.8**
CMA-ES	Basic	10/10	15.8	10/10	108.5	10/10	697.1	7/10	786.8	9/10	384.4	10/10	182.0	7/10	336.9	10/10	338.0
CMA-ES	P.W.	**10/10**	**10.8**	**10/10**	**65.7**	10/10	728.6	**7/10**	**767.8**	**10/10**	**648.1**	10/10	177.1	**8/10**	**272.9**	10/10	473.9
GNM	Breach	**10/10**	**5.4**	**10/10**	**151.4**	0/10	-	0/10	-	0/10	-	**10/10**	**171.4**	0/10	-	0/10	-
GNM	Basic	10/10	12.4	10/10	162.3	**10/10**	**185.6**	7/10	261.9	7/10	163.7	10/10	**227.1**	2/10	378.5	**10/10**	**162.2**
GNM	P.W.	10/10	60.8	9/10	110.7	8/10	211.2	**8/10**	**313.0**	**10/10**	**178.7**	10/10	252.0	**6/10**	**153.2**	6/10	197.4
SA	Breach	**10/10**	**160.1**	0/10	-	3/10	383.7	0/10	-	3/10	80.4	0/10	-	6/10	307.0	3/10	92.8
SA	Basic	10/10	264.8	9/10	236.1	**8/10**	**385.6**	**8/10**	**505.3**	7/10	341.2	5/10	391.3	**8/10**	**273.8**	**10/10**	**273.2**
SA	P.W.	10/10	208.7	**10/10**	**377.6**	8/10	666.0	7/10	795.4	**10/10**	**624.2**	**8/10**	**665.7**	6/10	293.7	10/10	390.9

S1 $\square_{[0,30]}\ (speed < 120)$

S2 $\square_{[0,30]}\ (gear = 3 \rightarrow speed \geq 20)$

S3 $\Diamond_{[10,30]}\ (speed \leq 53 \vee speed \geq 57)$

S4 $\square_{[0,29]}\ (speed < 100) \vee \square_{[29,30]}(speed > 65)$

S5 $\square_{[0,30]}(rpm < 4770 \vee \square_{[0,1]}(rpm > 600))$

Sbasic $\square_{[11,30]}(\neg(|AF - AF_{\text{ref}}| > 0.05 * 14.7))$

Sstable $\neg(\Diamond_{[6,26]}\square_{[0,4]}(AF - AF_{\text{ref}} > 0.01 * 14.7))$

Strap $\neg\Diamond_{[0,5]}(x, y \in [3.9, 4.1] \wedge \dot{x}, \dot{y} \in [-1, 1])$

Model:

- AT: Automatic Transmission
- AFC: Abstract Fuel Control
- FFR: Free Floating Robot

