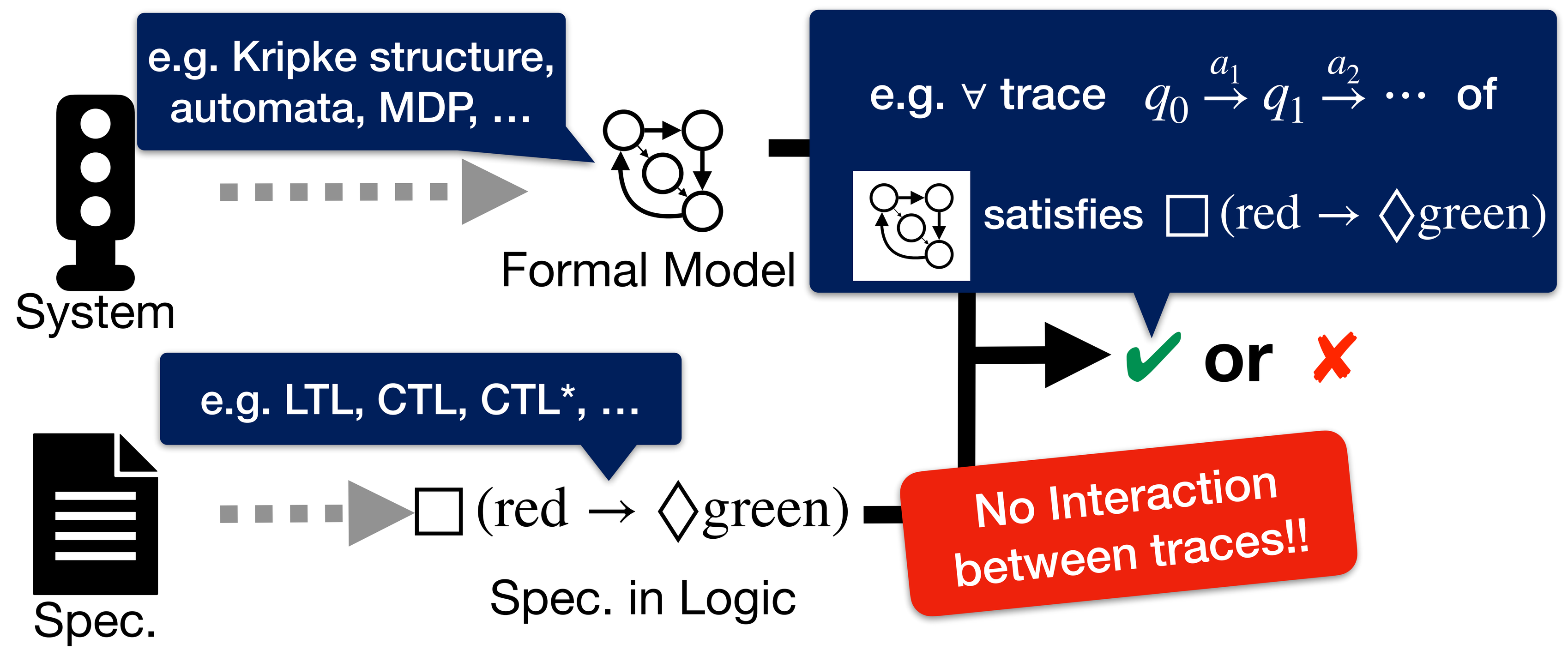
Hyper Parametric Timed CTL

Masaki Waga[1] & Étienne André[2] / Kyoto University[1], Université Sorbonne Paris Nord[2]

