Monitoring Hyperproperties

Bernd Finkbeiner
Reactive Systems Group
Saarland University

based on joint work with
Christopher Hahn, Marvin Stenger, and Leander Tentrup

Siracusa
September 3, 2018

Hyperproperties

Clarkson&Schneider '10:
Hyperproperty H: a set of sets of traces

System K satisfies H iff $\mathit{Traces}(K) \in H$.

Many information-flow properties can be formalized as hyperproperties.

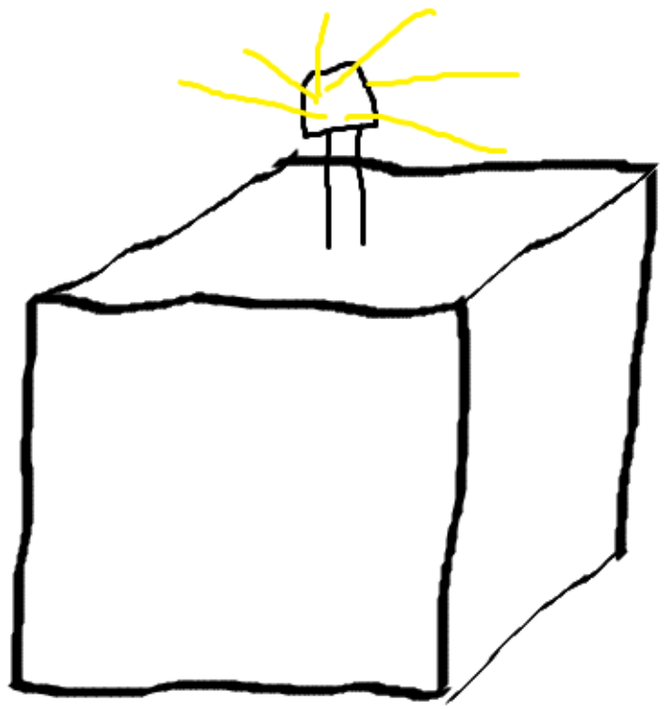
Noninterference as hyperproperty:

$$\{T \subseteq 2^{\mathit{Traces}} \mid \forall t, t' \in T : t =_{I_{\mathit{public}}} t' \;\Rightarrow\; t =_{O_{\mathbf{public}}} t'\}$$

Trace equality, observational determinism, noninference, declassification, ...

A simple information-flow property

"All executions have the light on at the same time."



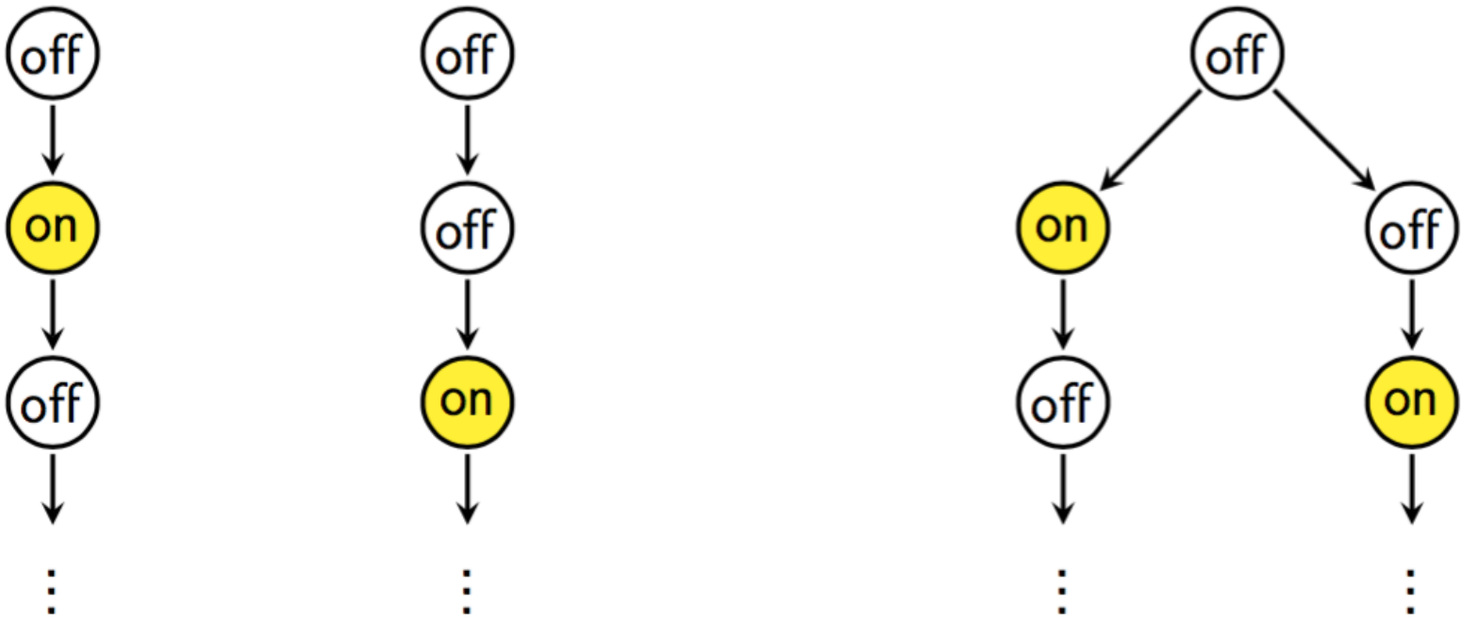
"For all pairs of traces and all points in time, the light is "on" on the one trace iff it is "on" on the other trace."

LTL ?

Syntax: $\varphi ::= a_\pi \mid \mathsf{X}\psi \mid \mathsf{G}\psi \mid \mathsf{F}\psi \mid \psi\,\mathsf{U}\,\psi$

Semantics: $K \models \varphi$ iff $\mathit{Traces}(K) \subseteq \mathit{Traces}(\varphi)$

"All executions have the light on at the same time."

