Concurrent Hyperproperties

Ernst-Rüdiger Olderog

joint work with Bernd Finkbeiner

September 2023

Carl von Ossietzky
Universität
Oldenburg

Hyperproperties

Aim: analyze executions of systems

- trace properties = sets of execution traces

  ... express properties of individual executions,

  e.g. safety:

  $$\forall \pi. \square(out_\pi \neq bad)$$

- hyperproperties = sets of sets of traces [CS10]

  ... express properties of sets of traces
  by relating different executions,

  e.g. observational determinism:

  $$\forall \pi. \forall \pi'. \square(in_\pi \leftrightarrow in_{\pi'}) \rightarrow \square(out_\pi \leftrightarrow out_{\pi'})$$

Information-flow Security Policies

Noninterference:

Consider actions of a high - security agent H
and obervations made by a low - security observer L .

For all computatations and *all* sequences of H-actions
the observations of L should be independent of H 's actions.

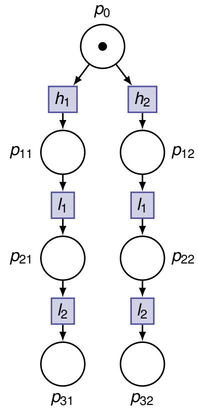
Hyperproperties refer to traces,
which represent concurrency by an interleaving semantics.

We model systems by Petri nets , which represent concurrency using partial orders. This brings us to concurrent hyperproperties .

Example Systems as Petri Nets

Hyperproperty: *All* traces must agree on occurrence and ordering of l_1 and l_2.

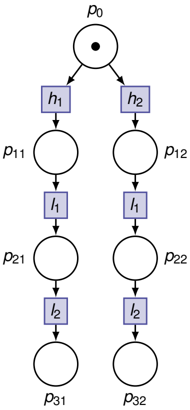
$\mathcal{N}_A$:



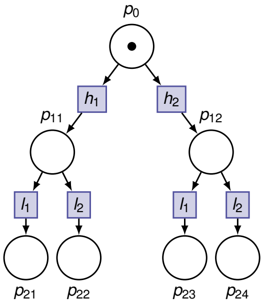
Example Systems as Petri Nets

Hyperproperty: *All* traces must agree on occurrence and ordering of l_1 and l_2.

$\mathcal{N}_A$:



$\mathcal{N}_B$:

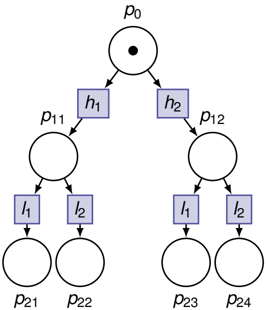


Generalized Noninterference (GNI)

For every pair of traces

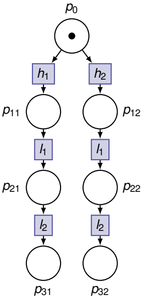
there exists a third trace that agrees
with the first trace on the low - security events and
with the second trace on the high - security events.

$\mathcal{N}_B$:

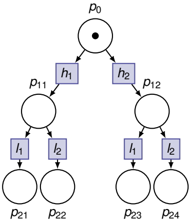


Example Systems as Petri Nets

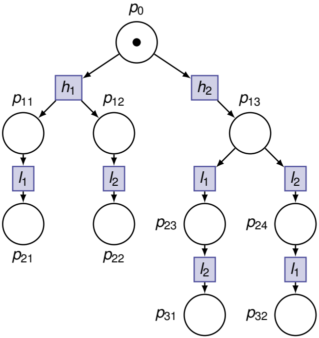
$\mathcal{N}_A$:



$\mathcal{N}_B$:



$\mathcal{N}_C$:



Concurrent Traces = Pomsets

Let Σ be a set of labels. A Σ-labeled partially ordered set is a triple $(X, <, \ell)$ where $<$ is an irreflexive partial order on X and $\ell : X \to \Sigma$ is a labeling function.

A partially ordered multiset (pomset) over Σ is an isomorphy class of Σ-labeled partial ordered sets, denoted as $[(X, <, \ell)]$ [Pra85].

A totally ordered multiset (tomset) is a pomset where $<$ is a total order.

