Logics for Hyperproperties

Martin Zimmermann

Saarland University

May, 19th 2017

Centre Fédéré en Vérification, Brussels, Belgium
Hyperproperties

The system $S$ is input-deterministic: for all traces $t, t'$ of $S$, $t = I t'$ implies $t = O t'$.

Noninterference: for all traces $t, t'$ of $S$, $t = I public t'$ implies $t = O public t'$.
The system $S$ is input-deterministic: for all traces $t, t'$ of $S$

\[ t =_I t' \implies t =_O t' \]
The system $S$ is input-deterministic: for all traces $t, t'$ of $S$

$$t =_I t' \implies t =_O t'$$

Noninterference: for all traces $t, t'$ of $S$

$$t =_{I_{\text{public}}} t' \implies t =_{O_{\text{public}}} t'$$
Both properties are not trace properties, i.e., sets $T \subseteq \text{Traces}(S)$ of traces, but

- **hyperproperties**, i.e., sets $H \subseteq 2^{\text{Traces}(S)}$ of sets of traces.
- A system $S$ satisfies a hyperproperty $H$, if $\text{Traces}(S) \in H$.

**Example:** Noninterference as trace property:

$$\{ T \subseteq \text{Traces}(S) \mid \forall t, t' \in T : t =_{\text{public}} t' \Rightarrow t =_{\text{public}} t' \}$$
Both properties are not trace properties, i.e., sets $T \subseteq \text{Traces}(S)$ of traces, but

hyperproperties, i.e., sets $H \subseteq 2^{\text{Traces}(S)}$ of sets of traces.

A system $S$ satisfies a hyperproperty $H$, if $\text{Traces}(S) \in H$.

Example: Noninterference as trace property:

$$\{ T \subseteq \text{Traces}(S) \mid \forall t, t' \in T : t =_{\text{public}} t' \Rightarrow t =_{\text{public}} t' \}$$

Specification languages for hyperproperties

HyperLTL: Extend LTL by trace quantifiers.

HyperCTL*: Extend CTL* by trace quantifiers.
Outline

1. HyperLTL
2. The Models Of HyperLTL
3. HyperLTL Satisfiability
4. HyperLTL Model-checking
5. The First-order Logic of Hyperproperties
6. Conclusion
1. HyperLTL
2. The Models Of HyperLTL
3. HyperLTL Satisfiability
4. HyperLTL Model-checking
5. The First-order Logic of Hyperproperties
6. Conclusion
Syntax

\[ \varphi ::= a \mid \neg \varphi \mid \varphi \lor \varphi \mid \varphi \land \varphi \mid X \varphi \mid \varphi U \varphi \]

where \( a \in \text{AP} \) (atomic propositions).
Syntax

\[ \varphi ::= a \mid \neg \varphi \mid \varphi \lor \varphi \mid \varphi \land \varphi \mid X \varphi \mid \varphi U \varphi \]

where \( a \in AP \) (atomic propositions).

Semantics

Given a trace \( w \in (2^{AP})^\omega \) and a position \( n \in \mathbb{N} \):

- \( w, n \models X \varphi \): 
  
  \[ w \mid \ldots \mid \varphi \mid \n \mid n+1 \]

- \( w, n \models \varphi_0 U \varphi_1 \): 
  
  \[ w \mid \ldots \mid \varphi_0 \mid \varphi_0 \mid \varphi_0 \mid \varphi_1 \mid n \]
LTL in One Slide

Syntax

\[ \varphi ::= a \mid \neg \varphi \mid \varphi \lor \varphi \mid \varphi \land \varphi \mid X \varphi \mid \varphi U \varphi \]

where \( a \in AP \) (atomic propositions).

Semantics

\( w, n \models \varphi \) for a trace \( w \in (2^{AP})^\omega \) and a position \( n \in \mathbb{N} \):  

- \( w, n \models X \varphi : \quad w \mid \cdots \mid \varphi \]
  \begin{align*}
  n & \quad n + 1
  \end{align*}

- \( w, n \models \varphi_0 U \varphi_1 : \quad w \mid \cdots \mid \varphi_0 \varphi_0 \varphi_0 \varphi_1 \]
  \begin{align*}
  n & \quad n
  \end{align*}

Syntactic Sugar

- \( F \psi = \text{true} U \psi \)
- \( G \psi = \neg F \neg \psi \)
HyperLTL

HyperLTL = LTL + trace quantification

\[ \varphi ::= \exists \pi. \varphi \mid \forall \pi. \varphi \mid \psi \]
\[ \psi ::= a_\pi \mid \neg \psi \mid \psi \lor \psi \mid X \psi \mid \psi U \psi \]

where \( a \in AP \) (atomic propositions) and \( \pi \in V \) (trace variables).
HyperLTL

HyperLTL = LTL + trace quantification

\[ \varphi ::= \exists \pi. \varphi \mid \forall \pi. \varphi \mid \psi \]
\[ \psi ::= a_\pi \mid \neg \psi \mid \psi \lor \psi \mid X \psi \mid \psi U \psi \]

where \( a \in AP \) (atomic propositions) and \( \pi \in V \) (trace variables).

