How undecidable are HyperLTL and HyperCTL*?

Marie Fortin, Louwe Kuijer, Patrick Totzke, Martin Zimmermann
University of Liverpool

Séminaire Automates, IRIF, Paris – December 10, 2021
Specifications that **relate multiple executions** of a system, such as in information-flow security policies.
Specifications that relate multiple executions of a system, such as in information-flow security policies.

e.g. “no secret information should leak to low-level users”
Specifications that relate multiple executions of a system, such as in information-flow security policies.

- Noninterference
- Observational determinism
- Declassification
- ...
HyperLTL and HyperCTL*

HyperLTL = CTL* + path quantifiers
= LTL + trace quantifiers

Linear time
Branching time

LTL

CTL*

3
HyperLTL and HyperCTL*

HyperLTL = CTL* + path quantifiers
HyperCTL = LTL + trace quantifiers

Linear time

Branching time

One execution at a time
HyperLTL and HyperCTL*

HyperLTL = LTL + trace quantifiers

CTL* one execution at a time
HyperLTL and HyperCTL$^*$

HyperLTL

= LTL + trace quantifiers

CTL$^*$

= CTL$^*$ + path quantifiers

branching time

linear time

one execution at a time
HyperLTL [Clarkson et al. 2014]

**Syntax of HyperLTL**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varphi ::= \exists \pi. \varphi \mid \forall \pi. \varphi \mid \psi$</td>
<td>LTL formula with trace quantifiers</td>
</tr>
<tr>
<td>$\psi ::= p_\pi \mid \neg \psi \mid \psi \lor \psi \mid \psi \land \psi \mid X \psi \mid \psi U \psi \mid F \psi \mid G \psi$</td>
<td>LTL operators</td>
</tr>
</tbody>
</table>

“Next $\psi$” “$\psi$ Until $\psi$” “Eventually $\psi$” “Globally $\psi$”
Syntax of HyperLTL

\[ \varphi ::= \exists \pi. \varphi \mid \forall \pi. \varphi \mid \psi \]

\[ \psi ::= p_\pi \mid \neg \psi \mid \psi \lor \psi \mid \psi \land \psi \mid X \psi \mid U \psi \mid F \psi \mid G \psi \]

“Next \( \psi \)” \hspace{1cm} “\( \psi \) Until \( \psi \)” \hspace{1cm} “Eventually \( \psi \)” \hspace{1cm} “Globally \( \psi \)”
## Syntax of HyperLTL

\[
\varphi ::= \exists \pi. \varphi \mid \forall \pi. \varphi \mid \psi
\]

\[
\psi ::= p_\pi \mid \neg \psi \mid \psi \lor \psi \mid \psi \land \psi \mid X \psi \mid \psi U \psi \mid F \psi \mid G \psi
\]

- "\( p \) holds at the current position on trace \( \pi \)"
- "Next \( \psi \)"
- "\( \psi \) Until \( \psi \)"
- "Eventually \( \psi \)"
- "Globally \( \psi \)"
HyperLTL [Clarkson et al. 2014]

Syntax of HyperLTL

$\phi ::= \exists \pi. \phi \mid \forall \pi. \phi \mid \psi$

$\psi ::= p_\pi \mid \neg \psi \mid \psi \lor \psi \mid \psi \land \psi \mid X \psi \mid \psi U \psi \mid F \psi \mid G \psi$

“$p$ holds at the current position on trace $\pi$”

“Next $\psi$” “$\psi$ Until $\psi$” “Eventually $\psi$” “Globally $\psi$”

Example:

$\forall \pi. \forall \pi'. G(in\_public_\pi \leftrightarrow in\_public_{\pi'})$

$\rightarrow G(out\_public_\pi \leftrightarrow out\_public_{\pi'})$

“Any two traces with the same public input have the same public output”
HyperLTL [Clarkson et al. 2014]

Syntax of HyperLTL

\[
\varphi ::= \exists \pi. \varphi \mid \forall \pi. \varphi \mid \psi
\]

\[
\psi ::= p_\pi \mid \neg \psi \mid \psi \vee \psi \mid \psi \wedge \psi \mid X \psi \mid \psi U \psi \mid F \psi \mid G \psi
\]