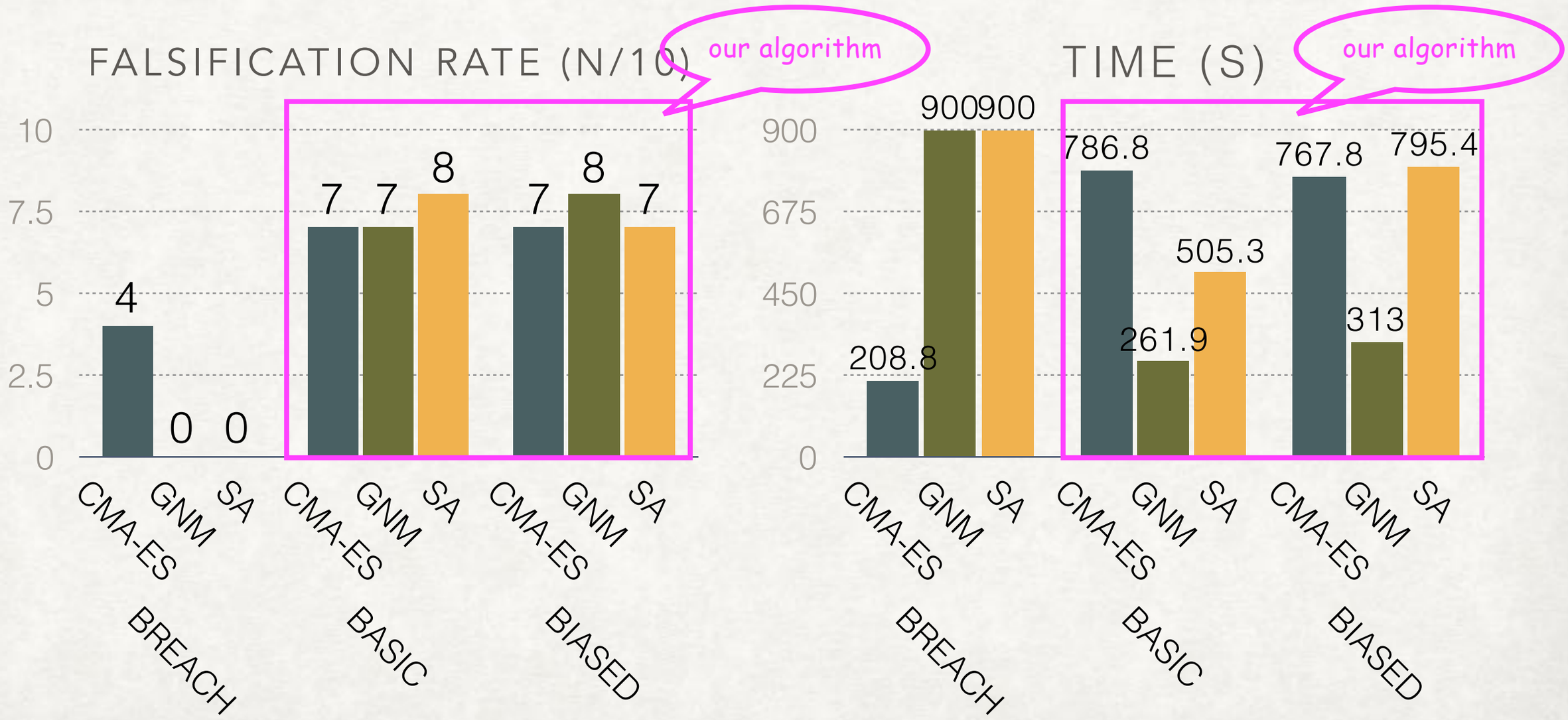
Algorithm:

- Breach: optimization
- Basic: MCTS + Hill-Climbing
- P.W.: MCTS + Hill-Climbing + progressive widening