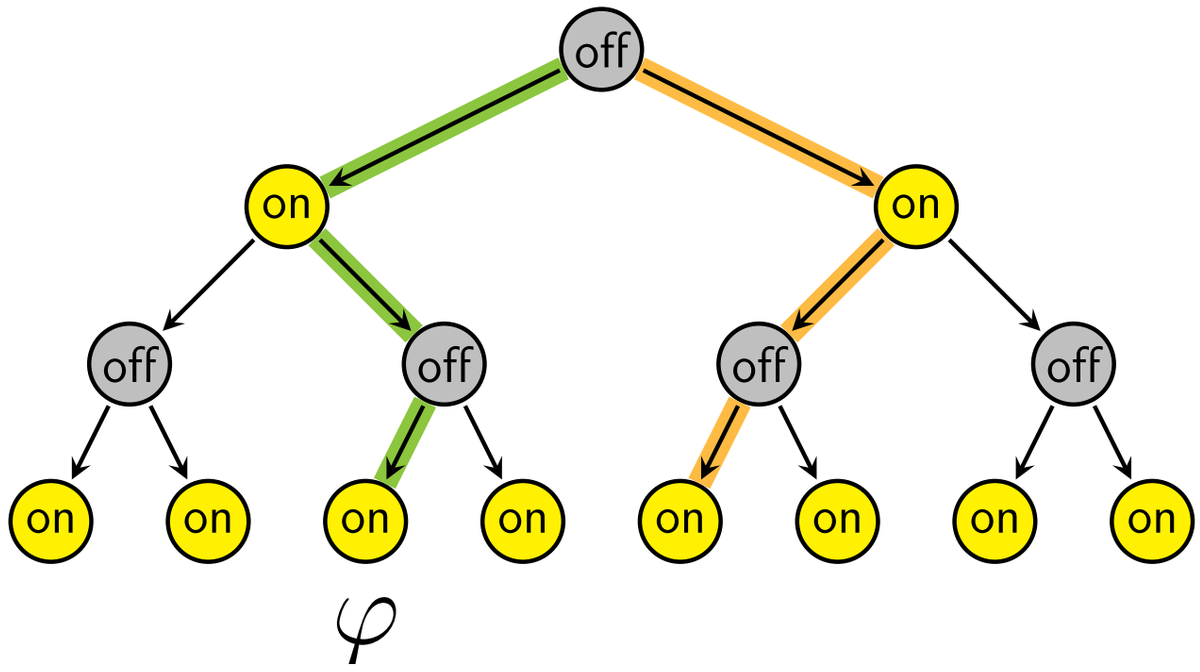


CTL* ?

Syntax: $\varphi ::= a \mid \mathsf{A}\varphi \mid \mathsf{E}\varphi \mid \mathsf{X}\varphi \mid \mathsf{G}\varphi \mid \varphi \mathsf{U} \varphi \mid \ldots$

Semantics: $K \models \mathsf{A}\varphi$ iff for all $p \in \mathsf{Paths}(K) : p \models \varphi$

"All executions have the light on at the same time." $\mathsf{AA}\varphi$?



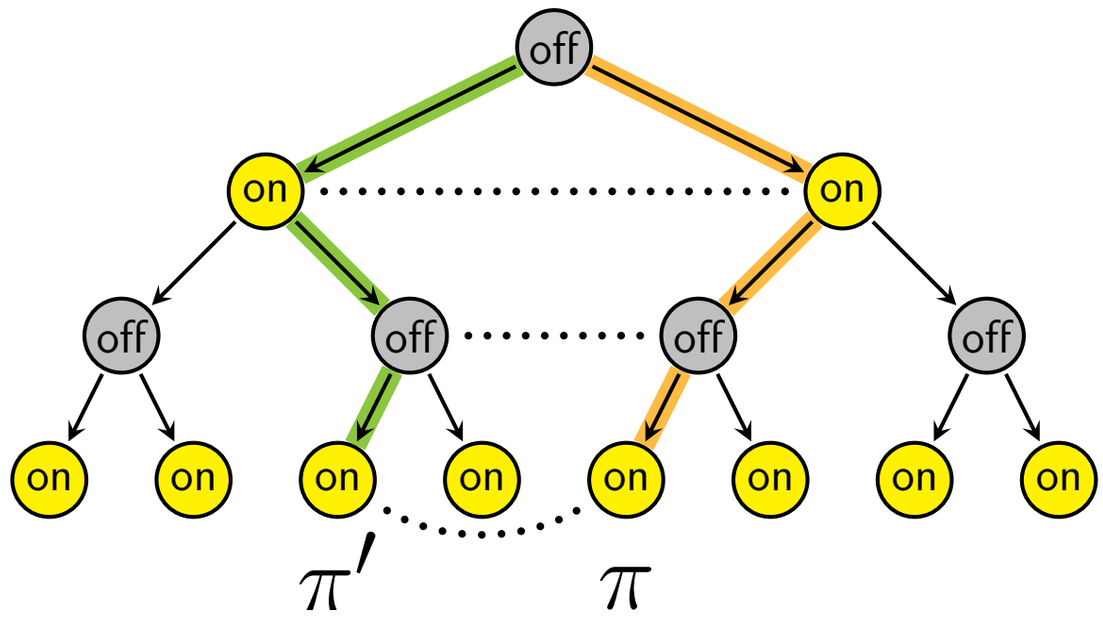
HyperLTL

Quantifiers with trace variables: $\forall\pi.\varphi$ [$\exists\pi.\varphi$]

$\varphi ::= \forall\pi.\varphi \mid \exists\pi.\varphi \mid \psi$
$\psi ::= a_\pi \mid \mathsf{X}\psi \mid \mathsf{G}\psi \mid \mathsf{F}\psi \mid \psi\,\mathsf{U}\,\psi \mid \neg\psi$

"All executions have the light on at the same time."

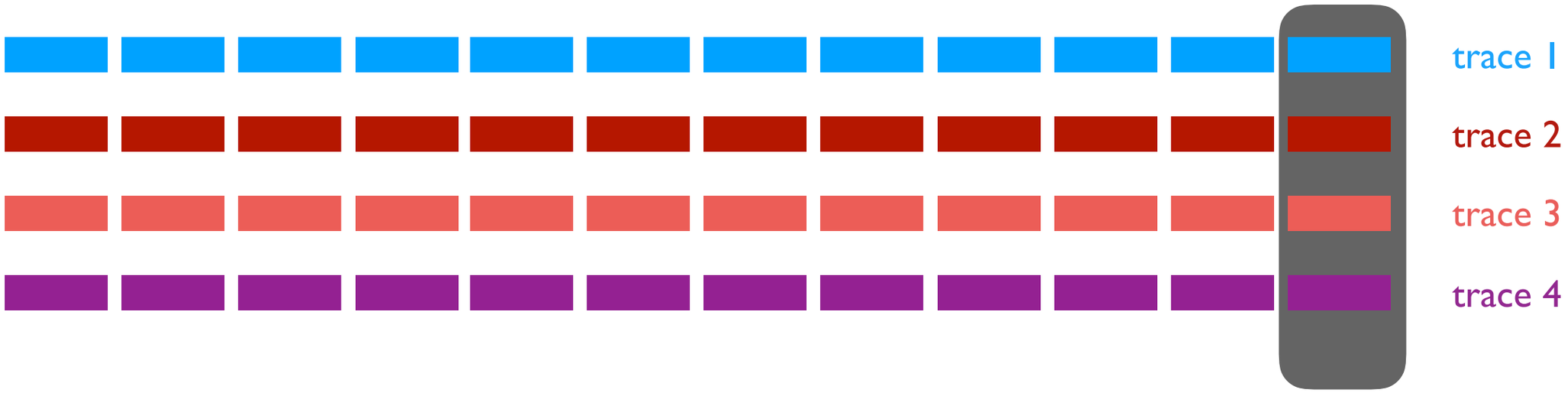
$\forall\pi.\forall\pi'.\ \mathsf{G}(\mathsf{on}_\pi \leftrightarrow \mathsf{on}_{\pi'})$