- Prenex normal form, but
- closed under boolean combinations.
Semantics

\[ \varphi = \forall \pi. \forall \pi'. \text{G on}_\pi \leftrightarrow \text{on}_\pi' \]

\( T \subseteq (2^{\text{AP}})^\omega \) is a model of \( \varphi \) iff
Semantics

\[ \varphi = \forall \pi. \forall \pi'. \mathbf{G} \text{on}_\pi \leftrightarrow \text{on}_\pi' \]

\( T \subseteq (2^{\text{AP}})^\omega \) is a model of \( \varphi \) iff

\( \{\} \models \forall \pi. \forall \pi'. \mathbf{G} \text{on}_\pi \leftrightarrow \text{on}_\pi' \)
Semantics

\[ \varphi = \forall \pi. \forall \pi'. G\, \text{on}_\pi \leftrightarrow \text{on}_\pi' \]

\( T \subseteq (2^{\text{AP}})^\omega \) is a model of \( \varphi \) iff

\[
\begin{align*}
\{\} & \models \forall \pi. \forall \pi'. G\, \text{on}_\pi \leftrightarrow \text{on}_\pi' \\
\{\pi \mapsto t\} & \models \forall \pi'. G\, \text{on}_\pi \leftrightarrow \text{on}_{\pi'} \quad \text{for all } t \in T
\end{align*}
\]
Semantics

\[ \varphi = \forall \pi. \forall \pi'. \text{G on}_\pi \leftrightarrow \text{on}_{\pi'} \]

\( T \subseteq (2^{\text{AP}})^\omega \) is a model of \( \varphi \) iff

\[ \{ \} \models \forall \pi. \forall \pi'. \text{G on}_\pi \leftrightarrow \text{on}_{\pi'} \]

\[ \{ \pi \mapsto t \} \models \forall \pi'. \text{G on}_\pi \leftrightarrow \text{on}_{\pi'} \quad \text{for all } t \in T \]

\[ \{ \pi \mapsto t, \pi' \mapsto t' \} \models \text{G on}_\pi \leftrightarrow \text{on}_{\pi'} \quad \text{for all } t' \in T \]
Semantics

\[ \varphi = \forall \pi. \forall \pi'. \text{G on}_\pi \leftrightarrow \text{on}_\pi' \]

\( T \subseteq (2^{\text{AP}})^\omega \) is a model of \( \varphi \) iff

\[ \{\} \models \forall \pi. \forall \pi'. \text{G on}_\pi \leftrightarrow \text{on}_\pi' \]

\[ \{\pi \mapsto t\} \models \forall \pi'. \text{G on}_\pi \leftrightarrow \text{on}_{\pi'} \quad \text{for all } t \in T \]

\[ \{\pi \mapsto t, \pi' \mapsto t'\} \models \text{G on}_\pi \leftrightarrow \text{on}_{\pi'} \quad \text{for all } t' \in T \]

\[ \{\pi \mapsto t[n, \infty), \pi' \mapsto t'[n, \infty)\} \models \text{on}_\pi \leftrightarrow \text{on}_{\pi'} \quad \text{for all } n \in \mathbb{N} \]
\[ \varphi = \forall \pi. \forall \pi'. \mathbf{G} \text{on}_\pi \leftrightarrow \text{on}_\pi' \]

\( T \subseteq (2^{\text{AP}})^\omega \) is a model of \( \varphi \) iff

\[ \{ \} \models \forall \pi. \forall \pi'. \mathbf{G} \text{on}_\pi \leftrightarrow \text{on}_\pi' \]

\[ \{ \pi \mapsto t \} \models \forall \pi'. \mathbf{G} \text{on}_\pi \leftrightarrow \text{on}_\pi' \quad \text{for all } t \in T \]

\[ \{ \pi \mapsto t, \pi' \mapsto t' \} \models \mathbf{G} \text{on}_\pi \leftrightarrow \text{on}_\pi' \quad \text{for all } t' \in T \]

\[ \{ \pi \mapsto t[n, \infty), \pi' \mapsto t'[n, \infty) \} \models \text{on}_\pi \leftrightarrow \text{on}_\pi' \quad \text{for all } n \in \mathbb{N} \]

\[ \text{on} \in t(n) \Leftrightarrow \text{on} \in t'(n) \]
Applications

- Uniform framework for information-flow control
  - Does a system leak information?
- Symmetries in distributed systems
  - Are clients treated symmetrically?
- Error resistant codes
  - Do codes for distinct inputs have at least Hamming distance $d$?
- Software doping
  - Think emission scandal in automotive industry
LTL has many desirables properties:

1. Every satisfiable LTL formula is satisfied by an ultimately periodic trace, i.e., by a finite and finitely-represented model.
2. LTL satisfiability and model-checking are PSpace-complete.
3. LTL and FO[<] are expressively equivalent.

Which properties does HyperLTL retain?
References


1. HyperLTL

2. The Models Of HyperLTL

3. HyperLTL Satisfiability

4. HyperLTL Model-checking

5. The First-order Logic of Hyperproperties

6. Conclusion
What about Finite Models?

Fix $\text{AP} = \{a\}$ and consider the conjunction $\varphi$ of

- $\forall \pi. \ (\neg a_\pi) \ U \ (a_\pi \land X G \neg a_\pi)$
What about Finite Models?