Q. Can we model check parametric timed automata (PTAs) against hyperproperties?

A. Yes, for an appropriate subclass. Idea: Reduction to model checking against PTCTL

Model Checking of Trace Properties

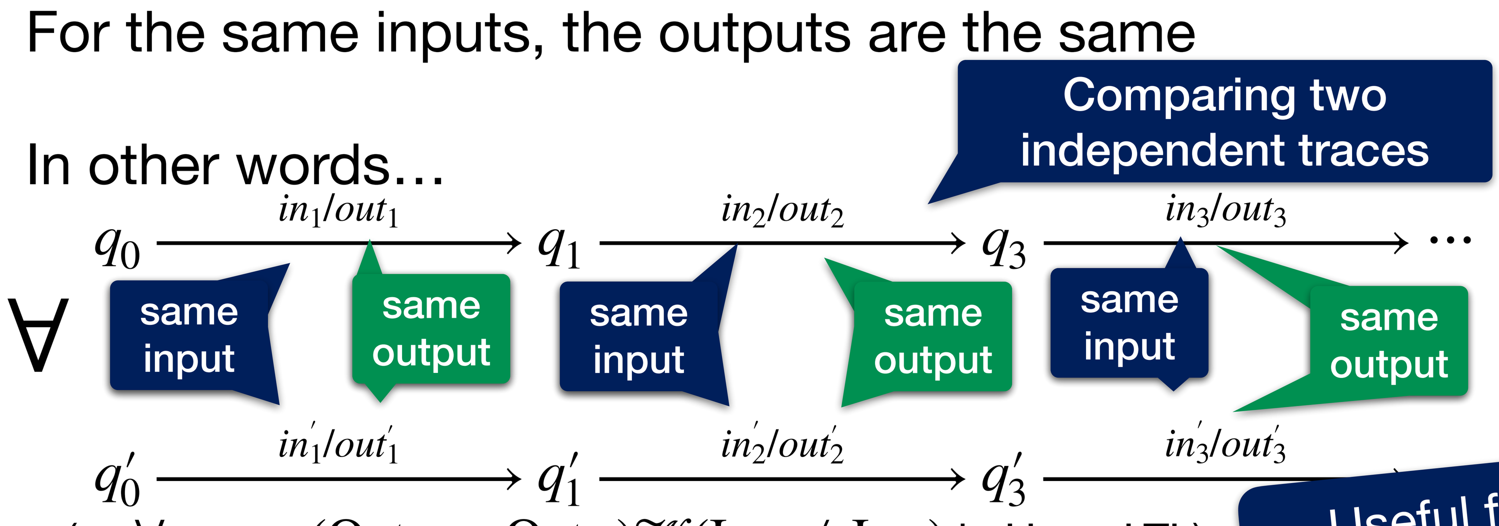


Untimed Hyperproperties, e.g. HyperCTL*

Example (Observational determinism)

For the same inputs, the outputs are the same

In other words…



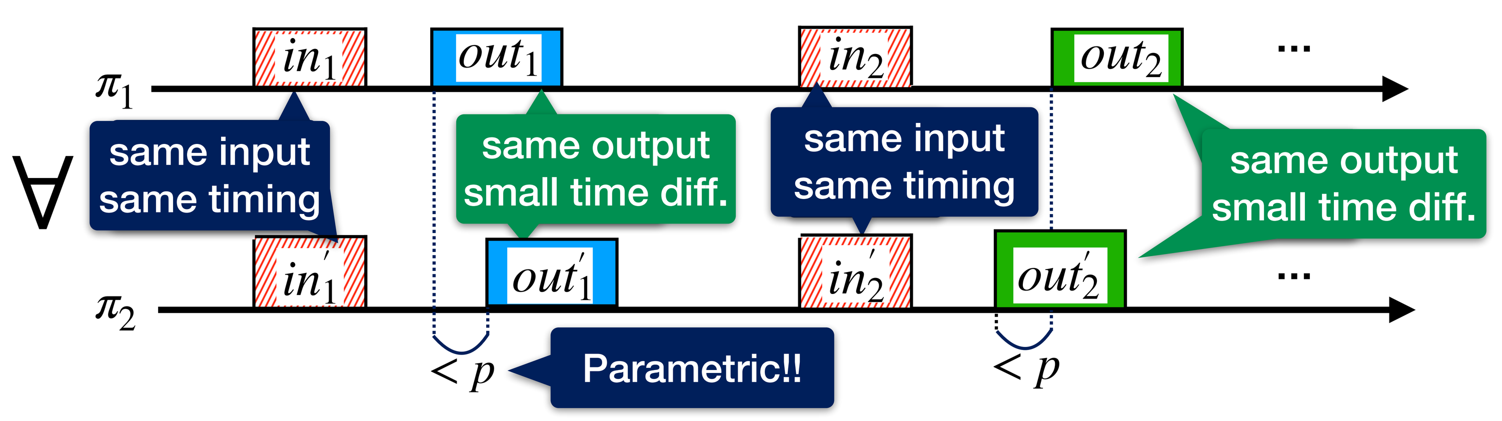
(or $\forall \pi_1, \pi_2 . (\mathrm{Out}_{\pi_1} = \mathrm{Out}_{\pi_2}) \mathscr{W} (\mathrm{In}_{\pi_1} \neq \mathrm{In}_{\pi_2})$ in HyperLTL)