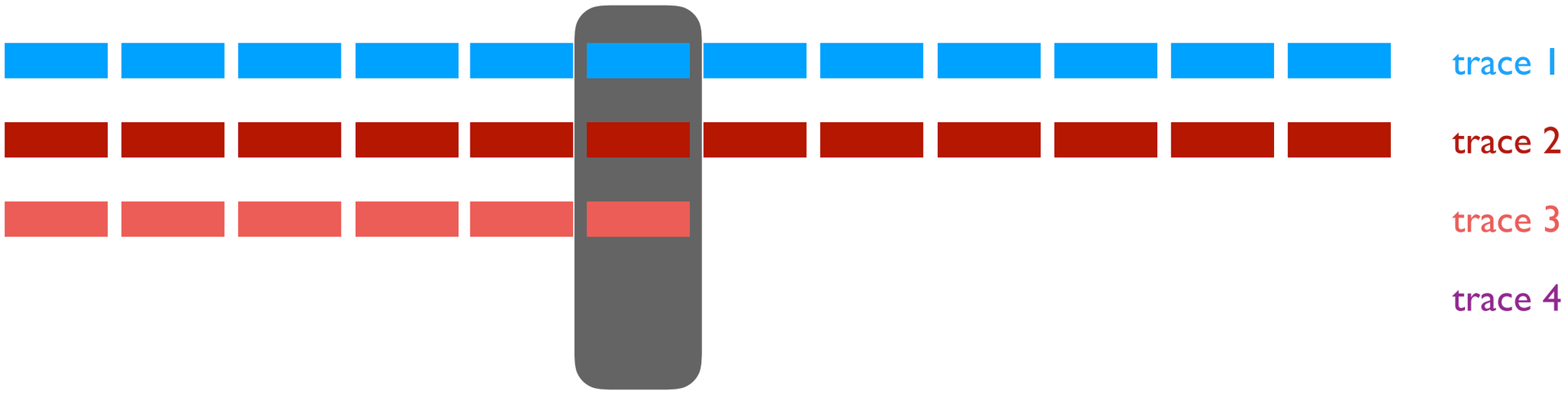
Clarkson, F., Koleini, Micinski, Rabe, Sánchez. Temporal Logics for Hyperproperties. **POST '14**

Monitoring

- Parallel monitoring

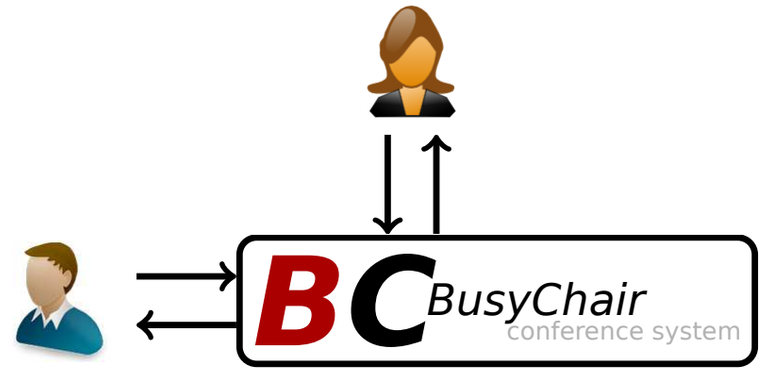


- Sequential monitoring



Parallel Monitoring

Example: conference management system with **author** and **pc** traces

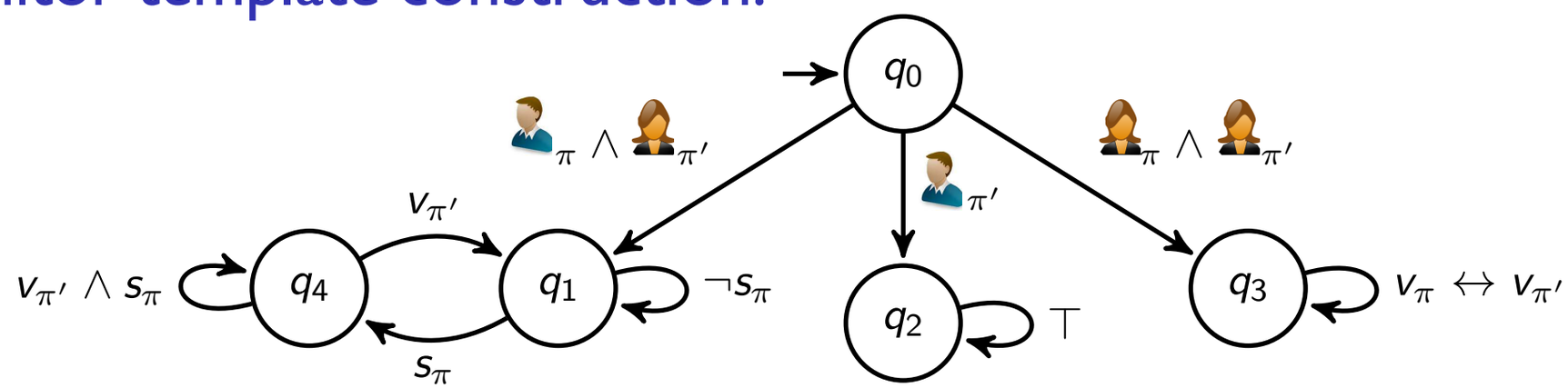


„No paper submission is lost“:

- every submission (s) is visible (v) to every PC member
- when comparing two pc traces, they must agree on v

$$\forall \pi.\, \forall \pi'.\, (\text{pc}_\pi \wedge \text{pc}_{\pi'}) \rightarrow \mathsf{X}\ \mathsf{G}\ (s_\pi \rightarrow \mathsf{X}\ v_{\pi'}) \wedge (\text{pc}_\pi \wedge \text{pc}_{\pi'}) \rightarrow \mathsf{X}\ \mathsf{G}\ (v_\pi \leftrightarrow v_{\pi'})$$

Monitor template construction:



The monitor runs in parallel over all permutations of the traces.

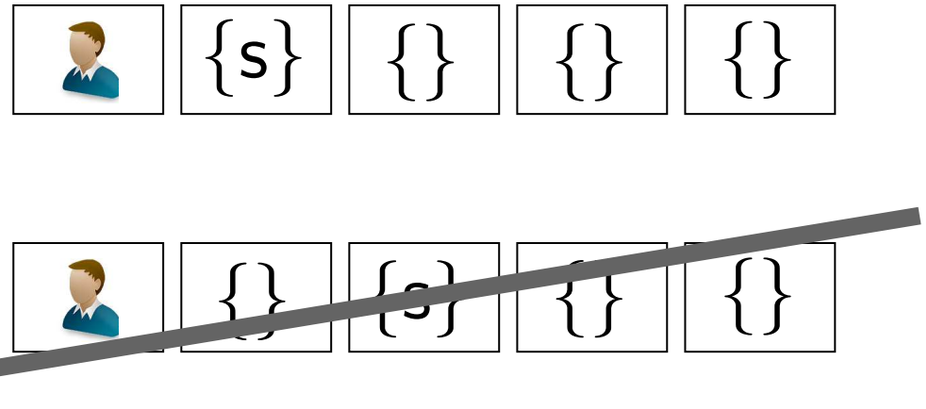
Sequential Monitoring

- Naive approach:
  - Store growing set of traces
  - Repeatedly apply parallel monitoring