Fix $\text{AP} = \{a\}$ and consider the conjunction $\varphi$ of

- $\forall \pi. \ (\neg a_\pi) \ U \ (a_\pi \land X G \neg a_\pi)$
- $\exists \pi. \ a_\pi$
What about Finite Models?

Fix $\text{AP} = \{a\}$ and consider the conjunction $\varphi$ of

- $\forall \pi. (\neg a_\pi) \mathbf{U} (a_\pi \land X G \neg a_\pi)$
- $\exists \pi. a_\pi$

$$\{a\} \quad \emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad \ldots$$
What about Finite Models?

Fix $\text{AP} = \{a\}$ and consider the conjunction $\varphi$ of

- $\forall \pi. (\neg a_\pi) \mathcal{U} (a_\pi \land XG \neg a_\pi)$
- $\exists \pi. a_\pi$
- $\forall \pi. \exists \pi'. F (a_\pi \land X a_{\pi'})$

$\{a\}$ $\emptyset$ $\emptyset$ $\emptyset$ $\emptyset$ $\emptyset$ $\emptyset$ $\emptyset$ $\emptyset$ $\emptyset$ $\cdots$
What about Finite Models?

Fix $\text{AP} = \{a\}$ and consider the conjunction $\varphi$ of

- $\forall \pi. (\neg a_\pi) \cup (a_\pi \land X G \neg a_\pi)$
- $\exists \pi. a_\pi$
- $\forall \pi. \exists \pi'. F (a_\pi \land X a_{\pi'})$

\[
\begin{array}{cccccccc}
\{a\} & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \cdots \\
\emptyset & \{a\} & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \cdots \\
\end{array}
\]
What about Finite Models?

Fix $\text{AP} = \{a\}$ and consider the conjunction $\varphi$ of

- $\forall \pi. \ (\neg a_\pi) \ U (a_\pi \land X G \neg a_\pi)$
- $\exists \pi. \ a_\pi$
- $\forall \pi. \ \exists \pi'. \ F (a_\pi \land X a_{\pi'})$

<table>
<thead>
<tr>
<th></th>
<th>$\emptyset$</th>
<th>$\emptyset$</th>
<th>$\emptyset$</th>
<th>$\emptyset$</th>
<th>$\emptyset$</th>
<th>$\emptyset$</th>
<th>$\emptyset$</th>
<th>$\emptyset$</th>
<th>$\emptyset$</th>
<th>$\ldots$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>${a}$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>${a}$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
</tr>
</tbody>
</table>

The unique model of $\varphi$ is $\{\emptyset^n \{a\} \emptyset^\omega \mid n \in \mathbb{N}\}$. 
What about Finite Models?

Fix $AP = \{a\}$ and consider the conjunction $\varphi$ of

- $\forall \pi. (\neg a_\pi) \mathcal{U} (a_\pi \land X \mathcal{G} \neg a_\pi)$
- $\exists \pi. a_\pi$
- $\forall \pi. \exists \pi'. F (a_\pi \land X a_{\pi'})$

$$
\begin{array}{cccccccccc}
\{a\} & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \cdots \\
\emptyset & \{a\} & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \cdots \\
\emptyset & \emptyset & \{a\} & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\end{array}
$$

The unique model of $\varphi$ is $\{\emptyset^n \{a\} \emptyset^\omega \mid n \in \mathbb{N}\}$.

**Theorem**

There is a satisfiable HyperLTL sentence that is not satisfied by any finite set of traces.
What about Countable Models?

Theorem

Every satisfiable HyperLTL sentence has a countable model.
What about Countable Models?

Theorem

*Every satisfiable HyperLTL sentence has a countable model.*

Proof

- W.l.o.g. \( \varphi = \forall \pi_0. \exists \pi'_0. \ldots \forall \pi_k. \exists \pi'_k. \psi \) with quantifier-free \( \psi \).
- Fix a Skolem function \( f_j \) for every existentially quantified \( \pi'_j \).
What about Countable Models?

Theorem

Every satisfiable HyperLTL sentence has a countable model.

Proof

- W.l.o.g. $\varphi = \forall \pi_0. \exists \pi'_0. \cdots \forall \pi_k. \exists \pi'_k. \psi$ with quantifier-free $\psi$.
- Fix a Skolem function $f_j$ for every existentially quantified $\pi'_j$.

$t$
What about Countable Models?

**Theorem**

*Every satisfiable HyperLTL sentence has a countable model.*

**Proof**

- W.l.o.g. $\varphi = \forall \pi_0. \exists \pi'_0. \cdots \forall \pi_k. \exists \pi'_k. \psi$ with quantifier-free $\psi$.
- Fix a Skolem function $f_j$ for every existentially quantified $\pi'_j$. 

![Diagram](image)
What about Countable Models?

**Theorem**

*Every satisfiable HyperLTL sentence has a countable model.*

**Proof**

- W.l.o.g. $\phi = \forall \pi_0. \exists \pi'_0. \cdots \forall \pi_k. \exists \pi'_k. \psi$ with quantifier-free $\psi$.
- Fix a Skolem function $f_j$ for every existentially quantified $\pi'_j$. 

\[
\begin{align*}
&\text{The limit is a model of } \phi \text{ and countable.}
\end{align*}
\]
What about Countable Models?