Terminology:

- traces = tomsets over Σ
- trace property = set of traces
- hyperproperty = set of sets of traces
- concurrent traces = pomsets over Σ
- concurrent trace property = set of concurrent traces
- concurrent hyperproperty = set of sets of concurrent traces

$\mathbb{T}(\Sigma)$ = set of all concurrent traces over Σ.

Information Flow Properties I

Every pair of concurrent traces agrees on the occurrence of the *low-security* events, independent on any other event.

Let Σ_{low} be the set of *low-security* events.

The requirement is formalized as the concurrent hyperproperty

$$
\begin{aligned}
H_1 = \{\, T \subseteq \mathbb{T}(\Sigma) \mid\ & \forall\, [(X, <, \ell)],\ [(X', <', \ell')] \in T. \\
& \exists \text{ bijection } f : X_{low} \to X'_{low}. \\
& \forall x \in X_{low}.\ \ell'(f(x)) = \ell(x) \,\}
\end{aligned}
$$

where

$$
\begin{aligned}
X_{low} &= \{x \in X \mid \ell(x) \in \Sigma_{low}\} \\
X'_{low} &= \{x \in X' \mid \ell'(x) \in \Sigma_{low}\}
\end{aligned}
$$

Every pair of concurrent traces agrees
both on the occurrence and the ordering of the *low-security* events.

This requirement is formalized as the concurrent hyperproperty

$$\begin{aligned} H_2 = \{\, T \subseteq \mathbb{T}(\Sigma) \mid\ & \forall\, [(X, <, \ell)],\ [(X', <', \ell')] \in T.\\ & \exists \text{ bijection } f : X_{low} \rightarrow X'_{low}.\\ & (\ \forall x \in X_{low}.\ \ell'(f(x)) = \ell(x)\\ & \wedge \forall x, y \in X_{low}.\ f(x) <' f(y) \Leftrightarrow x < y\,)\,\} \end{aligned}$$

Generalized Noninterference (GNI)

McCullough [McC88]

... for concurrent traces.

Distinguish low-security and high-security events: $\Sigma = \Sigma_{low} \cup \Sigma_{high}$.

For every pair of concurrent traces

there exists a third concurrent trace that agrees
with the first trace on the low - security events and
with the second trace on the high - security events.

Unlike the trace-based version,
this version of GNI distinguishes nondeterminism from concurrency.

GNI as Concurrent Hyperproperty

$$H_3 = \{\ T \subseteq \mathbb{T}(\Sigma)\ \mid\ \ \forall\,[(X,<,\ell)],\ [(X',<',\ell')] \in T.\ \exists\,[(X'',<'',\ell'')] \in T.\ F_{low} \wedge G_{high}\}$$

where

$$F_{low} \equiv \exists \text{bijection } f : X_{low} \to X''_{low}.\ (\ \forall x \in X_{low}.\ \ell''(f(x)) = \ell(x) \wedge \forall x, y \in X_{low}.\ f(x) <'' f(y) \Leftrightarrow x < y\ ),$$

$$G_{high} \equiv \exists \text{bijection } g : X'_{high} \to X''_{high}.\ (\ \forall x \in X'_{high}.\ \ell''(g(x)) = \ell'(x) \wedge \forall x, y \in X'_{high}.\ g(x) <'' g(y) \Leftrightarrow x <' y\ ),$$

$$X_{low} = \{x \in X \mid \ell(x) \in \Sigma_{low}\},$$

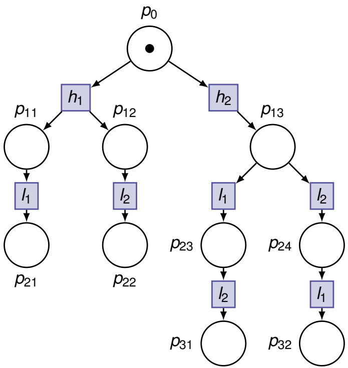
$$X''_{low} = \{x \in X'' \mid \ell''(x) \in \Sigma_{low}\},$$

$$X'_{high} = \{x \in X' \mid \ell'(x) \in \Sigma_{high}\},$$

$$X''_{high} = \{x \in X'' \mid \ell''(x) \in \Sigma_{high}\}.$$

GNI: Traces vs. Concurrent Traces

In the example system $\mathcal{N}_C$



GNI on traces is satisfied, but GNI on concurrent traces is violated.

Testing of Petri Nets

Idea: testing of processes due to De Nicola and Hennessy [DH84]:
interaction of a nondet. process with a user (test) ,
may and must testing.