- Memory explosion!
  - Storing every trace leads to unbounded memory consumption

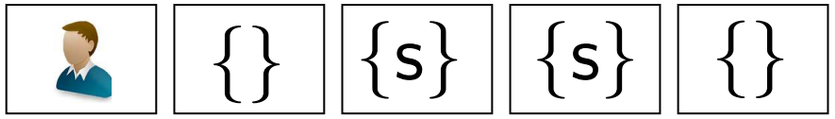
- Key: effective memory management
  - trace analysis: discard traces that are dominated by other traces
  - specification analysis: exploit symmetry, transitivity, reflexivity
  - trace set representation: organize traces by prefix

Trace Analysis: Example



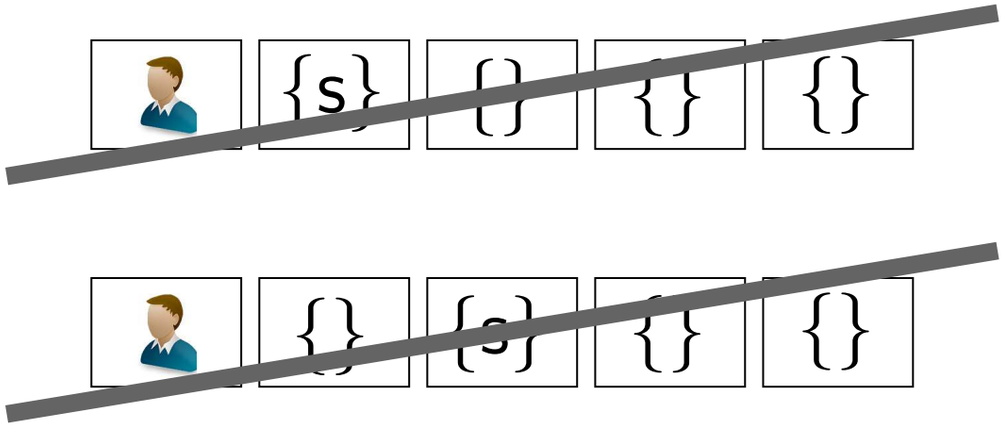
An author submits a paper

Another author submits a paper



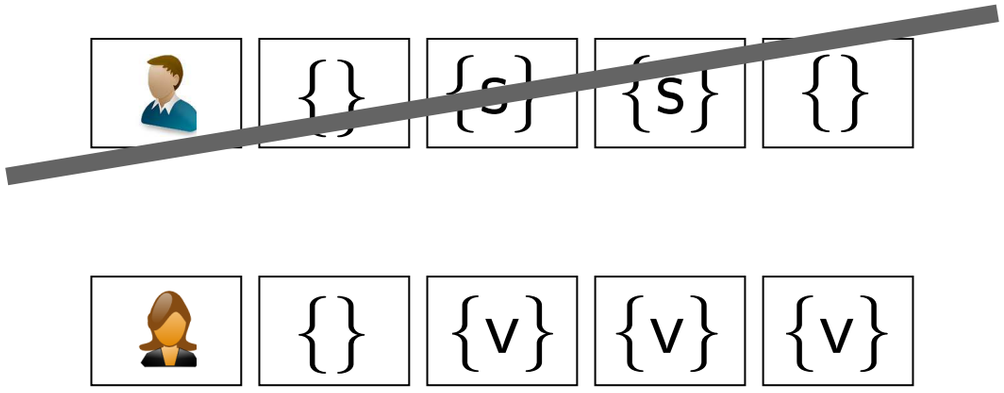
An author submits two papers

Trace Analysis: Example



An author submits a paper

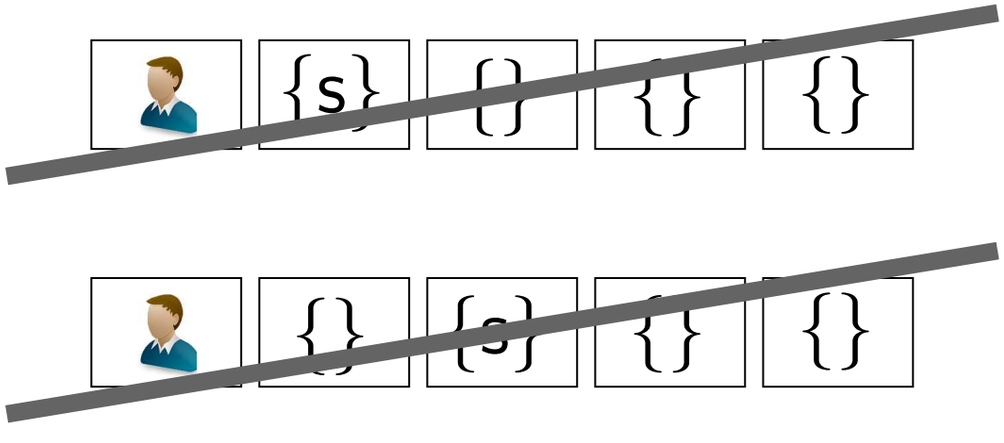
Another author submits a paper



An author submits two papers

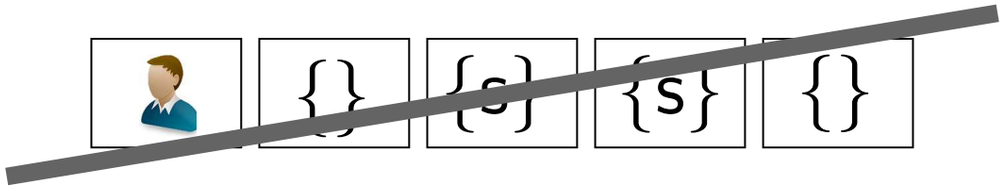
A pc member observes three submissions

Trace Analysis: Example

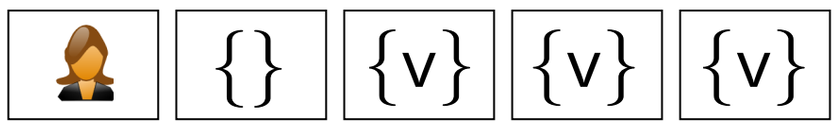


An author submits a paper

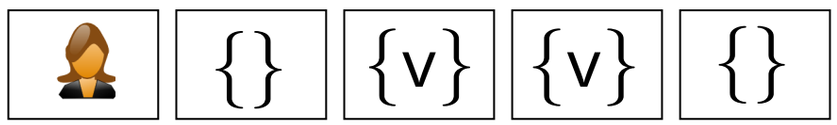
Another author submits a paper



An author submits two papers



A pc member observes three submissions



A pc member observes two submissions

Trace Analysis: Dominance

Given

- a universal HyperLTL formula φ,
- two traces t and t', and
- a monitor automaton M_φ,

t' dominates t if and only if for all trace variables $\boldsymbol{\pi}$, $L(M_\varphi[t'/\boldsymbol{\pi}]) \subseteq L(M_\varphi[t/\boldsymbol{\pi}])$.

Traces that are dominated by other traces in the stored trace set can safely be **removed** from the trace set.