“\(p\) holds at the current position on trace \(\pi\)”

“Next \(\psi\)”  “\(\psi\) Until \(\psi\)”  “Eventually \(\psi\)”  “Globally \(\psi\)”

Example:

\[
\forall \pi. \forall \pi'. G\left(\text{in\_public}_\pi \leftrightarrow \text{in\_public}_{\pi'}\right) \\
\rightarrow G\left(\text{out\_public}_\pi \leftrightarrow \text{out\_public}_{\pi'}\right)
\]

“Any two traces with the same public input have the same public output”

Always in prenex normal form: \(Q_1 \pi_1 Q_n \pi_n \ldots \varphi\)

trace quantifiers LTL formula
A transition system $\mathcal{T}$ satisfies a HyperLTL formula $\varphi$ if $\text{Traces}(\mathcal{T}) \models \varphi$. 

Example:

$\emptyset$ $\emptyset$ $\emptyset$
$\emptyset$ $\emptyset$ $\emptyset$ $\emptyset$
$\emptyset$ $\emptyset$ $\emptyset$ $\emptyset$ $\emptyset$ 

$\mathcal{T} |\models \forall \pi. \exists \pi'. G(a_\pi \leftrightarrow X a_{\pi'})$
HyperLTL [Clarkson et al. 2014]

A transition system $\mathcal{T}$ satisfies a HyperLTL formula $\varphi$ if $\text{Traces}(\mathcal{T}) \models \varphi$.

Example

$$\mathcal{T} = \begin{array}{c}
\emptyset \\
\{a\} \\
\emptyset \\
\{a\}
\end{array}$$
A transition system $\mathcal{T}$ satisfies a HyperLTL formula $\varphi$ if $\text{Traces}(\mathcal{T}) \models \varphi$.

**Example**

$$\mathcal{T} = \begin{array}{c}
\emptyset \\
\{a\} \\
\emptyset \\
\{a\}
\end{array}$$

$$\text{Traces}(\mathcal{T}) = \{ \emptyset \{a\} \emptyset \{a\} \{a\} \{a\} \{a\} \{a\} \ldots, \\
\emptyset \emptyset \{a\} \emptyset \{a\} \{a\} \{a\} \{a\} \ldots, \\
\emptyset \emptyset \emptyset \{a\} \emptyset \{a\} \{a\} \{a\} \ldots, \\
\ldots \}$$
HyperLTL [Clarkson et al. 2014]

A transition system $\mathcal{T}$ satisfies a HyperLTL formula $\varphi$ if $\text{Traces}(\mathcal{T}) \models \varphi$.

**Example**

$\mathcal{T} =$

$\text{Traces}(\mathcal{T}) = \{ \emptyset \{a\} \emptyset \{a\} \{a\} \{a\} \{a\} \{a\} \cdots , \emptyset \emptyset \{a\} \emptyset \{a\} \{a\} \{a\} \{a\} \cdots , \emptyset \emptyset \emptyset \{a\} \emptyset \{a\} \{a\} \{a\} \cdots , \cdots \}$

$\mathcal{T} \models \forall \pi. \exists \pi'. \text{G}(a_\pi \leftrightarrow X a_{\pi'})$
HyperCTL* [Clarkson et al. 2014]

Syntax of HyperCTL

\[ \phi ::= \exists \pi. \phi \mid \forall \pi. \phi \mid p_\pi \mid \neg \phi \mid \phi \lor \phi \mid X \phi \mid \phi U \phi \mid F \phi \mid G \phi \]

- No prenex normal form
- Branching-time semantics:
  - Evaluation over transition systems/computation trees rather than sets of traces
  - Quantifiers range over paths in the transition system, starting in the latest state
  - Time progresses synchronously in all branches when evaluating temporal operators
- Strict generalization of both HyperLTL and CTL
HyperCTL* [Clarkson et al. 2014]

Syntax of HyperCTL*

\[ \varphi ::= \exists \pi. \varphi \mid \forall \pi. \varphi \mid p_\pi \mid \neg \varphi \mid \varphi \lor \varphi \mid X \varphi \mid \varphi \lor \varphi \mid F \varphi \mid G \varphi \]
HyperCTL* [Clarkson et al. 2014]