THEOREM
Every satisfiable HyperLTL sentence has a countable model.

PROOF
- W.l.o.g. \( \varphi = \forall \pi_0. \exists \pi'_0. \cdots \forall \pi_k. \exists \pi'_k. \psi \) with quantifier-free \( \psi \).
- Fix a Skolem function \( f_j \) for every existentially quantified \( \pi'_j \).

\[ t \]

\[ \cdots \]
What about Countable Models?

**Theorem**

*Every satisfiable HyperLTL sentence has a countable model.*

**Proof**

- W.l.o.g. \( \varphi = \forall \pi_0. \exists \pi'_0. \cdots \forall \pi_k. \exists \pi'_k. \psi \) with quantifier-free \( \psi \).
- Fix a Skolem function \( f_j \) for every existentially quantified \( \pi'_j \).

The limit is a model of \( \varphi \) and countable.
Theorem

There is a satisfiable HyperLTL sentence that is not satisfied by any ω-regular set of traces.
What about Regular Models?

**Theorem**

There is a satisfiable HyperLTL sentence that is not satisfied by any \( \omega \)-regular set of traces.

**Proof**

Express that a model \( T \) contains..

1. \( \ldots (\{a\}\{b\})^n\emptyset^\omega \) for every \( n \).
What about Regular Models?

**Theorem**

There is a satisfiable HyperLTL sentence that is not satisfied by any \( \omega \)-regular set of traces.

**Proof**

Express that a model \( T \) contains.. \( \{a\} \{b\} \{a\} \{b\} \{a\} \{b\} \emptyset \omega \)

1. .. \( (\{a\}\{b\})^n \emptyset \omega \) for every \( n \).
What about Regular Models?

**Theorem**

There is a satisfiable HyperLTL sentence that is not satisfied by any $\omega$-regular set of traces.

**Proof**

Express that a model $T$ contains:

1. $(\{a\}\{b\})^n \emptyset^\omega$ for every $n$.
2. for every trace of the form $x\{b\}\{a\}y$ in $T$, also the trace $x\{a\}\{b\}y$. 

$\{a\} \{b\} \{a\} \{b\} \{a\} \{b\} \emptyset^\omega$
Theorem
There is a satisfiable HyperLTL sentence that is not satisfied by any \( \omega \)-regular set of traces.

Proof

Express that a model \( T \) contains.. \( \{a\} \{b\} \{a\} \{b\} \emptyset \omega \)

1. .. \( (\{a\}\{b\})^n \emptyset \omega \) for every \( n \).

2. .. for every trace of the form \( x\{b\}\{a\}y \) in \( T \), also the trace \( x\{a\}\{b\}y \).
What about Regular Models?

**Theorem**

There is a satisfiable HyperLTL sentence that is not satisfied by any \( \omega \)-regular set of traces.

**Proof**

Express that a model \( T \) contains..

1. \( (\{a\}\{b\})^n \emptyset \) for every \( n \).
2. for every trace of the form \( x\{b\}\{a\}y \) in \( T \), also the trace \( x\{a\}\{b\}y \).
What about Regular Models?

**Theorem**

There is a satisfiable HyperLTL sentence that is not satisfied by any \(\omega\)-regular set of traces.

**Proof**

Express that a model \(T\) contains:

1. \((\{a\}\{b\})^n\emptyset^\omega\) for every \(n\).
2. for every trace of the form \(x\{b\}\{a\}y\) in \(T\), also the trace \(x\{a\}\{b\}y\).
What about Regular Models?

**Theorem**

*There is a satisfiable HyperLTL sentence that is not satisfied by any \( \omega \)-regular set of traces.*

**Proof**

Express that a model \( T \) contains..\( \{a\} \{b\} \{a\} \{b\} \{a\} \{b\} \emptyset^\omega \)

1. .. \( (\{a\}\{b\})^n\emptyset^\omega \) for every \( n \). \( \{a\} \{b\} \{a\} \{b\} \{a\} \{b\} \emptyset^\omega \)

2. .. for every trace of the form \( x\{b\}\{a\}y \) in \( T \), also the trace \( x\{a\}\{b\}y \). \( \{a\} \{a\} \{b\} \{a\} \{b\} \{b\} \emptyset^\omega \)

Then, \( T \cap \{a\}^*\{b\}^*\emptyset^\omega = \{\{a\}^n\{b\}^n\emptyset^\omega \mid n \in \mathbb{N}\} \) is not \( \omega \)-regular.
What about Ultimately Periodic Models?

**Theorem**

*There is a satisfiable HyperLTL sentence that is not satisfied by any set of traces that contains an ultimately periodic trace.*
What about Ultimately Periodic Models?

Theorem

There is a satisfiable HyperLTL sentence that is not satisfied by any set of traces that contains an ultimately periodic trace.

One can even encode the prime numbers in HyperLTL!
1. HyperLTL
2. The Models Of HyperLTL
3. HyperLTL Satisfiability
4. HyperLTL Model-checking
5. The First-order Logic of Hyperproperties
6. Conclusion
Undecidability

The HyperLTL satisfiability problem:

Given $\varphi$, is there a non-empty set $T$ of traces with $T \models \varphi$?