Specification Analysis: Symmetry example

Observational determinism

$$\forall \pi . \forall \pi' . (O_\pi = O_{\pi'}) \, W \, (I_\pi \neq I_{\pi'})$$

is symmetric, we can therefore **omit symmetric monitor instantiations.**

To prove the symmetry of observational determinism,
we establish the validity of

$$\begin{aligned} \forall \pi . \forall \pi' . (O_\pi = O_{\pi'}) \, W \, (I_\pi \neq I_{\pi'}) \\ \leftrightarrow (O_{\pi'} = O_\pi) \, W \, (I_{\pi'} \neq I_\pi) \end{aligned}$$

Specification Analysis: Transitivity example

Output equality

$$\forall \pi . \forall \pi' . O_\pi = O_{\pi'}$$

is symmetric and transitive,
we therefore only need to store **a single trace.**

To prove the transitivity of output equality,
we establish the validity of

$$\forall \pi . \forall \pi' . \forall \pi'' . (O_\pi = O_{\pi'}) \wedge (O_{\pi'} = O_{\pi''}) \rightarrow (O_\pi = O_{\pi''})$$

Specification Analysis: Reflexivity example

Observational determinism

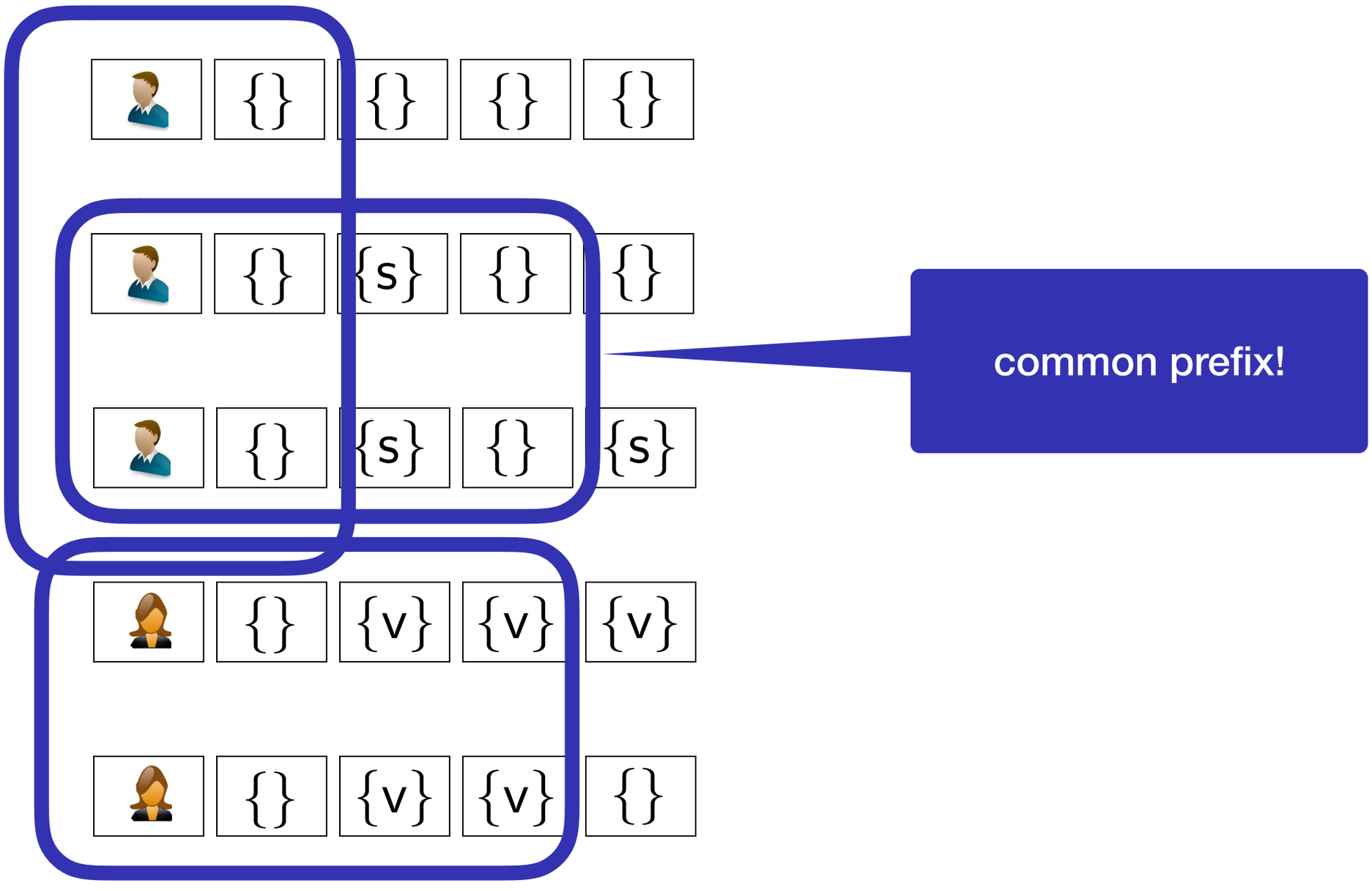
$$\forall \pi . \forall \pi' . (O_\pi = O_{\pi'}) \, W \, (I_\pi \neq I_{\pi'})$$

