# Statistical Model Checking for Hyperproperties

**Yu Wang***, Siddhartha Nalluri*, Borzoo Bonakdarpour**, Miroslav Pajic*

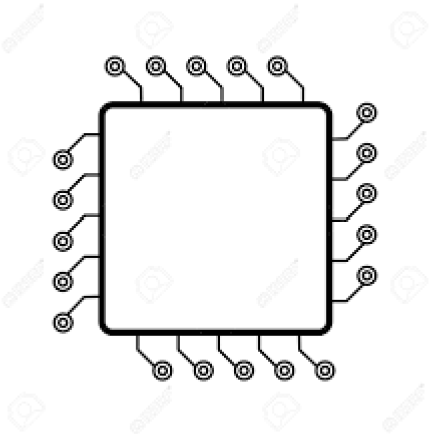
*Department of Electrical and Computer Engineering
Duke University

**Department of Computer Science
Michigan State University

# Probabilistic Systems

Many computer systems have probabilistic executions.

| Probabilistic Program | Randomized Network Protocol | Randomized Hardware Control | Cyber-Physical Systems |
|---|---|---|---|

PRIVATE and PUBLIC variables may have (implicit) information flow.

**Example** (Probabilistic Interference): Consider a parallel program $\boldsymbol{P}$ of two threads

$$\mathbf{th_1}:\ \text{while } h > 0 \text{ do } \{h \leftarrow h - 1; l \leftarrow 1\}\ |\ \mathbf{th_2}: l \leftarrow 2$$

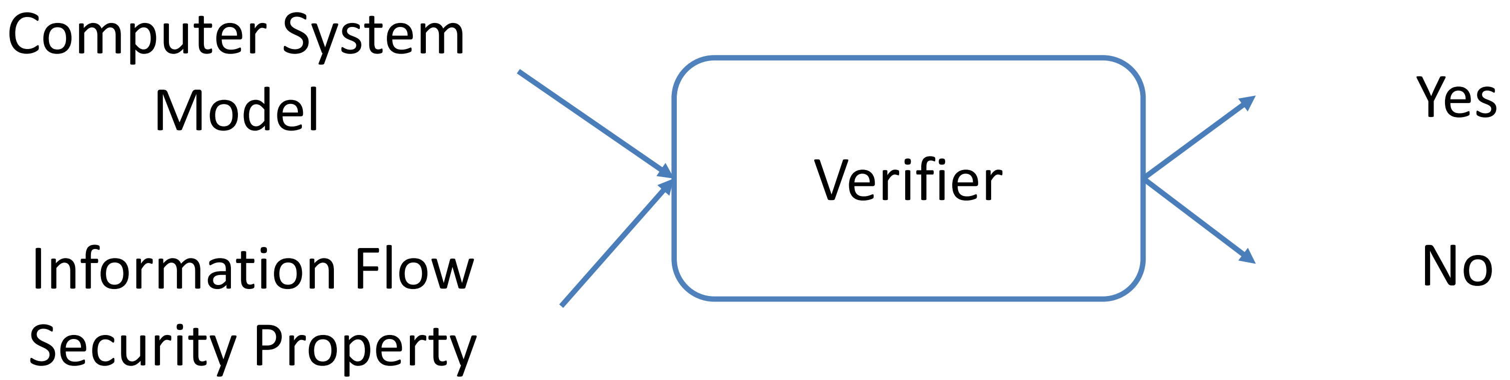
where $h \in \{1, 2\}$ is private; and $l \in \{1, 2\}$ is public.

At each time, the CPU randomly chooses to run one step of a thread.

- If $h = 1$, $\mathbf{th_1}$ has 1 steps, and $\mathbf{th_2}$ has 1 step. When $\boldsymbol{P}$ stops, $l = 1$ w.p. $1/2$.
- If $h = 2$, $\mathbf{th_1}$ has 2 steps, and $\mathbf{th_2}$ has 1 step. When $\boldsymbol{P}$ stops, $l = 1$ w.p. $1/3$.

# Formal Verification for Information Flow Security Specifications

Goal: Automated reasoning of general information security properties.




Main Questions:

1. How to formally express information flow security properties?
2. How to develop mathematically-rigorous verification algorithms?

# How to formally express properties?

Time-related properties of a single execution is formally expressible by temporal logic.