Syntax of HyperCTL*

\[ \phi ::= \exists \pi. \phi \mid \forall \pi. \phi \mid p_\pi \mid \neg \phi \mid \phi \lor \psi \mid X \phi \mid \phi U \psi \mid F \phi \mid G \phi \]

- No prenex normal form
HyperCTL* [Clarkson et al. 2014]

Syntax of HyperCTL*

\[ \varphi ::= \exists \pi. \varphi \mid \forall \pi. \varphi \mid p_{\pi} \mid \neg \varphi \mid \varphi \lor \varphi \mid X \varphi \mid \varphi U \varphi \mid F \varphi \mid G \varphi \]

- No prenex normal form
- Branching-time semantics:
  - evaluation over transition systems/computation trees rather than sets of traces
  - quantifiers range over paths in the transition system, starting in the latest state
HyperCTL* [Clarkson et al. 2014]

Syntax of HyperCTL*

\[ \varphi ::= \exists \pi. \varphi \mid \forall \pi. \varphi \mid p_\pi \mid \neg \varphi \mid \varphi \lor \varphi \mid X \varphi \mid \varphi U \varphi \mid F \varphi \mid G \varphi \]

- No prenex normal form
- Branching-time semantics:
  - evaluation over transition systems/computation trees rather than sets of traces
  - quantifiers range over paths in the transition system, starting in the latest state
- Time progresses synchronously in all branches when evaluating temporal operators
HyperCTL* [Clarkson et al. 2014]

Syntax of HyperCTL*

\[ \varphi ::= \exists \pi.\varphi \mid \forall \pi.\varphi \mid p_\pi \mid \neg \varphi \mid \varphi \lor \varphi \mid X \varphi \mid \varphi U \varphi \mid F \varphi \mid G \varphi \]

- No prenex normal form
- Branching-time semantics:
  - evaluation over transition systems/computation trees rather than sets of traces
  - quantifiers range over paths in the transition system, starting in the latest state
- Time progresses synchronously in all branches when evaluating temporal operators
- Strict generalization of both HyperLTL and CTL*
Example

\[ \varphi = \forall \pi. \ G(a_\pi \rightarrow \forall \pi'. G(b_\pi \leftrightarrow b_{\pi'})) \]
Example

\[ \varphi = \forall \pi. \ G(a_\pi \rightarrow \forall \pi'. G(b_\pi \leftrightarrow b_{\pi'})) \]

“All paths starting from an \( a \)-labeled state coincide on \( b \)”
HyperLTL and HyperCTL* model-checking problems are decidable . . .

. . . but their satisfiability problems are undecidable.
HyperLTL and HyperCTL* model-checking problems are decidable . . .

... but their satisfiability problems are undecidable.

How undecidable is HyperLTL or HyperCTL* satisfiability?
Levels of Undecidability

Decidable

Undecidable

Recursively enumerable

Co-recursively enumerable

\[
\begin{align*}
\Sigma_0^0 &= \Pi_0^0 \\
\Sigma_1^0 &= \Sigma_2^0 \\
\Pi_1^0 &= \Pi_2^0 \\
\ldots
\end{align*}
\]

\[
\begin{align*}
\Sigma_1^1 &= \Sigma_2^1 \\
\Pi_1^1 &= \Pi_2^1 \\
\ldots
\end{align*}
\]

\[
\begin{align*}
\Sigma_2^2 &= \Sigma_2^2 \\
\Pi_1^2 &= \Pi_2^2 \\
\ldots
\end{align*}
\]

arithmetical hierarchy
analytical hierarchy
Levels of Undecidability

Decidable

\[ \Sigma_0^0 = \Pi_0^0 \]

\[ \Sigma_1^0 \rightarrow \Sigma_2^0 \]

\[ \Pi_1^0 \rightarrow \Pi_2^0 \]

Undecidable

\[ \Sigma_0^1 = \Pi_0^1 \]

\[ \Sigma_1^1 \rightarrow \Sigma_2^1 \]

\[ \Pi_1^1 \rightarrow \Pi_2^1 \]

\[ \Sigma_0^2 = \Pi_0^2 \]

\[ \Sigma_1^2 \rightarrow \Sigma_2^2 \]

\[ \Pi_1^2 \rightarrow \Pi_2^2 \]