is reflexive, we can therefore **omit the reflexive monitor instantiations.**

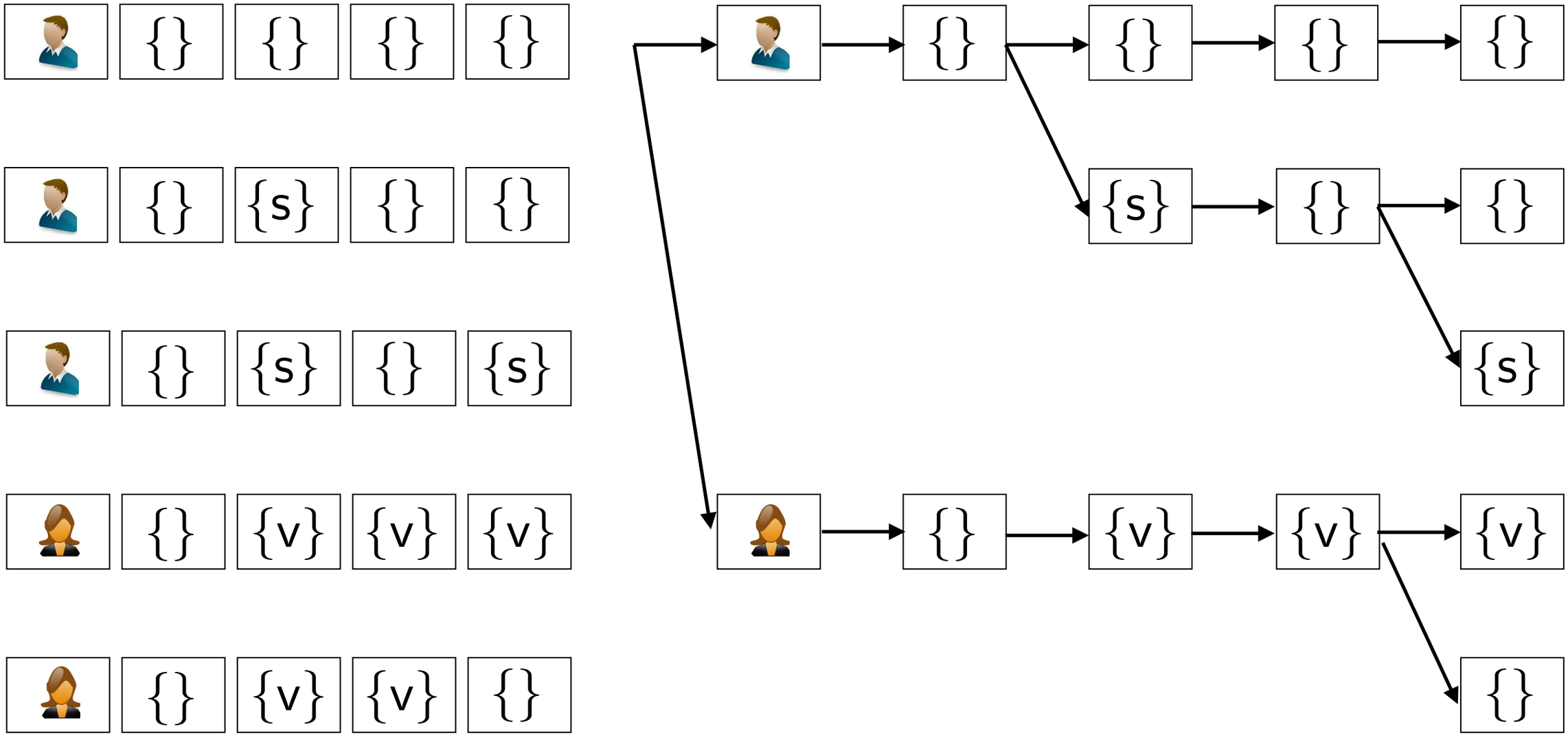
To prove the reflexivity of observational determinism,
we establish the validity of

$$\forall \pi . (O_\pi = O_\pi) \, W \, (I_\pi \neq I_\pi)$$

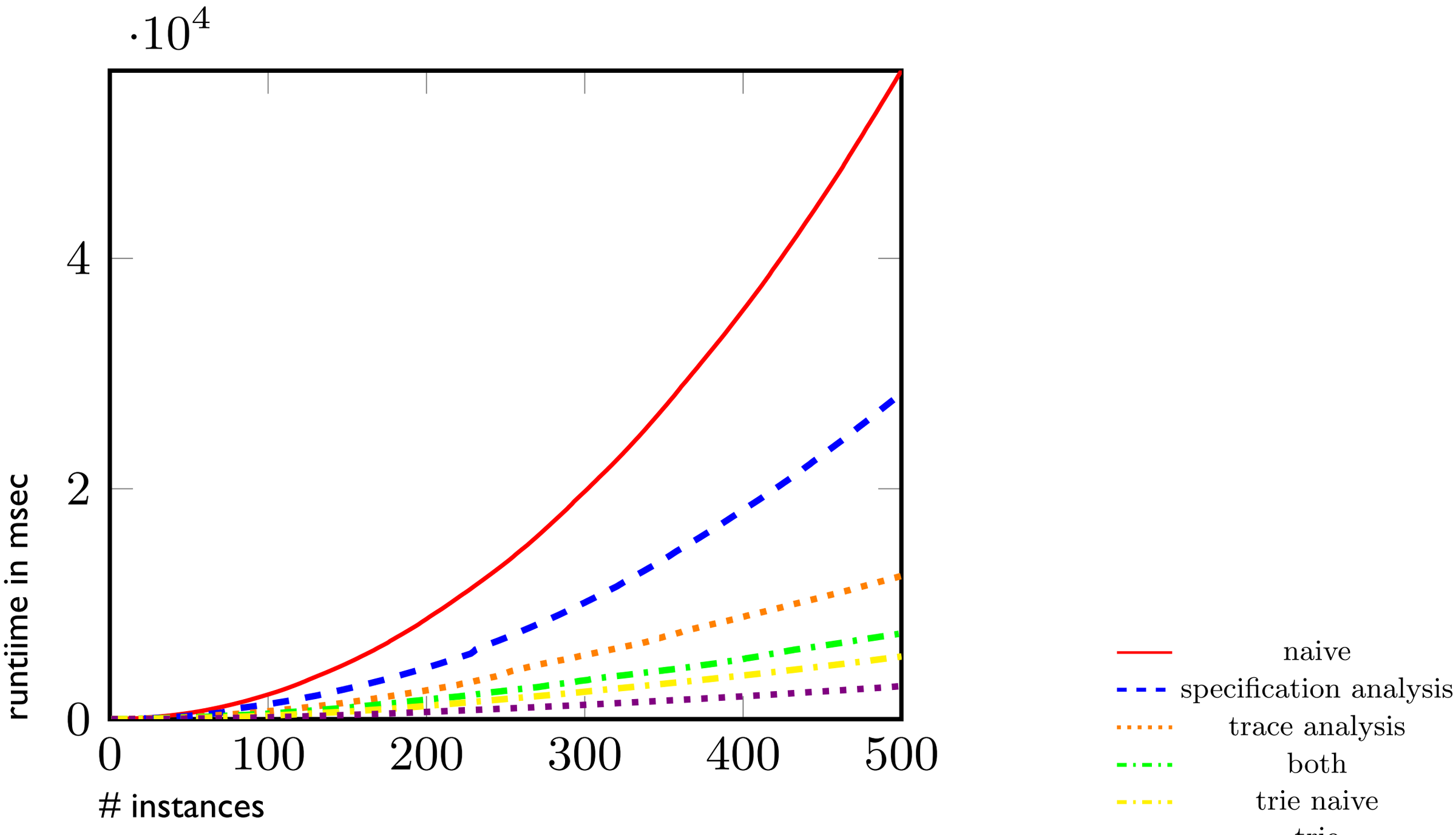
Trace Set Representation: Common Prefixes