Useful for Security, Fairness, Robustness, …

(Ext-)HyperPTCTL: HyperCTL + Time/Parameters + Additional Predicates [Contribution]

Example (Parametric timed observational determinism)

Observational determinism with *small timing deviation* of outputs



(simplified)

$\forall \pi_1, \pi_2 . (\forall i . \#(\mathrm{Out}^i_{\pi_1}) = \#(\mathrm{Out}^i_{\pi_2}) \Rightarrow |\mathrm{LAST}(\mathrm{Out}^i_{\pi_1}) - \mathrm{LAST}(\mathrm{Out}^i_{\pi_2})| < p) \mathscr{W} (\mathrm{In}_{\pi_1} \neq \mathrm{In}_{\pi_2})$

HyperPTCTL — Proposition on locations

temporal level:

$$\varphi ::= \top \mid \sigma_\pi \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists \pi_1, \pi_2, \dots, \pi_n . \varphi \mathcal{U}_{\bowtie \gamma} \varphi \mid \forall \pi_1, \pi_2, \dots, \pi_n . \varphi \mathcal{U}_{\bowtie \gamma} \varphi$$

top level:

$$\psi ::= \varphi \mid p \bowtie lt_{\geq 0} \mid \neg\psi \mid \psi \vee \psi \mid \tilde{\exists} p\, \psi$$

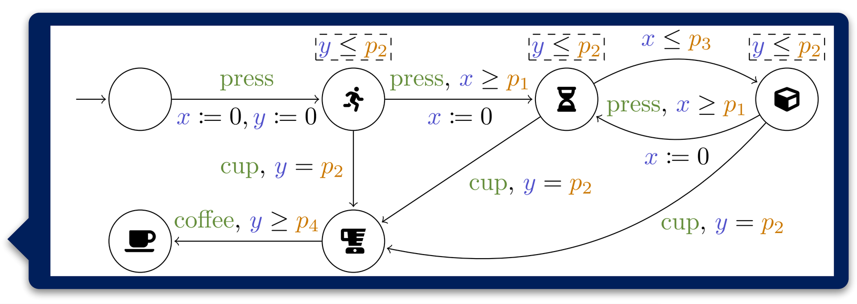
Ext-HyperPTCTL

$$\varphi ::= \top \mid \sigma_\pi \mid LAST(\sigma_\pi) - LAST(\sigma_\pi) \bowtie lt \mid cnt_{\geq 0} \bowtie d \mid (cnt \bmod N) \bowtie d \mid \cdots$$

Problem Definition

Input:

- Parametric timed automaton $\mathscr{A}$



- Ext-HyperPTCTL formula φ

$\forall \pi_1, \pi_2 (☕_{\pi_1} = ☕_{\pi_2}) \mathscr{W}_{[0,p)} (\mathrm{press}_{\pi_1} \neq \mathrm{press}_{\pi_2})$

$\{v \mid v(p_4) \leq v(p)\}$

Synthesis: Synthesize param. val. v s.t. $v(\mathscr{A}) \models v(\varphi)$

Exists!