Here, a test is a Petri net, extended by a set of successful places.
Graphically, we mark these places by $\checkmark$.

To perform a test $\mathscr{T}$ on a given Petri net $\mathscr{N}$,
we consider the parallel composition $\mathscr{N} \| \mathscr{T}$.

A run $\rho = (\mathscr{N}_R, f)$ of $\mathscr{N} \| \mathscr{T}$ is deadlock free if it is infinite;

it terminates successfully if it is finite and
all places of $\mathscr{T}$ inside the parallel composition without causal successor are marked with $\checkmark$.

May and Must

A net $\mathscr{N}$ may pass a test $\mathscr{T}$ if there exists a maximal run of $\mathscr{N} \| \mathscr{T}$ which is deadlock free or terminates successfully.

A net $\mathscr{N}$ must pass a test $\mathscr{T}$ if all maximal runs of $\mathscr{N} \| \mathscr{T}$ are deadlock free or terminate successfully.

To check a hyperproperty relating two concurrent traces of a system $\mathscr{N}_0$, we investigate maximal runs $\rho = (\mathscr{N}, f)$ and $\rho' = (\mathscr{N}', f')$ of $\mathscr{N}_0$, where $\mathscr{N}$ and $\mathscr{N}'$ are causal nets of $\mathscr{N}_0$,
but in $\mathscr{N}'$ every action u of $\mathscr{N}_0$ is relabled into a primed copy u' .

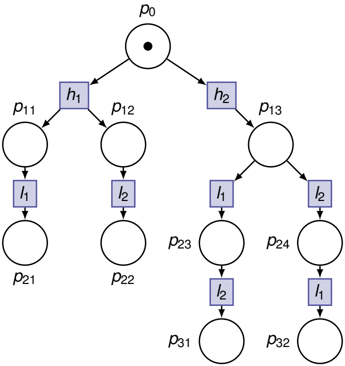
To represent the hyperproperty (with two quantifiers), we test

$$\mathscr{Q}\,\rho . \mathscr{Q}'\,\rho'.\ \mathscr{N} \| \mathscr{N}'\ \mathscr{m}\ \text{pass}\ \mathscr{T},$$

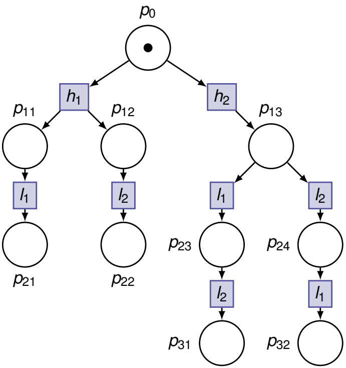
where $\mathscr{Q}, \mathscr{Q}' \in \{\exists, \forall\}$ and $\mathscr{m} \in \{\text{may, must}\}$.

For H_1 and H_2 consider net $\mathcal{N}_C$

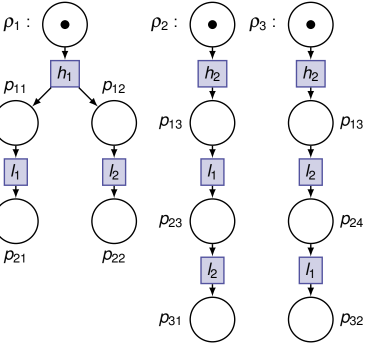
$\mathcal{N}_C$:



$\mathcal{N}_C$:



Three maximal runs:



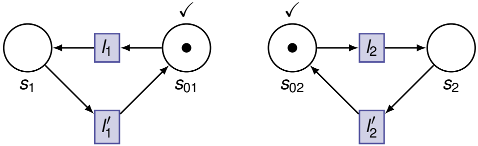
Corresponding traces π_1, π_2, π_3 ignore the places.

Testing Concurrent Hyperproperty H_1

Now we check the concurrent hyperproperty H_1:

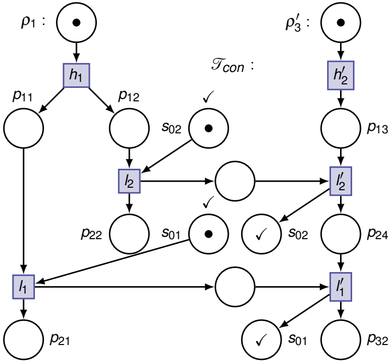
> every pair of concurrent traces π and π'
> agrees on occurrence of low-security events l_1 and l_2.