**Theorem**

HyperLTL satisfiability is undecidable.
The HyperLTL satisfiability problem:

Given $\varphi$, is there a non-empty set $T$ of traces with $T \models \varphi$?

**Theorem**

*HyperLTL satisfiability is undecidable.*

**Proof:**

By a reduction from Post’s correspondence problem.

**Example**

Blocks $(a, baa)$ $(ab, aa)$ $(bba, bb)$
Undecidability

The HyperLTL satisfiability problem:

Given \( \varphi \), is there a non-empty set \( T \) of traces with \( T \models \varphi \)?

**Theorem**

HyperLTL satisfiability is undecidable.

**Proof:**

By a reduction from Post’s correspondence problem.

**Example**

Blocks \((a, baa)\) \((ab, aa)\) \((bba, bb)\)

A solution:

\[
\begin{array}{cccccccc}
 b & b & a & a & b & b & b & a \\
 b & b & a & a & b & b & b & a \\
\end{array}
\]
The HyperLTL satisfiability problem:

Given $\varphi$, is there a non-empty set $T$ of traces with $T \models \varphi$?

**Theorem**

HyperLTL satisfiability is undecidable.

**Proof:**

By a reduction from Post’s correspondence problem.

**Example**

Blocks $(a, baa)$ $(ab, aa)$ $(bba, bb)$

A solution:

```
  b   b   a   a   b   b   b   a   a
  b   b   a   a   b   b   b   a   a
  b   b   a   a   b   b   b   a   a
```
1. There is a (solution) trace where top matches bottom.
1. There is a (solution) trace where top matches bottom.
1. There is a (solution) trace where top matches bottom.

2. Every trace is finite and starts with a block or is empty.
Undecidability

1. There is a (solution) trace where top matches bottom.

2. Every trace is \emph{finite} and starts with a block or is \emph{empty}.

\[
\begin{align*}
\{b\} & \{b\} \{a\} \{a\} \{b\} \{b\} \{a\} \{a\} \emptyset^\omega \\
\{b\} & \{b\} \{a\} \{a\} \{b\} \{b\} \{a\} \{a\} \emptyset^\omega
\end{align*}
\]
1. There is a (solution) trace where top matches bottom.

2. Every trace is finite and starts with a block or is empty.

3. For every non-empty trace, the trace obtained by removing the first block also exists.
Undecidability

1. There is a (solution) trace where top matches bottom.

2. Every trace is finite and starts with a block or is empty.

3. For every non-empty trace, the trace obtained by removing the first block also exists.
Undecidability

1. There is a (solution) trace where top matches bottom.
2. Every trace is finite and starts with a block or is empty.
3. For every non-empty trace, the trace obtained by removing the first block also exists.
Undecidability

1. There is a (solution) trace where top matches bottom.

2. Every trace is finite and starts with a block or is empty.

3. For every non-empty trace, the trace obtained by removing the first block also exists.
1. There is a (solution) trace where top matches bottom.

2. Every trace is finite and starts with a block or is empty.

3. For every non-empty trace, the trace obtained by removing the first block also exists.
Undecidability

1. There is a (solution) trace where top matches bottom.
2. Every trace is finite and starts with a block or is empty.
3. For every non-empty trace, the trace obtained by removing the first block also exists.
Undecidability

1. There is a (solution) trace where top matches bottom.
2. Every trace is finite and starts with a block or is empty.
3. For every non-empty trace, the trace obtained by removing the first block also exists.
1. There is a (solution) trace where top matches bottom.

2. Every trace is finite and starts with a block or is empty.

3. For every non-empty trace, the trace obtained by removing the first block also exists.
Undecidability

1. There is a (solution) trace where top matches bottom.
2. Every trace is finite and starts with a block or is empty.
3. For every non-empty trace, the trace obtained by removing the first block also exists.
Undecidability

1. There is a (solution) trace where top matches bottom.

2. Every trace is finite and starts with a block or is empty.

3. For every non-empty trace, the trace obtained by removing the first block also exists.
Theorem

$\exists^*-\text{HyperLTL satisfiability is PSpace-complete.}$
Decidability

Theorem

$\exists^*$-HyperLTL satisfiability is PSpace-complete.

Proof:

- Membership:
  - Consider $\varphi = \exists \pi_0 \ldots \exists \pi_k. \psi$.
  - Obtain $\psi'$ from $\psi$ by replacing each $a_{\pi_j}$ by a fresh proposition $a_j$.
  - Then: $\varphi$ and the LTL formula $\psi'$ are equi-satisfiable.

- Hardness: trivial reduction from LTL satisfiability
Decidability

**Theorem**

∀*-HyperLTL satisfiability is PSpace-complete.

Proof:

Membership:
Consider \( \phi = \forall \pi_0 \ldots \forall \pi_k . \psi \).
Obtain \( \psi' \) from \( \psi \) by replacing each \( a_{\pi_j} \) by \( a \).
Then: \( \phi \) and the LTL formula \( \psi' \) are equi-satisfiable.

Hardness: trivial reduction from LTL satisfiability.
Theorem

∀*-HyperLTL satisfiability is PSpace-complete.