The logic **PCTL***:

$$\varphi ::= a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \boldsymbol{X}\varphi \mid \varphi \boldsymbol{U_T} \varphi \mid \mathbb{P}_{\sim \boldsymbol{p}}\varphi$$

- a is an atomic proposition;
- ¬ means "not"; ∧ means "and";
- $\boldsymbol{X}\phi$ means $\phi$ holds NEXT;
- $\boldsymbol{\phi_1} \mathbf{U}_T \phi_2$ means $\phi_1$ holds UNTIL $\phi_2$ becomes true within time $T$;
- $\sim \in \{>, <, \geq, \leq\}$, $\mathbb{P}_{>\boldsymbol{p}}\ \varphi$ means $\phi$ holds with PROBABILITY $> p$

Examples

- Value of $h$ is ALWAYS above 2 with PROBABILITY below 0.1:
  $\mathbb{P}_{<0.1}(\mathrm{T}\boldsymbol{U}(h > 2))$

# PCTL* Cannot Express Information Flow Security

Probabilistic NON-Interference:

$$\mathbb{P}^{\pi_1}\big((h=0)^{\pi_1} \text{ finally leads to } (l=0)^{\pi_1}\big) \approx \mathbb{P}^{\pi_2}\big((h=1)^{\pi_2} \text{ finally leads to } (l=0)^{\pi_2}\big)$$

Probabilistic Noninterference is a hyperproperty about the relation between multiple system executions.

PCTL* cannot express hyperproperties, since the logical connectives are invariably taken for a single executions.

# The Logic HyperPCTL*

MICHIGAN STATE UNIVERSITY
Duke PRATT SCHOOL of ENGINEERING

HyperPCTL*:

$$\varphi ::= \mathrm{a}^{\pi} \mid \varphi^{\pi} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi\ \boldsymbol{U_T}\ \varphi \mid p \sim p$$

$$p ::= \mathbb{P}^{\Pi}\varphi \mid \mathbb{P}^{\Pi}p \mid f(p, \dots, p)$$

- a replaced by $\mathrm{a}^{\pi}$, $\pi$ is a path variable,
- $\mathbb{P}$ replaced by $\mathbb{P}^{\Pi}$, $\Pi$ is a set of path variables,
- $\mathbb{P}_{\sim \boldsymbol{p}}\ \varphi$ replaced by a set of rules $p ::= \mathbb{P}^{\Pi}\varphi \mid \mathbb{P}^{\Pi}p \mid f(p, \dots, p)$ and $p \sim p$

# Features of HyperPCTL*

Probabilistic Noninterference:

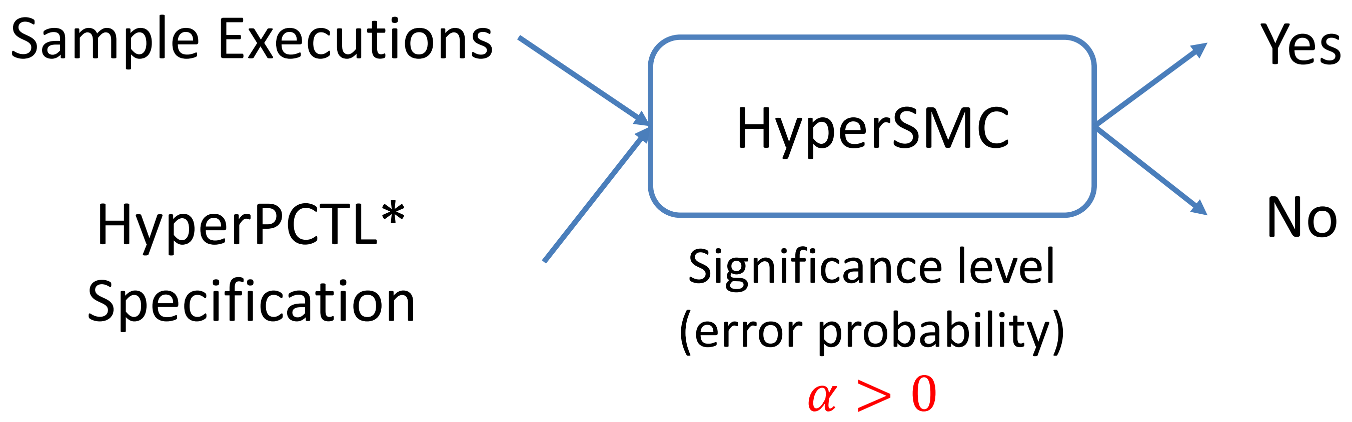
$$\mathbb{P}^{\pi_1}\big((l = 0)^{\pi_1} \rightarrow \mathbf{F}(h = 0)^{\pi_1}\big) \approx \mathbb{P}^{\pi_2}\big((l = 1)^{\pi_2} \rightarrow \mathbf{F}(h = 0)^{\pi_2}\big)$$