Experimental evaluation

Model: Automatic transmission:

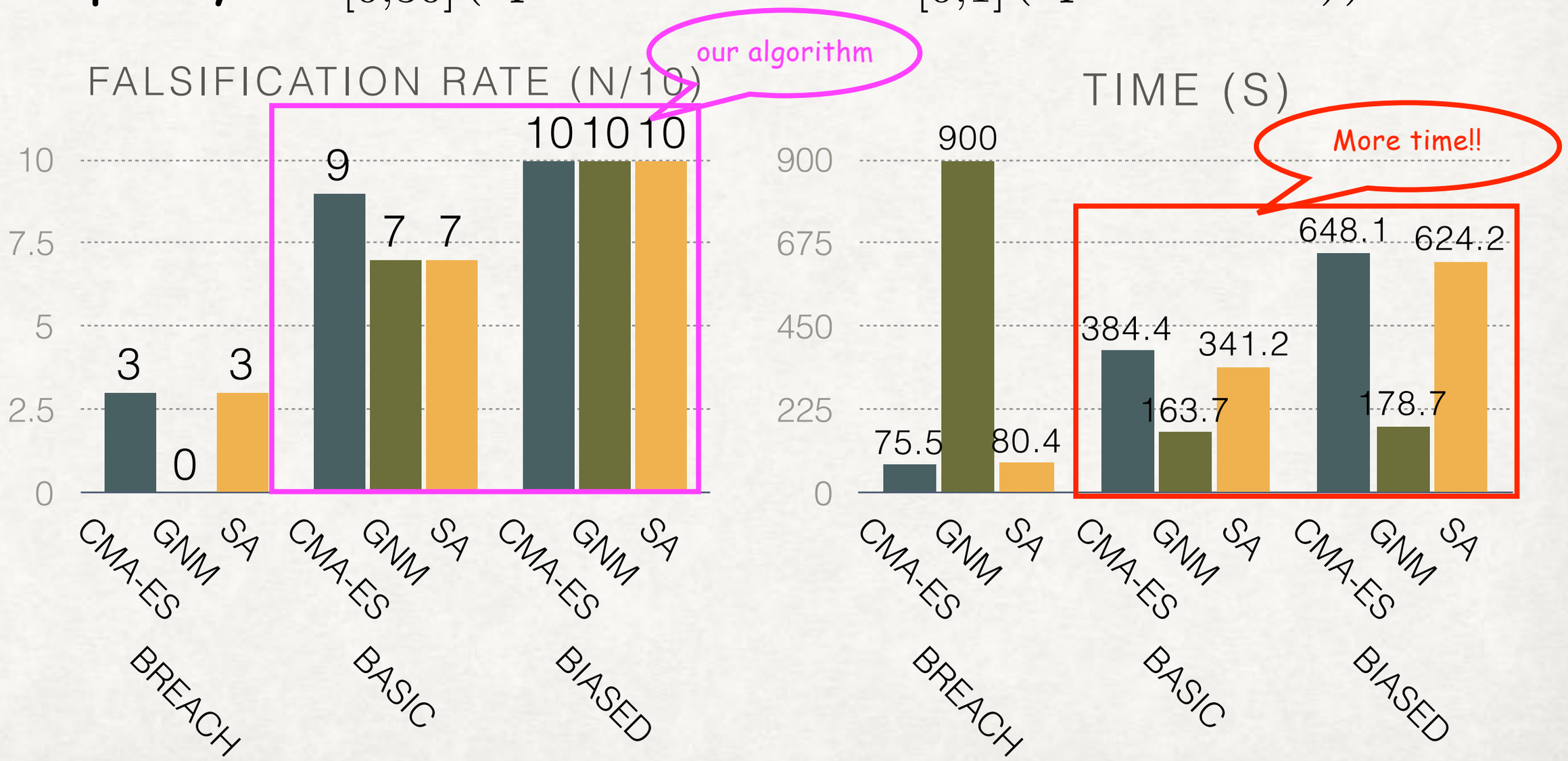
Property: $\square_{[0,29]}(speed < 100) \vee \square_{[29,30]}(speed > 65)$