Recursively enumerable

Co-recursively enumerable

\[ \Sigma_i^j : \text{definable by a formula of the form} \exists^* \forall^* \exists^* \forall^* \cdots \phi \]

i blocks

(j + 1)th order variables

j-th order arithmetic

arithmetical hierarchy

analytical hierarchy
Levels of Undecidability

\[ \Sigma_0 = \Pi_0 \]

\[ \Sigma_1 = \Sigma_2 \]

\[ \Pi_1 = \Pi_2 \]

\[ \cdots \]

\[ \Sigma^j_0 = \Pi^j_0 \]

\[ \Sigma^j_1 = \Sigma^j_2 \]

\[ \Pi^j_1 = \Pi^j_2 \]

\[ \cdots \]

\[ \Sigma^j_i : \text{definable by a formula of the form } \exists^* \forall^* \exists^* \forall^* \cdots \varphi \]

\[ \text{arithmetical hierarchy} \]

\[ \text{analytical hierarchy} \]

\[ \text{third-order arithmetic} \]

\[ \text{first-order arithmetic} \]

\[ \text{second-order arithmetic} \]
What Was Known

- HyperLTL satisfiability is $\Sigma^0_1$-hard. [Finkbeiner, Hahn 2016]
- HyperLTL satisfiability restricted to finite sets of traces is $\Sigma^0_1$-complete (complete for the class of recursively enumerable problems). [Finkbeiner, Hahn, Hans 2018], [Mascle, Zimmermann, 2020]
- HyperCTL$^\ast$ satisfiability is $\Sigma^1_1$-hard. [Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez 2014]
- Finite-state HyperCTL$^\ast$ satisfiability is $\Sigma^0_1$-complete. [Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez 2014]
What Was Known

• HyperLTL satisfiability is $\Sigma^0_1$-hard.  
  [Finkbeiner, Hahn 2016]

• HyperLTL satisfiability restricted to finite sets of traces is $\Sigma^0_1$-complete (complete for the class of recursively enumerable problems).  
  [Finkbeiner, Hahn, Hans 2018], [Mascle, Zimmermann, 2020]

• HyperCTL* satisfiability is $\Sigma^1_1$-hard.  
  [Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez 2014]

• Finite-state HyperCTL* satisfiability is $\Sigma^0_1$-complete.  
  [Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez 2014]
What Was Known

- HyperLTL satisfiability is $\Sigma^0_1$-hard.  [Finkbeiner, Hahn 2016]
- HyperLTL satisfiability restricted to finite sets of traces is $\Sigma^0_1$-complete (complete for the class of recursively enumerable problems).
  
  [Finkbeiner, Hahn, Hans 2018], [Mascle, Zimmermann, 2020]
What Was Known

- HyperLTL satisfiability is $\Sigma^0_1$-hard.  
  [Finkbeiner, Hahn 2016]

- HyperLTL satisfiability **restricted to finite sets of traces** is $\Sigma^0_1$-complete (complete for the class of recursively enumerable problems).
  
  [Finkbeiner, Hahn, Hans 2018], [Mascle, Zimmermann, 2020]

- HyperCTL* satisfiability is $\Sigma^1_1$-hard.
  
  [Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez 2014]
• HyperLTL satisfiability is $\Sigma^0_1$-hard. [Finkbeiner, Hahn 2016]

• HyperLTL satisfiability restricted to finite sets of traces is $\Sigma^0_1$-complete (complete for the class of recursively enumerable problems).
  [Finkbeiner, Hahn, Hans 2018], [Mascle, Zimmermann, 2020]

• HyperCTL* satisfiability is $\Sigma^1_1$-hard.
  [Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez 2014]

• Finite-state HyperCTL* satisfiability is $\Sigma^0_1$-complete.
  [Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez 2014]
Completeness Results

Decidable

Undecidable

\[ \Sigma_0^0 = \Pi_0^0 \]

\[ \Sigma_1^0 \rightarrow \Sigma_2^0 \]

\[ \Pi_1^0 \rightarrow \Pi_2^0 \]

\[ \vdots \]

\[ \Sigma_0^1 = \Pi_0^1 \]

\[ \Sigma_1^1 \rightarrow \Sigma_2^1 \]

\[ \Pi_1^1 \rightarrow \Pi_2^1 \]

\[ \vdots \]

\[ \Sigma_0^2 = \Pi_0^2 \]

\[ \Sigma_1^2 \rightarrow \Sigma_2^2 \]

\[ \Pi_1^2 \rightarrow \Pi_2^2 \]

\[ \vdots \]