Trace Set Representation: Prefix trees

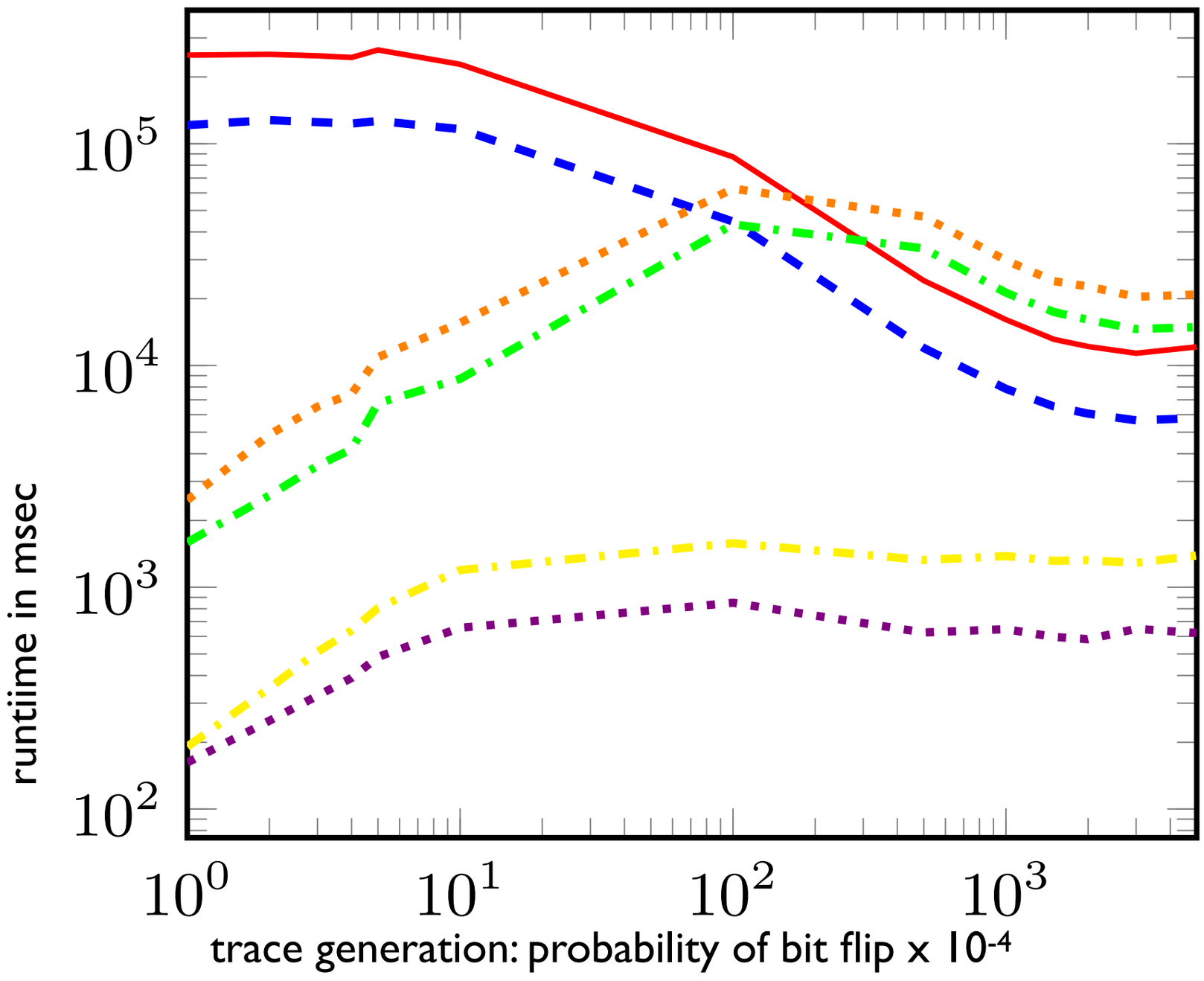


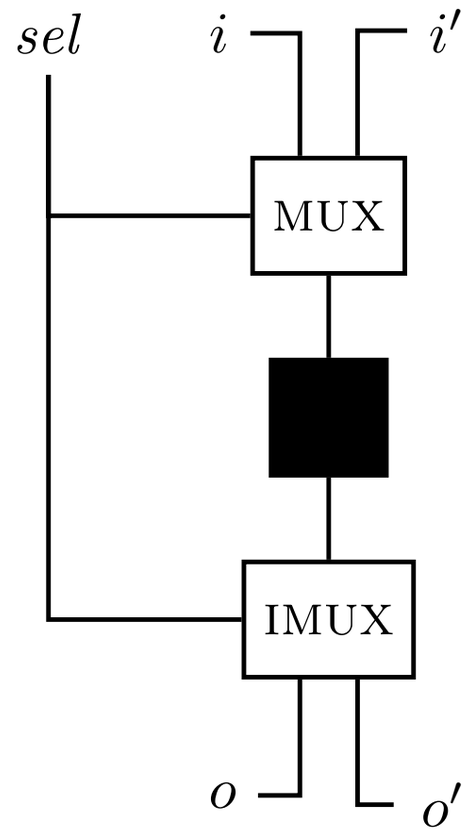
Experiments (error-resilient encoder)

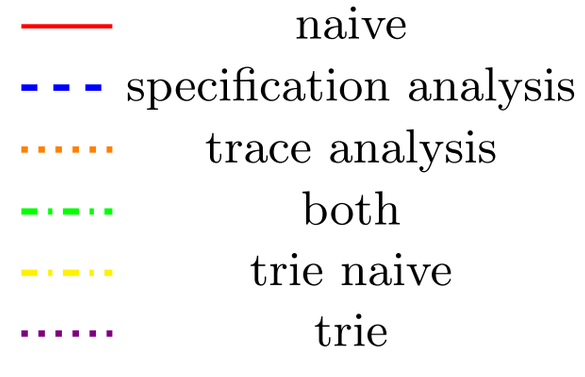


Experiments (black-box circuit)

$$\forall \pi_1 \forall \pi_2 . \, (\boldsymbol{o}_{\pi_1} = \boldsymbol{o}_{\pi_2}) \, \mathcal{W} \, (\bar{\boldsymbol{i}}_{\pi_1} \neq \bar{\boldsymbol{i}}_{\pi_2})$$