Experimental evaluation

Model: Automatic transmission

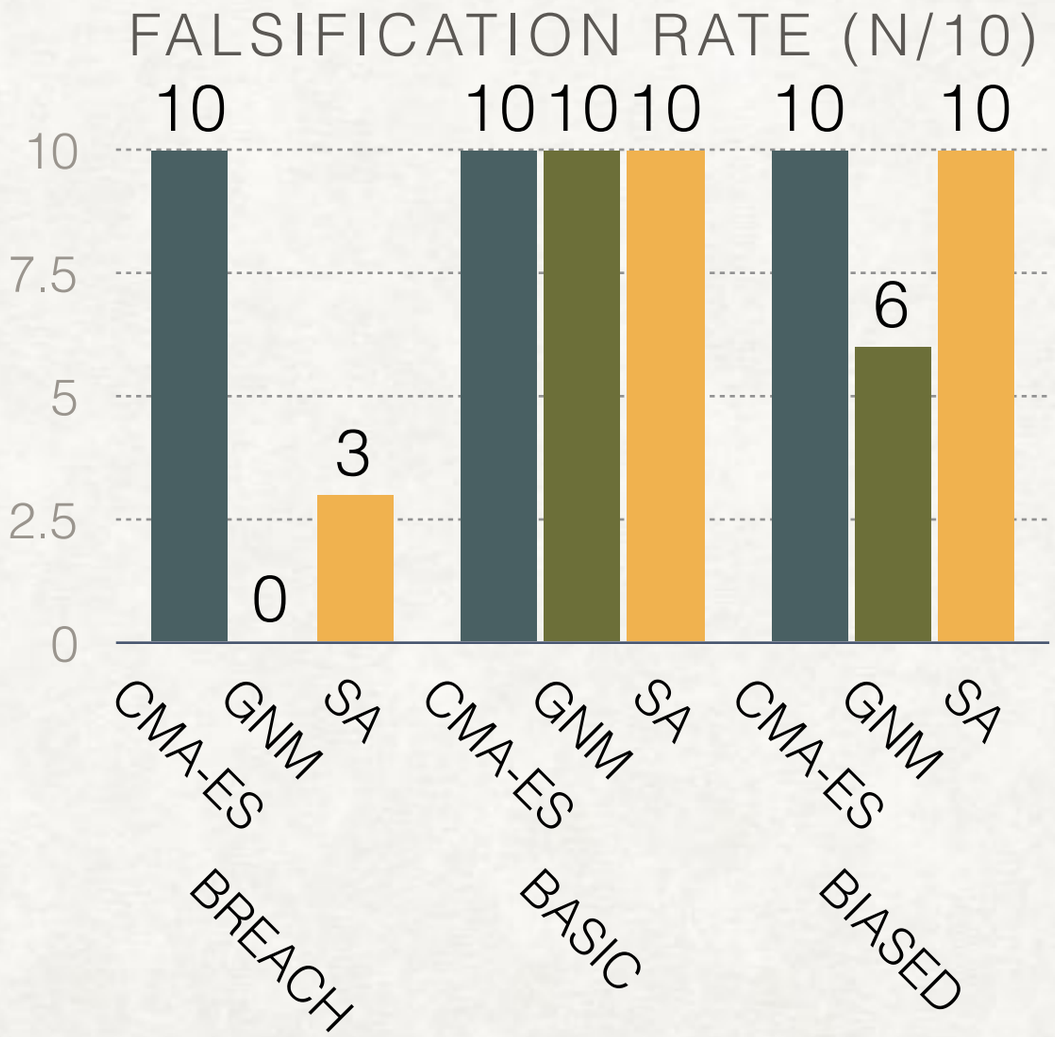
Property: $\square_{[0,30]}(rpm < 4700 \vee \square_{[0,1]}(rpm > 600))$