Recursively enumerable

Co-recursively enumerable

arithmetical hierarchy

first-order arithmetic

analytical hierarchy

second-order arithmetic

third-order arithmetic
Completeness Results

$\Sigma_0^0 = \Pi_0^0$

Decidable

Undecidable

Recursively enumerable

Co-recursively enumerable

$\Sigma_1^0 \rightarrow \Sigma_2^0$

$\Pi_1^0 \rightarrow \Pi_2^0$

$\Sigma_1^1 \rightarrow \Sigma_2^1$

$\Pi_1^1 \rightarrow \Pi_2^1$

$\Sigma_1^2 \rightarrow \Sigma_2^2$

$\Pi_1^2 \rightarrow \Pi_2^2$

HyperlTL satisfiability

11
Completeness Results

$\Sigma_0 = \Pi_0$

$\Sigma_1 \rightarrow \Sigma_2$

$\Pi_1 \rightarrow \Pi_2$

Decidable

Recursively enumerable

Undecidable

HyperLTL satisfiability

HyperCTL* satisfiability

Co-recursively enumerable

11

Arithmetical hierarchy
- first-order arithmetic

Analytical hierarchy
- second-order arithmetic
- third-order arithmetic
Completeness Results

Decidable

Undecidable

Recursively enumerable

HyperLTL satisfiability

HyperCTL* satisfiability

\[ \Sigma^0_0 = \Pi^0_0 \]
\[ \Sigma^1_1 = \Sigma^1_1 \]
\[ \Sigma^2_2 = \Sigma^2_2 \]

\[ \Pi^0_0 \rightarrow \Pi^0_0 \]
\[ \Pi^1_1 \rightarrow \Pi^1_1 \]
\[ \Pi^2_2 \rightarrow \Pi^2_2 \]

Co-recursively enumerable

Membership at any level of HyperLTL quantifier alternation hierarchy

arithmetical hierarchy

analytical hierarchy

First-order arithmetic

Second-order arithmetic

Third-order arithmetic

Recursive enumerability

Co-recursively enumerable

Decidability

Arithmetical hierarchy

Analytical hierarchy

First-order arithmetic

Second-order arithmetic

Third-order arithmetic
HyperLTL Satisfiability – Membership

**Lemma**

HyperLTL satisfiability is in $\Sigma_1^1$
HyperLTL Satisfiability – Membership

**Lemma**

HyperLTL satisfiability is in $\Sigma^1_1$, i.e., there is a formula

$$\Phi(x) = \exists x_1, \ldots, \exists x_n. \Phi_0(x, x_1, \ldots, x_n)$$

such that

$\psi \in \text{HyperLTL}$ is satisfiable iff $\Phi([\psi])$ holds.

Some intuitions:

- Minimal size of a model: every satisfiable HyperLTL formula has a countable model [Finkbeiner, Zimmermann '17]
- Countable models can be seen as functions from $\mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ mapping a pair (trace, position) to a label.
- Existential second-order quantification is used to encode the existence of a model.

encoding of $\psi$ as a natural number
**HyperLTL Satisfiability – Membership**

**Lemma**

HyperLTL satisfiability is in $\Sigma_1^1$, i.e., there is a formula

$$\Phi(x) = \exists x_1, \ldots, \exists x_n. \Phi_0(x, x_1, \ldots, x_n)$$

such that

$$\psi \in \text{HyperLTL} \text{ is satisfiable iff } \Phi(\lceil \psi \rceil) \text{ holds.}$$

**Some intuitions:**

- **Minimal size of a model:** every satisfiable HyperLTL formula has a countable model [Finkbeiner, Zimmermann '17]
## Lemma

HyperLTL satisfiability is in $\Sigma^1_1$, i.e., there is a formula

$$\Phi(x) = \exists x_1, \ldots, \exists x_n. \Phi_0(x, x_1, \ldots, x_n)$$

such that

$$\psi \in \text{HyperLTL is satisfiable iff } \Phi(\lceil \psi \rceil) \text{ holds.}$$