Model Checking: Decide the existence of such v

Implementation (HyPTCTLChecker) & Experiments

- Implemented the reduction to IMITATOR
- Reduction slightly differs from theoretical one e.g. IMITATOR's discrete var. not encoding w/ loc.
- The reduction is almost immediate → Report the result of synthesis with IMITATOR

Prop. (ψ)	PTA ($\mathcal{A}$)	$\lvert L\rvert$	$\lvert\mathbb{C}\rvert$	$\lvert\mathbb{P}\rvert_\psi$	$\lvert\mathbb{P}\rvert_\mathcal{A}$	$\lvert\mathcal{V}\rvert$	Time [sec.]
Deviation	ClkGen	2	1	1	1	2	4.116
Opacity	Coffee	6	2	0	3	2	0.723
Opacity	STAC1:n	8	2	0	2	2	0.178
Opacity	STAC4:n	9	2	0	5	2	< 0.001
Unfair	FIFO	63	2	0	4	2	71.955
Unfair	Priority	72	2	0	4	2	6.855
Unfair	R.R.	81	3	0	4	2	12550.979
RobOND	Coffee	6	2	1	3	2	3.182
RobOND	WFAS^1_0	24	4	1	0	2	1.665
RobOND	WFAS^2_0	24	4	1	0	2	2.570
RobOND	WFAS_1	24	4	1	1	2	67.644
RobOND	WFAS_2	24	4	1	2	2	1332.310
RobOND	ATM	7	2	1	0	2	T.O.
RobOND	ATM′	5	2	1	0	2	4179.197
EF_2	Coffee	6	2	1	0	2	0.034
EF_3	Coffee	6	2	1	0	3	159.541
EF_4	Coffee	6	2	1	0	4	T.O.

Idea of Our Semi-Algorithm: Reduction to PTCTL Model Checking

Idea of the Reduction

1. Ext-HyperPTCTL→HyperPTCTL: encode w/ PTAs
2. HyperPTCTL → PTCTL: self-composition of PTAs

Note: PTCTL synthesis is in general undecidable
→ We only have semi-algorithm

1. Ext-HyperPTCTL→HyperPTCTL

Slogan: Restrict the terms so that:

- Predicates' truth values are updated only at transitions
- Only finite (discrete) counting is sufficient

2. HyperPTCTL→PTCTL

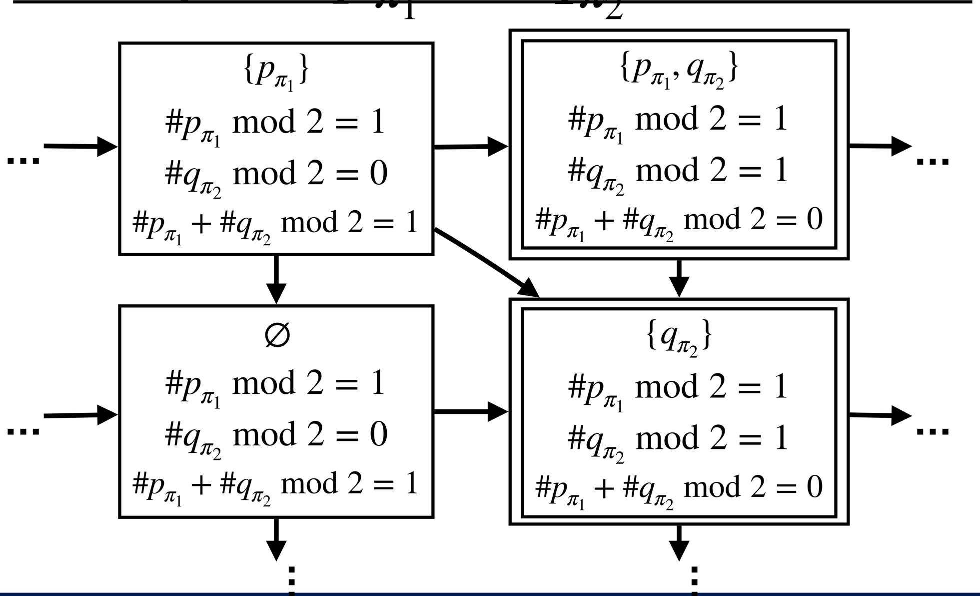
- Reduction s.t. traces $\pi_1, \pi_2, \dots, \pi_n$ of $\mathscr{A}$ is captured by trace $\pi_1 \| \pi_2 \| \cdots \| \pi_n$ of $\mathscr{A} \| \mathscr{A} \| \cdots \| \mathscr{A}$