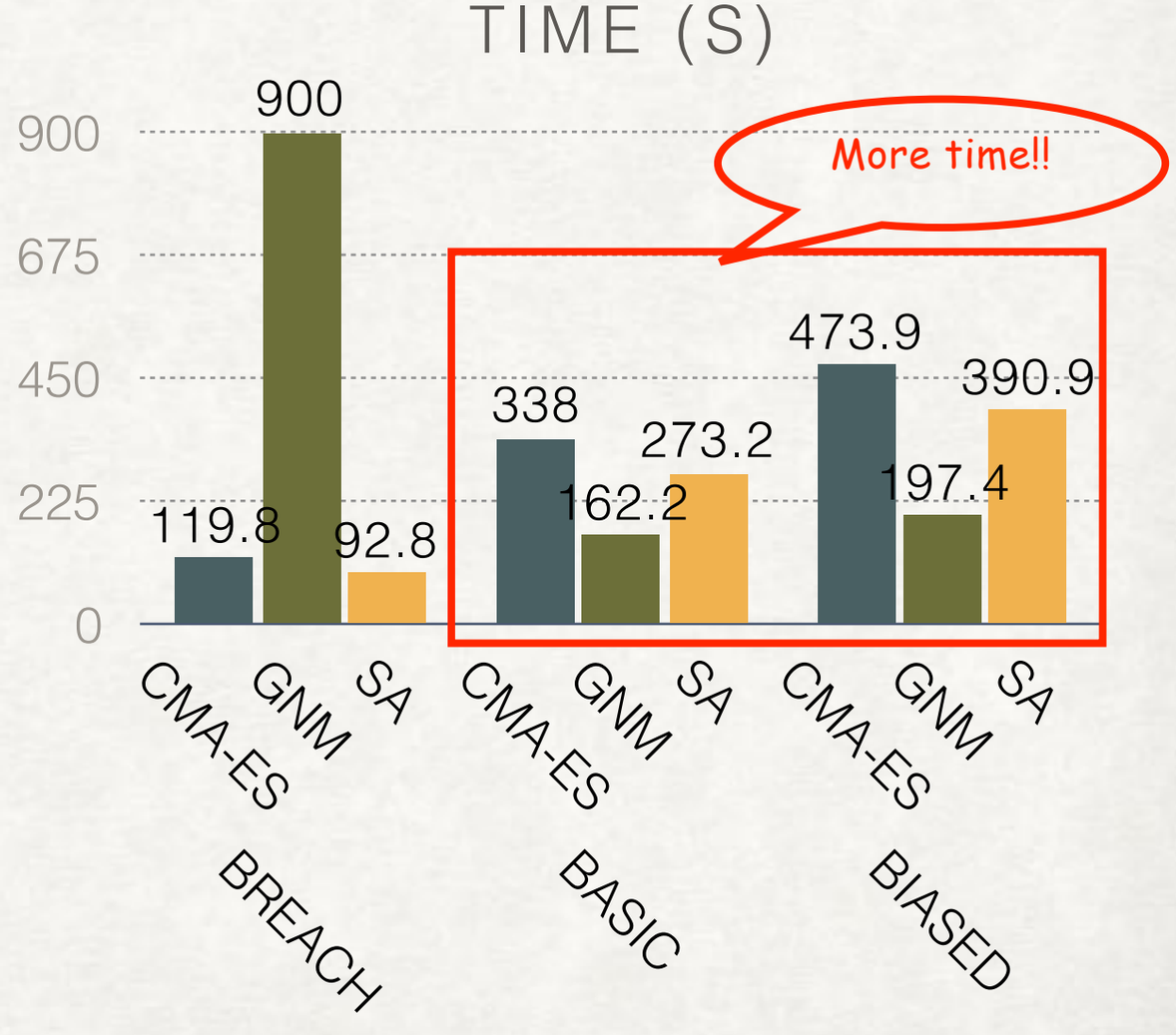


Experimental evaluation

Model: Free floating robot

Property: $\neg\Diamond_{[0,5]}(x, y \in [3.9, 4.1] \wedge \dot{x}, \dot{y} \in [-1, 1])$





Experimental evaluation

- Our approach improves the falsification rate in most cases
- Sometimes our approach takes more time to find the solution
  - MCTS takes more time on exploration and it is acceptable.