To this end, we use the concurrent test $\mathscr{T}_{con}$:



Outcome of Concurrent Test $\mathscr{T}_{con}$

The outcome of testing ρ_1 and ρ_3 of $\mathscr{N}_C$:



We conclude that $\rho_1 \| \rho_3$ must pass $\mathscr{T}_{con}$. Indeed, we have

$$\forall \rho, \rho' \,.\, \mathscr{N} \| \mathscr{N}' \text{ must pass } \mathscr{T}_{con}.$$

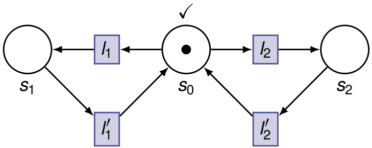
This shows that the system $\mathscr{N}_C$ satisfies H_1 .

Testing Concurrent Hyperproperty H_2

Next we check the concurrent hyperproperty H_2:

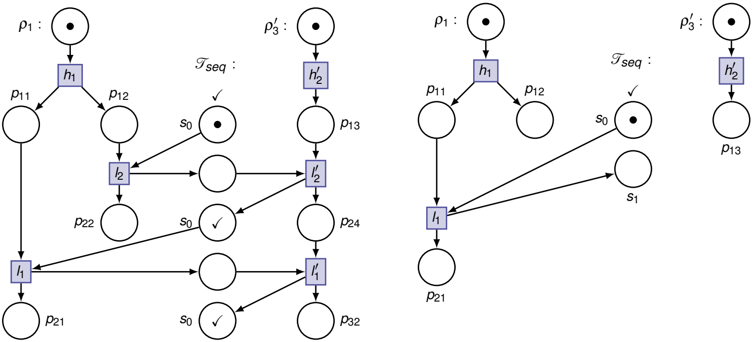
> every pair of concurrent traces π and π'
> agrees on occurrence and ordering of low-security events l_1 and l_2.

To this end, we use the sequential test $\mathscr{T}_{seq}$:



Outcome of Sequential Test $\mathcal{T}_{seq}$

The outcomes of testing ρ_1 and ρ'_3 of $\mathcal{N}_C$:



Two maximal runs of $\rho_1 \parallel \mathcal{T}_{seq} \parallel \rho'_3$.

Left: Here at first the alternative starting with l_2 of the test $\mathcal{T}_{seq}$ is chosen.
This runs terminates successful .

Right: Here at first the alternative starting with l_1 of $\mathcal{T}_{seq}$ is chosen.
This runs ends in a deadlock because ρ_3 engages first in l_2.

Test Result for $\mathscr{N}_{\mathrm{C}}$

- May testing of $\mathscr{N}_{\mathrm{C}}$ successful:

$$\exists \rho, \rho' . \mathscr{N} \| \mathscr{N}' \text{ may pass } \mathscr{T}_{seq}$$

- Must testing $\mathscr{N}_{\mathrm{C}}$ not successful:

$$\forall \rho, \rho' . \mathscr{N} \| \mathscr{N}' \text{ must pass } \mathscr{T}_{seq}$$

does not hold.

So $\mathscr{N}_{\mathrm{C}}$ does not satisfy the concurrent hyperproperty H_2.

Decidability

Universal must testing of a net $\mathscr{N}_0$ of the form

$$(*) \qquad \forall \rho_1, \cdots, \forall \rho_k.\ \mathscr{N}_1 \parallel \cdots \parallel \mathscr{N}_k \text{ must pass } \mathscr{T},$$

can be decided
because its falsification is a reachability problem for Petri nets.

Since we consider safe Petri nets, reachablity is PSPACE-complete [EN94].

Theorem.
Universal may testing is undecidable for tests with two maximal runs.