Proof:

- Membership:
  - Consider $\varphi = \forall \pi_0 \ldots \forall \pi_k. \psi$.
  - Obtain $\psi'$ from $\psi$ by replacing each $a_{\pi_j}$ by $a$.
  - Then: $\varphi$ and the LTL formula $\psi'$ are equi-satisfiable.

- Hardness: trivial reduction from LTL satisfiability
Decidability

Theorem

$\exists^* \forall^* \text{-HyperLTL satisfiability is ExpSpace-complete.}$
Decidability

Theorem

∃∗∀*-HyperLTL satisfiability is ExpSpace-complete.

Proof:

- Membership:
  - Consider $\varphi = \exists \pi_0 \ldots \exists \pi_k \forall \pi_0' \ldots \forall \pi_{\ell}' \cdot \psi$.
  - Let

\[
\varphi' = \exists \pi_0 \ldots \exists \pi_k \bigwedge_{j_0=0}^{k} \cdots \bigwedge_{j_{\ell}=0}^{k} \psi_{j_0,\ldots,j_{\ell}}
\]

where $\psi_{j_0,\ldots,j_{\ell}}$ is obtained from $\psi$ by replacing each occurrence of $\pi_i'$ by $\pi_{j_i}$.

- Then: $\varphi$ and $\varphi'$ are equi-satisfiable.

- Hardness: encoding of exponential-space Turing machines.
Further Results

HyperLTL implication checking: given $\varphi$ and $\varphi'$, does, for every $T$, $T \models \varphi$ imply $T \models \varphi'$?

**Lemma**

$\varphi$ does not imply $\varphi'$ iff $(\varphi \land \neg \varphi')$ is satisfiable.
Further Results

HyperLTL implication checking: given $\varphi$ and $\varphi'$, does, for every $T$, $T \models \varphi$ imply $T \models \varphi'$?

**Lemma**

$\varphi$ does not imply $\varphi'$ iff $(\varphi \land \neg \varphi')$ is satisfiable.

**Corollary**

*Implication checking for alternation-free HyperLTL formulas is ExpSpace-complete.*

**Tool EAHyper:**

- satisfiability, implication, and equivalence checking for HyperLTL
References


Outline

1. HyperLTL
2. The Models Of HyperLTL
3. HyperLTL Satisfiability
4. HyperLTL Model-checking
5. The First-order Logic of Hyperproperties
6. Conclusion
The HyperLTL model-checking problem:

Given a transition system $S$ and $\varphi$, does $\text{Traces}(S) \models \varphi$?

**Theorem**

The HyperLTL model-checking problem is decidable.
Proof:

- Consider $\varphi = \exists \pi_1. \forall \pi_2. \ldots \exists \pi_{k-1}. \forall \pi_k. \psi$.
- Rewrite as $\exists \pi_1. \neg \exists \pi_2. \neg \ldots \exists \pi_{k-1}. \neg \exists \pi_k. \neg \psi$. 
Proof:

- Consider $\varphi = \exists \pi_1. \forall \pi_2. \ldots \exists \pi_{k-1}. \forall \pi_k. \psi$.
- Rewrite as $\exists \pi_1. \neg \exists \pi_2. \neg \ldots \exists \pi_{k-1}. \neg \exists \pi_k. \neg \psi$.
- By induction over quantifier prefix construct non-deterministic Büchi automaton $A$ with $L(A) \neq \emptyset$ iff $\text{Traces}(S) \models \varphi$.
  - Induction start: build automaton for LTL formula obtained from $\neg \psi$ by replacing $a_{\pi_j}$ by $a_j$.
  - For $\exists \pi_j \theta$ restrict automaton for $\theta$ in dimension $j$ to traces of $S$.
  - For $\neg \theta$ complement automaton for $\theta$. 

Non-elementary complexity, but alternation-free fragments are as hard as LTL.
Proof:

- Consider $\varphi = \exists \pi_1. \forall \pi_2. \ldots \exists \pi_{k-1}. \forall \pi_k. \psi$.
- Rewrite as $\exists \pi_1. \neg \exists \pi_2. \neg \ldots \exists \pi_{k-1}. \neg \exists \pi_k. \neg \psi$.
- By induction over quantifier prefix construct non-deterministic Büchi automaton $A$ with $L(A) \neq \emptyset$ iff $\text{Traces}(S) \models \varphi$.
  
  - Induction start: build automaton for LTL formula obtained from $\neg \psi$ by replacing $a_{\pi_j}$ by $a_j$.
  - For $\exists \pi_j \theta$ restrict automaton for $\theta$ in dimension $j$ to traces of $S$.
  - For $\neg \theta$ complement automaton for $\theta$.

$\Rightarrow$ Non-elementary complexity, but alternation-free fragments are as hard as LTL.
Outline

1. HyperLTL
2. The Models Of HyperLTL
3. HyperLTL Satisfiability
4. HyperLTL Model-checking
5. The First-order Logic of Hyperproperties
6. Conclusion
First-order Logic vs. LTL

FO[<]: first-order order logic over signature \( \{<\} \cup \{P_a \mid a \in AP\} \) over structures with universe \( \mathbb{N} \).