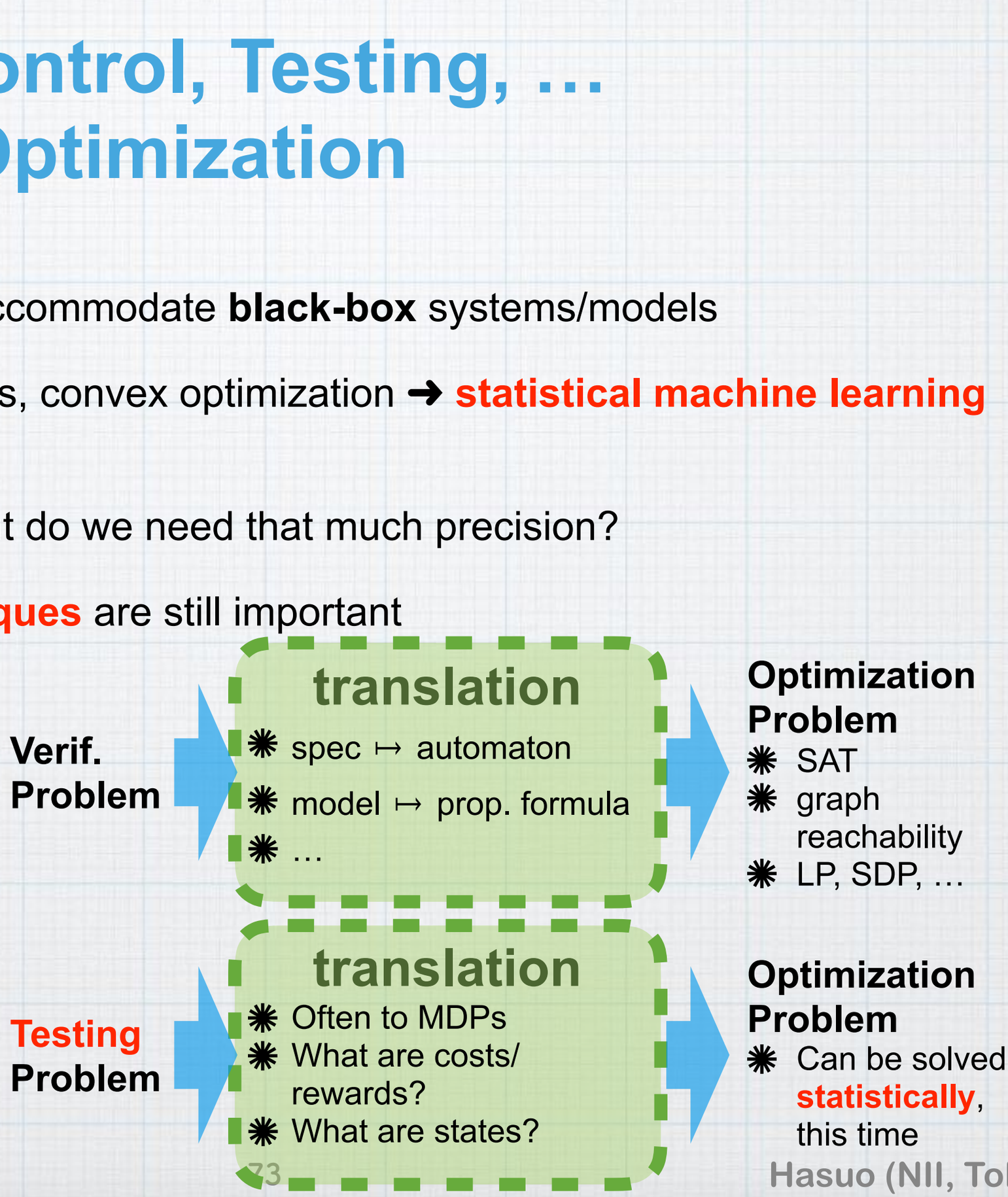
Conclusion and Future work

- This paper proposes a two-layered framework:
  - High level: Monte Carlo Tree Search to guide the search direction
  - Low level: Hill-climbing to do playout to give reward to each node
- Experimental evaluation shows that our approach can solve some hard problems within acceptable time.

- Future work:
  - Quantitative coverage metric

Verification, Control, Testing, … Reduction to Optimization

- Verification ➜ **testing**
  - Fight **complexity**. Accommodate **black-box** systems/models
- Rigorous graph algorithms, convex optimization ➜ **statistical machine learning**
  - Faster
  - Maybe imprecise. But do we need that much precision?
- **Formal methods techniques** are still important
  - Formal verification:
  - FM-assisted testing

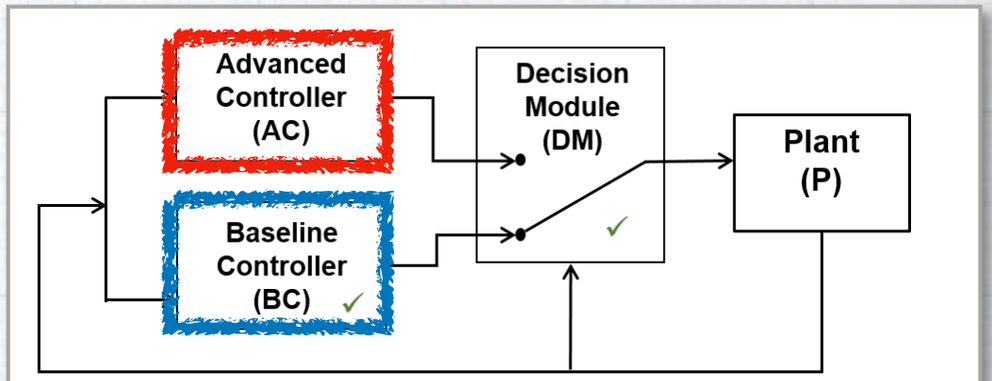


Outline

* Cyber-physical systems under uncertainties
* Formal methods and testing for CPS
  * Hybrid system falsification
    * Logical connectives [CAV'19]
    * Causality in time [EMSOFT'18]
    * … as demonstration of combining logical and statistical
  * Other topics

Formal Safety Architecture

* Example: simplex architecture (right)
  * AC is
    complex, performance-oriented, black-box
  * BC is
    simple, safety-oriented, (hopefully) white-box