Proof. We reduce the falsification of the Post Correspondence Problem (PCP) to universal may testing using a test with two maximal runs. $\square$

Proof idea for PCP over alphabet $\{a, b\}$. As an input, consider the set

$$I = \{(u_1, v_1), (u_2, v_2), (u_3, v_3)\},$$

of pairs of subwords, where

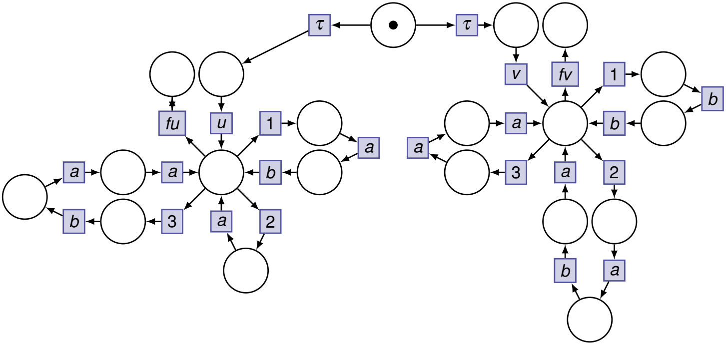
$$u_1 = ab,\ v_1 = bb,\ u_2 = a,\ v_2 = aba,\ u_3 = baa,\ v_3 = aa.$$

This PCP is solvable by the correspondence $(2, 3, 1, 3)$ because

$$u_2 u_3 u_1 u_3 = a\,b\,a\,a\,a\,b\,b\,a\,a = v_2 v_3 v_1 v_3.$$

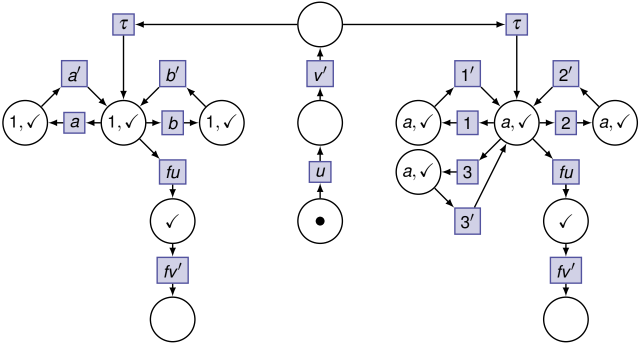
Simulating the Input I

Petri net $\mathcal{N}_I$ simulating the input I of the PCP:



Test $\mathscr{T}_{PCP}$

Test $\mathscr{T}_{PCP}$ for checking whether two runs of $\mathscr{N}$
do not simulate a correspondence of the PCP:



Conclusion

Summary:

- We introduced concurrent hyperproperties as sets of sets of concurrent traces.
- We used Petri nets as semantic model.
- We adapted the testing approach by De Nicola and Hennessy to check concurrent hyperporperties.
- We achieved (un)decidablity results.

Future work:

- Suitable logic for specifying concurrent hyperproperties, extending HyperLTL introduced for normal traces [CFK$^+$14].
- Possible starting point: event structure logic [MT92, Pen95].

References I

Michael R. Clarkson, Bernd Finkbeiner, Masoud Koleini, Kristopher K. Micinski, Markus N. Rabe, and César Sánchez.
Temporal logics for hyperproperties.
In Martín Abadi and Steve Kremer, editors, *Principles of Security and Trust – Third International Conference, POST 2014, Held as Part of ETAPS 2014, Proceedings*, volume 8414 of *LNCS*, pages 265–284. Springer, 2014.

Michael R. Clarkson and Fred B. Schneider.
Hyperproperties.
J. Comput. Secur., 18(6):1157–1210, 2010.

R. DeNicola and M. Hennessy.
Testing equivalences for processes.
TCS, 34:83–134, 1984.

Javier Esparza and Mogens Nielsen.
Decidability issues for Petri nets – a survey.
Bull. EATCS, 52:244–262, 1994.

Daryl McCullough.
Noninterference and the composability of security properties.
In *Proc. IEEE Symposium on Security and Privacy*, pages 177–186. IEEE Computer Society, April 1988.

Madhavan Mukund and P. S. Thiagarajan.
A logical characterization of well branching event structures.
Theor. Comput. Sci., 96(1):35–72, 1992.

Wojciech Penczek.
Branching time and partial order in temporal logics.
In L. Bolc and A. Szalas, editors, *Time and Logics: A Computational Approach*, pages 203–257. UCL Press Ltd., 1995.

References II

Vaughan R. Pratt.
The pomset model of parallel processes: Unifying the temporal and the spatial.
In Stephen D. Brookes, A. W. Roscoe, and Glynn Winskel, editors, *Seminar on Concurrency, Carnegie-Mellon University*, volume 197 of *LNCS*, pages 180–196. Springer, 1985.