**Theorem (Kamp ’68, Gabbay et al. ’80)**

*LTL and FO[<] are expressively equivalent.*
First-order Logic vs. LTL

FO[<]: first-order order logic over signature \( \{<\} \cup \{P_a \mid a \in AP\} \) over structures with universe \( \mathbb{N} \).

**Theorem (Kamp ’68, Gabbay et al. ’80)**

LTL and FO[<] are expressively equivalent.

**Example**

\[ \forall x (P_q(x) \land \neg P_p(x)) \rightarrow \exists y (x < y \land P_p(y)) \]

and

\[ G(q \rightarrow Fp) \]

are equivalent.
First-order Logic for Hyperproperties

\[ \mathbb{N} \]
First-order Logic for Hyperproperties

\[ T \{ \mathbb{N} \} \]

\[ \langle,\rangle \cup \{ \text{pred} | \text{pred} \in \text{AP} \} \]

over structures with universe \( T \times \mathbb{N} \).
First-order Logic for Hyperproperties

\[ \mathcal{T} \{ \lesssim, E \}, \mathbb{N} \]

Martin Zimmermann  Saarland University  Logics for Hyperproperties  32/40
First-order Logic for Hyperproperties

\[ \forall x \forall x' \ E(x, x') \rightarrow (P_{on}(x) \leftrightarrow P_{on}(x')) \]

- **FO[<, E]**: first-order logic with equality over the signature \( \{<, E\} \cup \{P_a \mid a \in AP\} \) over structures with universe \( T \times \mathbb{N} \).

**Example**
First-order Logic for Hyperproperties

**Proposition**

For every HyperLTL sentence there is an equivalent $\text{FO}[<, E]$ sentence.
Let $\varphi$ be the following property of sets $T \subseteq (2\{p\})^\omega$:

There is an $n$ such that $p \notin t(n)$ for every $t \in T$.

**Theorem (Bozzelli et al. ’15)**

$\varphi$ is not expressible in HyperLTL.
Let $\varphi$ be the following property of sets $T \subseteq (2\{p\})^\omega$:

There is an $n$ such that $p \notin t(n)$ for every $t \in T$.

**Theorem (Bozzelli et al. ’15)**

$\varphi$ is not expressible in HyperLTL.

But, $\varphi$ is easily expressible in $\text{FO}[<, E]$: 

$$\exists x \forall y \ E(x, y) \rightarrow \neg P_p(y)$$

**Corollary**

$\text{FO}[<, E]$ strictly subsumes HyperLTL.
HyperFO

- $\exists^M x$ and $\forall^M x$: quantifiers restricted to initial positions.
- $\exists^G y \geq x$ and $\forall^G y \geq x$: if $x$ is initial, then quantifiers restricted to positions on the same trace as $x$. 
HyperFO

- $\exists^M x$ and $\forall^M x$: quantifiers restricted to initial positions.
- $\exists^G y \geq x$ and $\forall^G y \geq x$: if $x$ is initial, then quantifiers restricted to positions on the same trace as $x$.

**HyperFO**: sentences of the form

$$\varphi = Q_1^M x_1 \cdot \ldots \cdot Q_k^M x_k. \ Q_1^G y_1 \geq x_{g_1} \cdot \ldots \cdot Q_\ell^G y_\ell \geq x_{g_\ell}. \ \psi$$

- $Q \in \{\exists, \forall\}$,
- $\{x_1, \ldots, x_k\}$ and $\{y_1, \ldots, y_\ell\}$ are disjoint,
- every guard $x_{g_i}$ is in $\{x_1, \ldots, x_k\}$, and
- $\psi$ is quantifier-free over signature $\{<, E\} \cup \{P_a \mid a \in \text{AP}\}$ with free variables in $\{y_1, \ldots, y_\ell\}$. 

Martin Zimmermann Saarland University Logics for Hyperproperties 34/40
Equivalence

Theorem

HyperLTL and HyperFO are equally expressive.
Equivalence

Theorem

HyperLTL and HyperFO are equally expressive.

Proof

- From HyperLTL to HyperFO: structural induction.
- From HyperFO to HyperLTL: reduction to Kamp’s theorem.
\( \forall x \forall x' \ E(x, x') \rightarrow (P_{on}(x) \leftrightarrow P_{on}(x')) \)
∀x∀x' \quad E(x, x') \rightarrow (P_{on}(x) \Leftrightarrow P_{on}(x'))

∀Mx_1 \forall^Mx_2 \quad \forall^G y_1 \geq x_1 \forall^G y_2 \geq x_2 E(y_1, y_2) \rightarrow (P_{on}(y_1) \Leftrightarrow P_{on}(y_2))
∀x∀x’ E(x, x’) → (P_{on}(x) ↔ P_{on}(x’))