Phan et al., ACSD'17

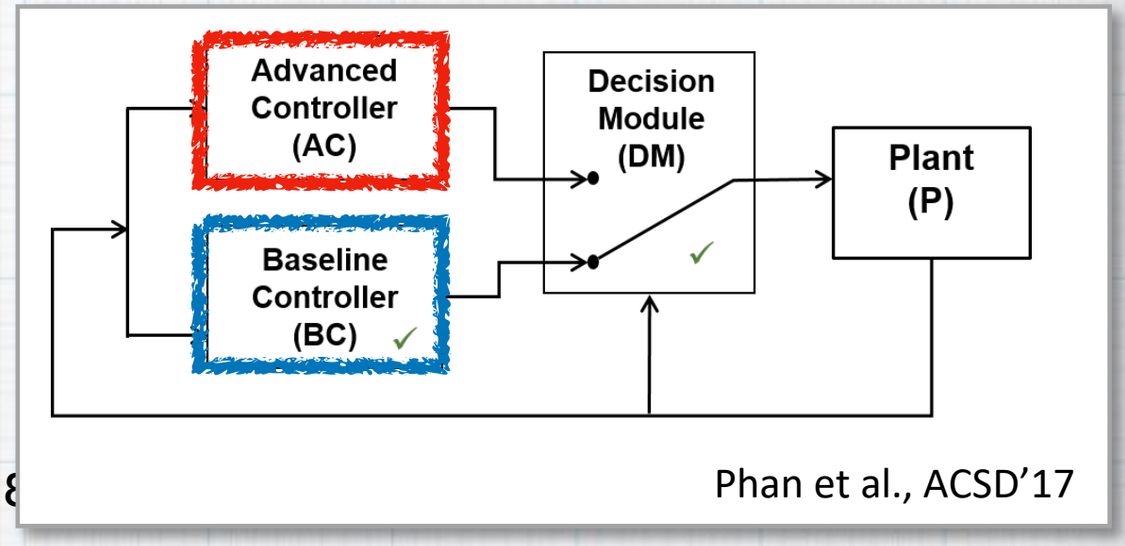
* Idea: we can verify the whole system
  even if AC is a black-box!
  * Enough to show:
    safety of BC, and correctness of DM
  * We impose certain contracts on AC

* Related
  * FM4AI (ML/AI components as AC)
  * Monitoring (checking contracts on AC)

* One promising way to
  make formal verification down-scalable
  * Weaker contracts on AC
    ➜ weaker safety guarantee (but hopefully non-zero)

Formal Safety Architecture

* At ERATO MMSD:
  we formalize, verify and refine
  safety architectures



Phan et al., ACSD'17

* Event-B: a formal modeling language
  [Abrial, "The Event-B Book", 2010 CUP] [Kobayashi+, ICFEM'18
  Based on state transition systems.

A tool Rodin supports:

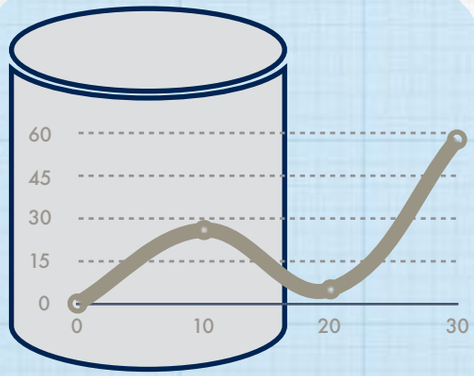
* Safety proofs
* Incremental modeling by refinements
  * Flexibility in choosing model fidelity ➜ down-scaling
  * From (our) general model to (industry partner's) individual model

Pattern Matching against Timed Automata, Monitoring

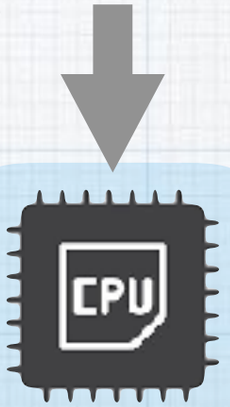
Specification

"Frequent gear changes within 3 sec after shifting up to 4th"

Running system



Log



Monitor

Monitoring result

"From 18.9 sec. to 23.2 sec."

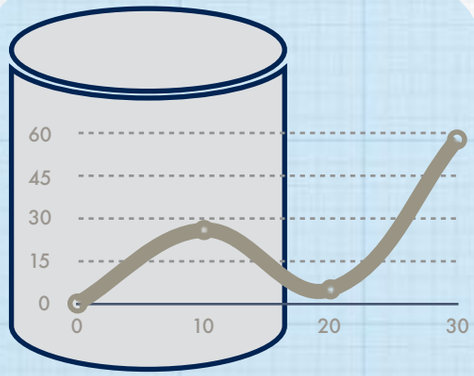
- Runtime verification, monitoring
- Not straightforward, esp. when specs involve timing constraints
  - Speed requirements (GBs of log per second)
  - Computing resource (embedded)
  - …
- Industry needs
- Technically: theory of **(parametrized) timed automata**

Pattern Matching against Timed Automata, Monitoring

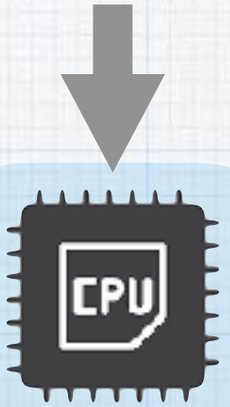
Specification

"Frequent gear changes within 3 sec after shifting up to 4th"

Running system



Log



Monitor

Monitoring result

"From 18.9 sec. to 23.2 sec."