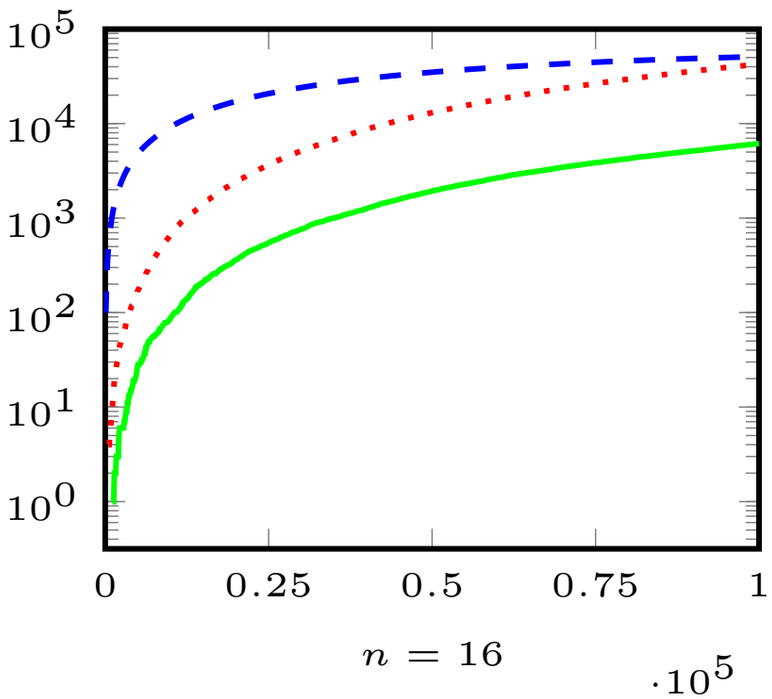
Experiments: Specification Analysis

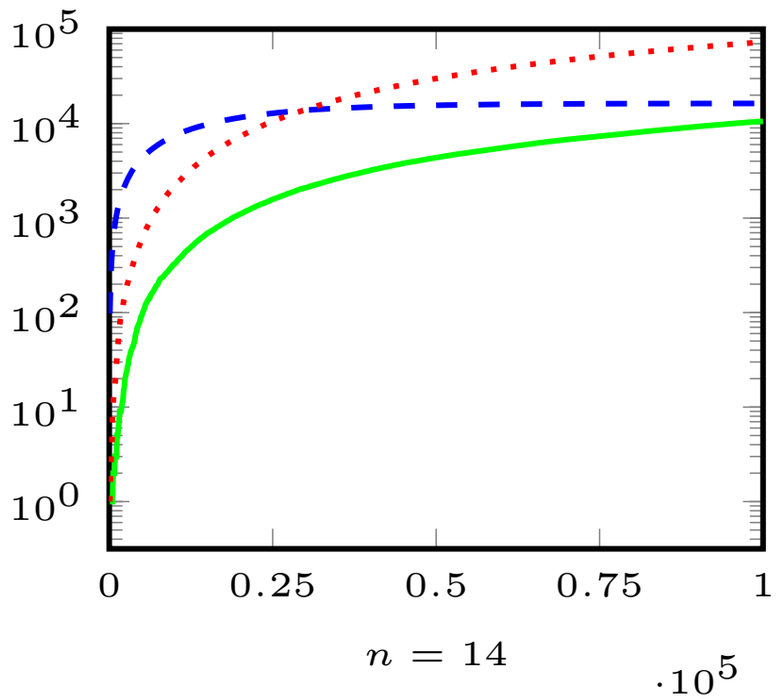
		symm	trans	refl
ObsDet1	$\forall\pi.\forall\pi'.\ \mathsf{G}\ (I_\pi = I_{\pi'}) \rightarrow \mathsf{G}\ (O_\pi = O_{\pi'})$	✓	✗	✓
ObsDet2	$\forall\pi.\forall\pi'.\ (I_\pi = I_{\pi'}) \rightarrow \mathsf{G}\ (O_\pi = O_{\pi'})$	✓	✗	✓
ObsDet3	$\forall\pi.\forall\pi'.(O_\pi = O'_\pi)\mathcal{W}(I_\pi \neq I'_\pi)$	✓	✗	✓
QuantNoninf	$\forall\pi_0 \ldots \forall\pi_c.\ \neg((\bigwedge_i I_{\pi_i} = I_{\pi_0}) \wedge \bigwedge_{i\neq j} O_{\pi_i} \neq O_{\pi_j})$	✓	✗	✓
EQ	$\forall\pi.\forall\pi'.\mathsf{G}\ (a_\pi \leftrightarrow a_{\pi'})$	✓	✓	✓
ConfMan	$\forall\pi\forall\pi'.\big((\neg pc_\pi \wedge pc_{\pi'}) \rightarrow \mathsf{XG}\ (s_\pi \rightarrow \bigcirc v_{\pi'})\big)$ $\wedge\big((pc_\pi \wedge pc_{\pi'}) \rightarrow \mathsf{XG}\ (v_\pi \leftrightarrow v_{\pi'})\big)$	✗	✗	✗

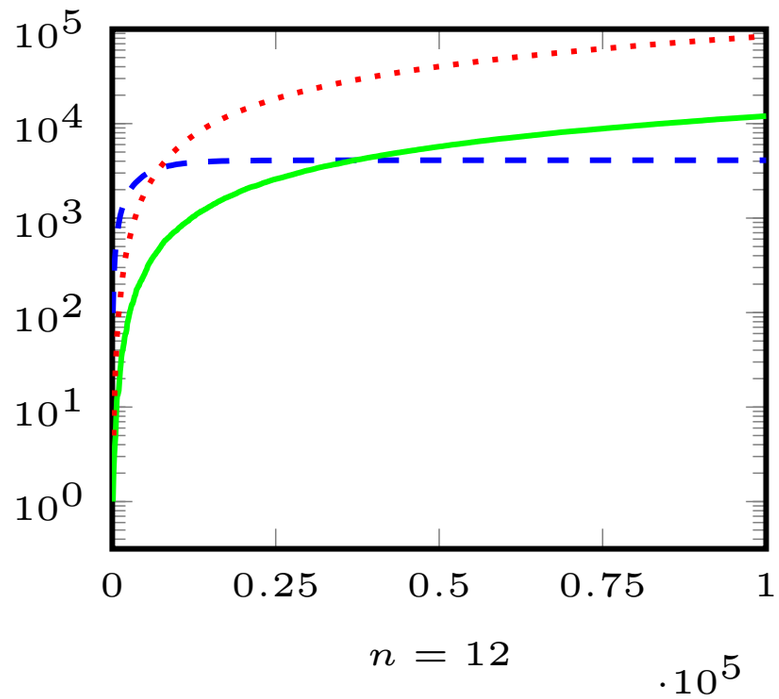
satisfiability checks < 2sec

Experiments: Trace Analysis

$$\forall \pi.\, \forall \pi'.\, \mathsf{G}_{<n}(I_\pi = I_{\pi'}) \rightarrow \mathsf{G}_{<n+c}(O_\pi = O_{\pi'})$$







- number of violations
- number of stored traces
- number of pruned traces

Conclusions

- Sequential monitoring of hyperproperties more difficult than parallel monitoring: in principle **unbounded memory**

- Effective optimizations:
  - Trace dominance: substantially reduces number of **stored traces**
  - Symmetry: omits at least **half** of the monitor instantiations
  - Transitivity: reduces the instantiations **to two**
  - Reflexivity: omits the **reflexive** monitor instantiations
  - Prefix tree representation: dramatically reduces memory & runtime

- Tools:
  - RVHyper: runtime verification tool for hyperproperties
  - EAHyper: satisfiability solver for hyperproperties

https://www.react.uni-saarland.de/tools/

Bibliography

- B. Finkbeiner, Ch. Hahn, M. Stenger, L. Tentrup. Monitoring Hyperproperties, Runtime Verification: 17th International Conference, RV 2017. Full version: https://arxiv.org/abs/1807.00758

- B. Finkbeiner, Ch. Hahn, M. Stenger, L. Tentrup. RVHyper: A Runtime Verification Tool for Temporal Hyperproperties. 24th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2018.

- B. Finkbeiner, Temporal Hyperproperties, Bulletin of the EATCS, volume 123, 2017

- M. R. Clarkson, B. Finkbeiner, M. Koleini, K. K. Micinski, M. N. Rabe, C. Sánchez. Temporal logics for hyperproperties. In International Conference on Principles of Security and Trust (pp. 265-284), POST 2014.

- B. Finkbeiner, Ch. Hahn. Deciding hyperproperties. 27th International Conference on Concurrency Theory, CONCUR 2016

- B. Finkbeiner, Ch. Hahn, M. Stenger. EAHyper: Satisfiability, Implication, and Equivalence Checking of Hyperproperties. International Conference on Computer Aided Verification, CAV 2017.