Limitation/Challenge — Limited, but still likely useful

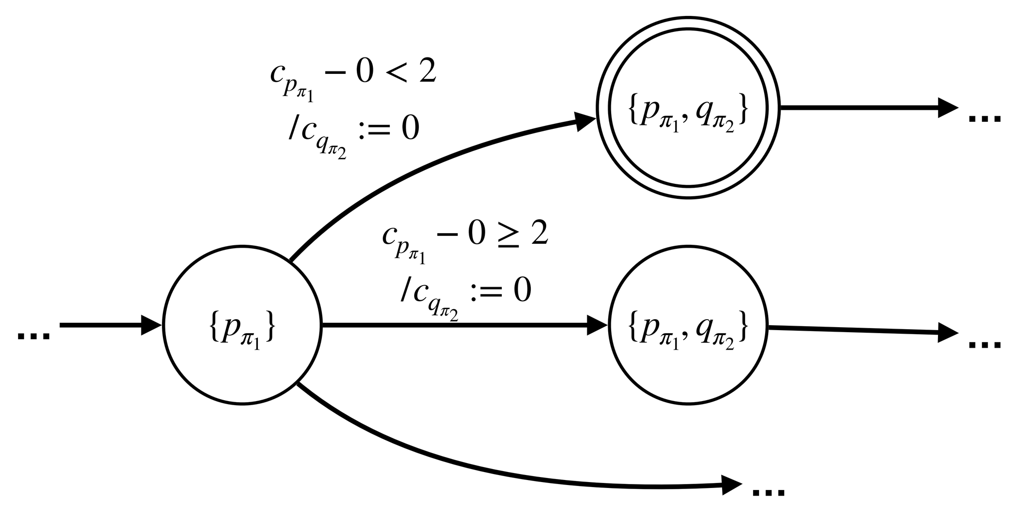
- Complement is impossible → Focus on nest-free fragment
- "Zero-time behavior" is tricky

Multiple "simultaneous" jumps
e.g. $\mathrm{jump}_1 \| \mathrm{jump}'_1 \rightarrow \mathrm{jump}_2$
vs. $\mathrm{jump}_1 \rightarrow \mathrm{jump}_2 \| \mathrm{jump}'_1$

Example: $\#p_{\pi_1} + \#q_{\pi_2} \bmod 2 = 0$



Example: $LAST(p_{\pi_1}) - LAST(q_{\pi_2}) < 2$



Explicit Transition Ordering

Idea: Path valuation := (paths, order)

Prevent multiple possible ordering of jumps

$$\pi_1 = (l_0, \nu_0) \xrightarrow{\mathrm{jump}_1} (l_1, \nu_1) \xrightarrow{\mathrm{jump}_2} (l_2, \nu_2)$$

$$\pi_2 = (l'_0, \nu'_0) \xrightarrow{\mathrm{jump}'_1} (l'_1, \nu'_1) \xrightarrow{\tau = 2.4} (l'_2, \nu'_2)$$

Transition ordering: $\mathrm{jump}_1 \sim \mathrm{jump}'_1 \prec \mathrm{jump}_2$