* Use cases
  * "Here is 1 PB of log, and I want to extract its relevant parts"
  * "Raise an alert if this specific type of anomaly occurs"

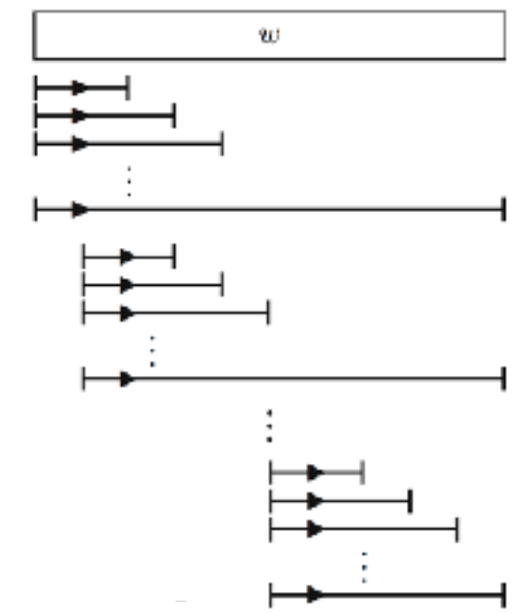
Monitoring: Problem Formulations

* Given: a log, discrete time w = abaaacb…bbc
  a spec φ ”no occurrence of c for 6 steps after b”
  Answer: all subsequences of w that satisfy φ

* Given: a log, continuous time w = (a, 0.12) (b, 1.28) …
  a spec φ ”no occurrence of c for 6 seconds after b”
  Answer: all subsequences of w that satisfy φ

[Ulus, CAV’17] [Waga+, FORMATS’17]

Nontrivial due to temporal causality



Infinitely many such subsequences (starting at t=1? t = 1.01? t = 1.001? …)
➜ Efficient representation & computation by zones

* Given: a log, continuous time w = (a, 0.12) (b, 1.28) …
  a parametrized spec φ(p)
  ”no occurrence of c for p seconds after b” “b occurs with a period of p seconds”
  Answer: all the pairs of (p, (a subseq. of w that satisfies φ)))

[Andre+, ICECCS’18] [Waga+, NFM’19] [Waga+, CAV’19]

In industry, fixing a spec is a big challenge.
Parameters ➜ flexibility in specs

Monitoring: Our Achievements

* Given: a log, discrete time w = abaaacb…bbc
  a spec φ "no occurrence of c for 6 steps after b"
  Answer: all subsequences of w that satisfy φ

* Given: a log, continuous time w = (a, 0.12) (b, 1.28) …
  a spec φ "no occurrence of c for 6 seconds after b"
  Answer: all subsequences of w that satisfy φ
  [Ulus, CAV'17] [Waga+, FORMATS'17]

Efficient algorithm from theory of timed automata. Processes ~ 1M events/second (laptop).

[Waga+, FORMATS'17]
https://github.com/maswag/monaa

Also implemented on Renesas RH850

Efficient algorithm using parametrized timed automata. Processes ~ 10K events/second (laptop). [Waga+, NFM'19]
https://github.com/maswag/symon

* Given: a log, continuous time w = (a, 0.12) (b, 1.28) …
  a parametrized spec φ(p)
  "no occurrence of c for p seconds after b" "b occurs with a period of p seconds"
  Answer: all the pairs of (p, (a subseq. of w that satisfies φ)))
  [Andre+, ICECCS'18] [Waga+, NFM'19] [Waga+, CAV'19] …

Outline

* Cyber-physical systems under uncertainties
* Formal methods and testing for CPS
  * Hybrid system falsification
    * Logical connectives [CAV'19]
    * Causality in time [EMSOFT'18]
    * … as demonstration of combining logical and statistical
  * Other topics

Conclusions

* Focused on **safety-critical** CPS with statistical ML components
* Don't trust statistical AI.
  Use them with precaution
  * System-level safety
* **Formal methods** are not applicable per se
  ➜ combine **logical** and **statistical** reasoning
  * Hierarchical optimization in falsification, as an example
* Quality assurance under uncertainties
  * A scientific and engineering problem,
  * that is challenging but rewarding!