**Theorem 1: HyperPCTL*** is well-defined.

**Theorem 2: HyperPCTL*** is strictly more expressive than **PCTL*.**

# How to Check HyperPCTL*?

Statistical Model Checking (SMC): Statistically infer the correctness of HyperPCTL* specificaitons by sample system executions.




Advantages:

- Tolerate Unknowns
- Scalability
- Probabilistic Guarantee

For any pre-given $\alpha > 0$, the result is correct with probability at least $1 - \alpha$.

# Statistical Model Checking for HyperPCTL*

MICHIGAN STATE UNIVERSITY
Duke PRATT SCHOOL of ENGINEERING

HyperSMC is based on

1) Divide a specification into basic sub-specifications;
2) Check each of them with sufficient statistical accuracy.

Three kinds of basic sub-specifications:

- Probabilistic quantifications of **multiple** parallel paths $\mathbb{P}^{(\pi_1,\pi_2)}\varphi^{(\pi_1,\pi_2)} < p$
- **Nested** probabilistic path quantification $\mathbb{P}^{\pi_1}\left(\mathbb{P}^{\pi_2}\varphi^{(\pi_1,\pi_2)} < p_2\right) < p_1$
- **Joint** probabilities $(\mathbb{P}^{\Pi_1}\varphi_1, \mathbb{P}^{\Pi_2}\varphi_2) \in D$

We proposed NEW statistical inference methods to handle each of them!

# Case Studies 1

Dining $N$ Cryptographers

- Markov model of at least $2^N$ states
- We verified information security

$$\mathbb{P}^{\pi_1}\left(\Diamond(\neg \mathrm{S}_{ij}^{\pi_1} \wedge \Diamond \mathrm{P}^{\pi_1})\right) \approx_{\varepsilon} \mathbb{P}^{\pi_2}\left(\Diamond(\mathrm{S}_{ij}^{\pi_2} \wedge \Diamond \mathrm{P}^{\pi_2})\right) \approx_{\varepsilon} \mathbb{P}^{\pi_3}\left(\Diamond(\neg \mathrm{S}_{ij}^{\pi_3} \wedge \Diamond \mathrm{P}^{\pi_3})\right) \approx_{\varepsilon} \mathbb{P}^{\pi_4}\left(\Diamond(\mathrm{S}_{ij}^{\pi_4} \wedge \Diamond \mathrm{P}^{\pi_4})\right)$$

| Agents | $\delta$ | Acc. | No. Samples | Time (s) |
|---|---|---|---|---|
| 100 | 0.05 | 1.00 | 1.0e+03 | 0.91 |
| 100 | 0.1 | 1.00 | 5.2e+02 | 0.39 |
| 100 | 0.2 | 1.00 | 2.8e+02 | 0.14 |
| 1000 | 0.05 | **0.98** | 1.1e+03 | 3.27 |
| 1000 | 0.1 | 1.00 | 5.5e+02 | 1.52 |
| 1000 | 0.2 | 1.00 | 2.8e+02 | 0.69 |

[Significance level 0.01]

# Case Studies 2

Parallel Program with $N$ threads

- Markov model of $N!$ states.
- We verified probabilistic interference.

$$\mathbb{P}^{\pi_1}\left((l=0)^{\pi_1} \rightarrow \mathbf{F}(h=0)^{\pi_1}\right)$$
$$\approx \mathbb{P}^{\pi_2}\left((l=1)^{\pi_2} \rightarrow \mathbf{F}(h=0)^{\pi_2}\right)$$

| Threads | Significance | Acc. | No. Samples | Time (s) |
|---|---|---|---|---|
| 20 | 0.01 | 1.00 | 7.7e+02 | 0.49 |
| 20 | 0.001 | 1.00 | 7.6e+03 | 6.45 |
| 50 | 0.01 | 1.00 | 7.0e+02 | 0.48 |
| 50 | 0.001 | 1.00 | 6.8e+03 | 6.39 |
| 100 | 0.01 | 1.00 | 6.5e+02 | 0.54 |
| 100 | 0.001 | 1.00 | 6.6e+03 | 7.10 |