∀^M x_1 ∀^M x_2 ∀^G y_1 ≥ x_1 ∀^G y_2 ≥ x_2 E(y_1, y_2) → (P_{on}(y_1) ↔ P_{on}(y_2))
From HyperFO to HyperLTL

\[ \forall x \forall x' \ E(x, x') \rightarrow (P_{on}(x) \leftrightarrow P_{on}(x')) \]

\[ \forall^G y_1 \geq x_1 \forall^G y_2 \geq x_2 E(y_1, y_2) \rightarrow (P_{on}(y_1) \leftrightarrow P_{on}(y_2)) \]

\[ x_1 \mapsto \{ \text{on} \} \quad \{ \text{on} \} \quad \emptyset \quad \{ \text{on} \} \quad \cdots \]

\[ x_2 \mapsto \{ \text{on} \} \quad \emptyset \quad \emptyset \quad \{ \text{on} \} \quad \cdots \]
\[
\forall x \forall x' \ E(x, x') \rightarrow (P_{\text{on}}(x) \leftrightarrow P_{\text{on}}(x')) \\
\forall^G y_1 \geq x_1 \forall^G y_2 \geq x_2 E(y_1, y_2) \rightarrow (P_{\text{on}}(y_1) \leftrightarrow P_{\text{on}}(y_2)) \\
\forall y_1 \forall y_2 \ (y_1 = y_2) \rightarrow (P_{(\text{on},1)}(y_1) \leftrightarrow P_{(\text{on},2)}(y_2))
\]
∀x∀x' \ E(x, x') \rightarrow (P_{on}(x) \leftrightarrow P_{on}(x'))

∀^G y_1 \geq x_1 \forall^G y_2 \geq x_2 \ E(y_1, y_2) \rightarrow (P_{on}(y_1) \leftrightarrow P_{on}(y_2))

∀y_1 \forall y_2 \ (y_1 = y_2) \rightarrow (P_{(on, 1)}(y_1) \leftrightarrow P_{(on, 2)}(y_2))

G((on, 1) \leftrightarrow (on, 2))
\[\forall x \forall x' \quad E(x, x') \rightarrow (P_{\text{on}}(x) \leftrightarrow P_{\text{on}}(x'))\]

\[\forall^M x_1 \forall^M x_2 \quad \forall^G y_1 \geq x_1 \forall^G y_2 \geq x_2 E(y_1, y_2) \rightarrow (P_{\text{on}}(y_1) \leftrightarrow P_{\text{on}}(y_2))\]

\[\forall y_1 \forall y_2 \quad (y_1 = y_2) \rightarrow (P_{(\text{on}, 1)}(y_1) \leftrightarrow P_{(\text{on}, 2)}(y_2))\]

\[G ((\text{on}, 1) \leftrightarrow (\text{on}, 2))\]
\[ \forall x \forall x' \ E(x, x') \rightarrow (P_{\text{on}}(x) \leftrightarrow P_{\text{on}}(x')) \]

\[ \forall^M x_1 \forall^M x_2 \ \forall^G y_1 \geq x_1 \forall^G y_2 \geq x_2 \ E(y_1, y_2) \rightarrow (P_{\text{on}}(y_1) \leftrightarrow P_{\text{on}}(y_2)) \]

\[ \forall y_1 \forall y_2 \ (y_1 = y_2) \rightarrow (P_{(\text{on}, 1)}(y_1) \leftrightarrow P_{(\text{on}, 2)}(y_2)) \]

\[ \mathbf{G} ((\text{on}, 1) \leftrightarrow (\text{on}, 2)) \]

\[ \forall \pi_1 \forall \pi_2 \ \mathbf{G} (\text{on}_{\pi_1} \leftrightarrow \text{on}_{\pi_2}) \]

\[ \pi_1 \mapsto \{\text{on}\} \ \{\text{on}\} \ \emptyset \ \{\text{on}\} \ \cdots \]

\[ \pi_2 \mapsto \{\text{on}\} \ \emptyset \ \emptyset \ \{\text{on}\} \ \cdots \]
References

Outline

1. HyperLTL
2. The Models Of HyperLTL
3. HyperLTL Satisfiability
4. HyperLTL Model-checking
5. The First-order Logic of Hyperproperties
6. Conclusion
Conclusion

HyperLTL behaves quite differently than LTL:

- The models of HyperLTL are rather not well-behaved, i.e., in general (countably) infinite, non-regular, and non-periodic.
- Satisfiability is in general undecidable.
- Model-checking is decidable, but non-elementary.
Conclusion

HyperLTL behaves quite differently than LTL:

- The models of HyperLTL are rather not well-behaved, i.e., in general (countably) infinite, non-regular, and non-periodic.
- Satisfiability is in general undecidable.
- Model-checking is decidable, but non-elementary.

But with the feasible problems, you can do exciting things:

HyperLTL is a powerful tool for information security and beyond

- Information-flow control
- Symmetries in distributed systems
- Error resistant codes
- Software doping
Open Problems

- Is there a class of languages $\mathcal{L}$ such that every satisfiable HyperLTL sentence has a model from $\mathcal{L}$?
- Is the quantifier alternation hierarchy strict?
- HyperLTL synthesis
- Is there a temporal logic that is expressively equivalent to $\text{FO}[<, E]$?
- What about HyperCTL$^*$?
- Software model-checking
- Quantitative hyperproperties
Open Problems

- Is there a class of languages $\mathcal{L}$ such that every satisfiable HyperLTL sentence has a model from $\mathcal{L}$?
- Is the quantifier alternation hierarchy strict?
- HyperLTL synthesis
- Is there a temporal logic that is expressively equivalent to $\text{FO}[<, E]$?
- What about HyperCTL*?
- Software model-checking
- Quantitative hyperproperties

Thank you