# Case Studies 3

GabFeed

- Chat server with encryption.
- We verified a time side-channel.

$$\mathbb{P}^{\pi_1}\left((\bigcirc \mathrm{S}_1^{\pi_1}) \Rightarrow (\Diamond^{\leq k} \mathrm{F}^{\pi_1})\right) \approx_{\varepsilon} \mathbb{P}^{\pi_2}\left((\bigcirc \mathrm{S}_2^{\pi_2}) \Rightarrow (\Diamond^{\leq k} \mathrm{F}^{\pi_2})\right)$$

| Horizon $k$ | $\varepsilon$ | Significance | Acc. | No. Samples | Time (s) |
|---|---|---|---|---|---|
| 60 | 0.05 | 0.01 | 1.00 | 5.5e+02 | 0.54 |
| 60 | 0.05 | 0.001 | 1.00 | 5.5e+03 | 5.76 |
| 60 | 0.1 | 0.01 | 1.00 | 6.1e+02 | 0.60 |
| 60 | 0.1 | 0.001 | 1.00 | 6.2e+03 | 7.16 |
| 90 | 0.05 | 0.01 | 1.00 | 3.7e+02 | 0.46 |
| 90 | 0.05 | 0.001 | 1.00 | 3.7e+03 | 4.94 |
| 90 | 0.1 | 0.01 | 1.00 | 4.1e+02 | 0.48 |
| 90 | 0.1 | 0.001 | 1.00 | 4.1e+03 | 5.37 |
| 120 | 0.05 | 0.01 | 1.00 | 3.8e+02 | 6.96 |
| 120 | 0.05 | 0.001 | 1.00 | 2.2e+03 | 11.24 |
| 120 | 0.1 | 0.01 | 1.00 | 3.8e+02 | 6.05 |
| 120 | 0.1 | 0.001 | 1.00 | 2.3e+03 | 9.46 |

# Case Studies 4

Randomized Cache Replacement Policy

- Least recently used (LRU) is not secure.
- We verified security.

$$\mathbb{P}^{\pi_1}(\bigcirc^{(N)}\Box^{\leq T}\,\mathrm{H}^{\pi_1}) > \mathbb{P}^{\pi_2}(\bigcirc^{(N)}\,\varphi^{\pi_2}) + \varepsilon$$

$$\varphi^{\pi_2} = \left(\mathrm{M}^{\pi_2} \wedge \bigcirc\mathrm{H}^{\pi_2} \wedge \ldots \wedge \bigcirc^{(T-1)}\,\mathrm{H}^{\pi_2}\right) \vee \ldots \vee \left(\mathrm{H}^{\pi_2} \wedge \ldots \wedge \bigcirc^{(T-2)}\,\mathrm{H}^{\pi_2} \wedge \bigcirc^{(T-1)}\,\mathrm{M}^{\pi_2}\right)$$

| Horizon $T$ | $\varepsilon$ | Significance | Acc. | No. Samples | Time (s) |
|---|---|---|---|---|---|
| 10 | 0.05 | 0.01 | 1.00 | 1.1e+02 | 0.13 |
| 10 | 0.05 | 0.001 | 1.00 | 1.0e+03 | 2.56 |
| 10 | 0.01 | 0.01 | 1.00 | 1.2e+02 | 0.14 |
| 10 | 0.01 | 0.001 | 1.00 | 1.2e+03 | 2.79 |
| 20 | 0.05 | 0.01 | 1.00 | 6.0e+02 | 1.49 |
| 20 | 0.05 | 0.001 | 1.00 | 6.2e+03 | 16.73 |
| 20 | 0.01 | 0.01 | 0.99 | 1.2e+03 | 2.97 |
| 20 | 0.01 | 0.001 | 1.00 | 1.1e+04 | 28.99 |

# Thank you

**Code**: https://gitlab.oit.duke.edu/cpsl/hpctls