### Some intuitions:

- **Minimal size of a model:** every satisfiable HyperLTL formula has a countable model \cite{FinkbeinerZimmermann17}

- **Countable models:** can be seen as functions from $\mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ mapping a pair (trace, position) to a label.
HyperLTL Satisfiability – Membership

Lemma

HyperLTL satisfiability is in $\Sigma_1^1$, i.e., there is a formula

$$\Phi(x) = \exists x_1, \ldots, \exists x_n. \Phi_0(x, x_1, \ldots, x_n)$$

such that

$\psi \in \text{HyperLTL}$ is satisfiable iff $\Phi(\lceil \psi \rceil)$ holds.

Some intuitions:

• Minimal size of a model: every satisfiable HyperLTL formula has a countable model [Finkbeiner, Zimmermann ’17]

• Countable models can be seen as functions from $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ mapping a pair (trace, position) to a label

• Existential second-order quantification is used to encode the existence of a model
Lemma

HyperLTL satisfiability is $\Sigma_1^1$-hard.

**Proof idea:** by reduction from the **recurring tiling problem:**
given a set of tiles, is there a tiling of $\mathbb{N} \times \mathbb{N}$ such that a specific tile occurs infinitely often on the $\mathbb{N} \times 0$ border?
**Lemma**

HyperLTL satisfiability is $\Sigma^1_1$-hard.

**Proof idea:** by reduction from the recurring tiling problem: given a set of tiles, is there a tiling of $\mathbb{N} \times \mathbb{N}$ such that a specific tile occurs infinitely often on the $\mathbb{N} \times 0$ border?

- Each tile is encoded by an atomic proposition
HyperLTL Satisfiability – Hardness

Lemma

HyperLTL satisfiability is $\Sigma^1_1$-hard.

Proof idea: by reduction from the recurring tiling problem: given a set of tiles, is there a tiling of $\mathbb{N} \times \mathbb{N}$ such that a specific tile occurs infinitely often on the $\mathbb{N} \times 0$ border?

- Each tile is encoded by an atomic proposition
- Each row is encoded by a trace
Lemma

HyperLTL satisfiability is $\Sigma_1^1$-hard.

Proof idea: by reduction from the recurring tiling problem: given a set of tiles, is there a tiling of $\mathbb{N} \times \mathbb{N}$ such that a specific tile occurs infinitely often on the $\mathbb{N} \times 0$ border?

- Each tile is encoded by an atomic proposition
- Each row is encoded by a trace
- Rows/traces are ordered vertically using a special atomic proposition $y$ true exactly once in each trace: $y$ is true at time $i$ on the trace representing row $\mathbb{N} \times i$
Proof idea (continued):
Define $\varphi \in \text{HyperLTL}$ expressing that:

1. there is exactly one tile at every position in every trace
2. $y$ is true exactly once on each trace
3. for every $i$, there is a trace with $y$ at position $i$
4. two traces with the same position for $y$ are identical
5. tiles match horizontally
6. tiles match vertically
7. the specified tile occurs infinitely often on the trace where $y$ is true at 0
Proof idea (continued):
Define $\varphi \in \text{HyperLTL}$ expressing that:

1. there is exactly one tile at every position in every trace
HyperLTL Satisfiability – Hardness

Proof idea (continued):
Define $\varphi \in \text{HyperLTL}$ expressing that:

1. there is exactly one tile at every position in every trace
2. $y$ is true exactly once on each trace

\[ \forall \pi. (\neg y_\pi \ U (y_\pi \land X G \neg y_\pi)) \]
Proof idea (continued):
Define $\varphi \in \text{HyperLTL}$ expressing that:

1. there is exactly one tile at every position in every trace
2. $y$ is true exactly once on each trace
3. for every $i$, there is a trace with $y$ at position $i$

$$(\exists \pi. y_\pi) \land (\forall \pi. \exists \pi'. F(y_\pi \land Xy_{\pi'}))$$
Proof idea (continued):
Define $\varphi \in \text{HyperLTL}$ expressing that:

1. there is exactly one tile at every position in every trace
2. $y$ is true exactly once on each trace
3. for every $i$, there is a trace with $y$ at position $i$
4. two traces with the same position for $y$ are identical

$$\forall \pi, \pi'. F(y_\pi \land y_{\pi'}) \rightarrow G \left( \bigwedge_\tau \tau_\pi \leftrightarrow \tau_{\pi'} \right)$$
Proof idea (continued): Define $\varphi \in \text{HyperLTL}$ expressing that:

1. there is exactly one tile at every position in every trace
2. $y$ is true exactly once on each trace
3. for every $i$, there is a trace with $y$ at position $i$
4. two traces with the same position for $y$ are identical
5. tiles match horizontally

\[
\forall \pi. \ G \left( \bigvee_{(\tau,\tau') \in H} \tau_\pi \land X \tau_{\pi'} \right)
\]
HyperLTL Satisfiability – Hardness

Proof idea (continued):
Define $\varphi \in \text{HyperLTL}$ expressing that:

1. there is exactly one tile at every position in every trace
2. $y$ is true exactly once on each trace
3. for every $i$, there is a trace with $y$ at position $i$
4. two traces with the same position for $y$ are identical
5. tiles match horizontally
6. tiles match vertically

$$\forall \pi, \pi'. \ F(\neg \pi \land X \neg \pi') \rightarrow G \left( \bigvee_{(\tau, \tau') \in V} \tau_\pi \land \tau_{\pi'} \right)$$
Proof idea (continued):
Define $\varphi \in \text{HyperLTL}$ expressing that:

1. there is exactly one tile at every position in every trace
2. $y$ is true exactly once on each trace
3. for every $i$, there is a trace with $y$ at position $i$
4. two traces with the same position for $y$ are identical
5. tiles match horizontally
6. tiles match vertically
7. the specified tile occurs infinitely often on the trace where $y$ is true at 0

$$\exists \pi. (y_\pi \land GF(\tau_0)_\pi)$$
Theorem

HyperLTL satisfiability is $\Sigma^1_1$-complete.

Why?

• Every satisfiable HyperLTL formula has a countable model
• Some formulas of HyperCTL* require models of cardinality $2^\aleph_0$ (and this bound is optimal)
<table>
<thead>
<tr>
<th>Theorem</th>
<th>HyperLTL satisfiability is $\Sigma^1_1$-complete.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theorem</td>
<td>HyperCTL$^*$ satisfiability is $\Sigma^2_1$-complete.</td>
</tr>
</tbody>
</table>

Theorem
HyperLTL satisfiability is $\Sigma^1_1$-complete.

Theorem
HyperCTL$^*$ satisfiability is $\Sigma^2_1$-complete.
HyperLTL vs. HyperCTL*

Theorem
HyperLTL satisfiability is $\Sigma_1^1$-complete.

Theorem
HyperCTL* satisfiability is $\Sigma_1^2$-complete.

existential second-order $\rightarrow$ existential third-order arithmetic
### HyperLTL vs. HyperCTL*

**Theorem**
HyperLTL satisfiability is $\Sigma_1^1$-complete.

**Theorem**
HyperCTL* satisfiability is $\Sigma_1^2$-complete.

existential second-order $\rightarrow$ existential third-order arithmetic

**Why?**

- Every satisfiable HyperLTL formula has a countable model
- Some formulas of HyperCTL* require models of cardinality $\mathfrak{c} = |2^\mathbb{N}|$ (and this bound is optimal)
Models of Cardinality at Least $\aleph_0$

The following can be expressed in HyperCTL$^*$:

- Every state is labeled red or black, and 1 or 0.
- Every black state has a (black) 0- and 1-successor.
- Every black path has a copy in the red part.
  $$\forall \pi. (X_{\text{black}}(\pi) \rightarrow \exists \pi'. X_{\text{red}}(\pi') \land G(0_\pi \leftrightarrow 0_{\pi'}))$$
- Two red paths starting in the same state have the same sequence of labels (red paths are initially disjoint).
  $$\forall \pi. G(\text{red}_\pi \rightarrow \forall \pi'. G(0_\pi \leftrightarrow 0_{\pi'}))$$
Models of Cardinality at Least $\mathfrak{c}$

The following can be expressed in HyperCTL*:

- Every state is labeled red or black, and 1 or 0
The following can be expressed in HyperCTL*:

- Every state is labeled red or black, and 1 or 0
- Every black state has a (black) 0- and 1-successor
The following can be expressed in HyperCTL*:

- Every state is labeled red or black, and 1 or 0
- Every black state has a (black) 0- and 1-successor
- Every black path has a copy in the red part

\[
\forall \pi. \left( X \text{black}_\pi \rightarrow \exists \pi'. X(\text{red}_{\pi'} \land G(0_\pi \leftrightarrow 0_{\pi'})) \right)
\]
Models of Cardinality at Least $\mathfrak{c}$

The following can be expressed in Hyper$\text{CTL}^*$:

- Every state is labeled red or black, and 1 or 0
- Every black state has a (black) 0- and 1-successor
- Every black path has a copy in the red part
  \[ \forall \pi. (X \text{black}_\pi \rightarrow \exists \pi'. X(\text{red}_\pi', \land G(0_\pi \leftrightarrow 0_{\pi'}))) \]
- Two red paths starting in the same state have the same sequence of labels (red paths are initially disjoint)
  \[ \forall \pi. G(\text{red}_\pi \rightarrow \forall \pi'. G(0_\pi \leftrightarrow 0_{\pi'})) \]
Models of Cardinality at Least $\beth_0$

The following can be expressed in HyperCTL*:

- Every state is labeled red or black, and 1 or 0
- Every black state has a (black) 0- and 1-successor
- Every black path has a copy in the red part
  $$\forall \pi. (X\ black_\pi \to \exists \pi'. X(\text{red}_\pi', \wedge G(0_\pi \leftrightarrow 0_{\pi'})))$$
- Two red paths starting in the same state have the same sequence of labels (red paths are initially disjoint)
  $$\forall \pi. G(\text{red}_\pi \to \forall \pi'. G(0_\pi \leftrightarrow 0_{\pi'}))$$
Every satisfiable HyperCTL* formula $\varphi$ has a model of cardinality at most $\mathfrak{c}$. 
Matching upper bound

Every satisfiable HyperCTL* formula $\varphi$ has a model of cardinality at most $\mathfrak{c}$.

- Start with an arbitrary model $\mathcal{T}$ of $\varphi$, and Skolem functions witnessing satisfaction.
Every satisfiable HyperCTL* formula $\varphi$ has a model of cardinality at most $\mathfrak{c}$.

- Start with an arbitrary model $\mathcal{T}$ of $\varphi$, and Skolem functions witnessing satisfaction.
- Saturation procedure:

$$
\mathcal{T}_0 = \{s_0 \rightarrow s_1 \rightarrow \cdots\}
$$

$$
\mathcal{T}_{\alpha+1} = \mathcal{T}_\alpha \cup \bigcup f(\bar{x}) \quad \bar{x} \text{ inputs from } \mathcal{T}_\alpha \\
\phantom{\mathcal{T}_{\alpha+1} = \mathcal{T}_\alpha \cup } \text{ for a skolem function } f
$$

$$
\mathcal{T}_\alpha = \bigcup_{\alpha' < \alpha} \mathcal{T}_{\alpha'} \quad \text{for limit ordinals}
$$
Matching upper bound

Every satisfiable HyperCTL* formula $\varphi$ has a model of cardinality at most $\aleph$.  

- Start with an arbitrary model $T$ of $\varphi$, and Skolem functions witnessing satisfaction.  
- Saturation procedure:

$$T_0 = \{ s_0 \rightarrow s_1 \rightarrow \cdots \}$$

$$T_{\alpha + 1} = T_\alpha \cup \bigcup f(\bar{x})$$

$\bar{x}$ inputs from $T_\alpha$

$f$ a skolem function

$$T_\alpha = \bigcup_{\alpha' < \alpha} T_{\alpha'}$$ for limit ordinals
Every satisfiable HyperCTL* formula \( \varphi \) has a model of cardinality at most \( c \).

- Start with an arbitrary model \( \mathcal{T} \) of \( \varphi \), and Skolem functions witnessing satisfaction.
- Saturation procedure:

\[
\mathcal{T}_0 = \{ s_0 \rightarrow s_1 \rightarrow \cdots \}
\]

\[
\mathcal{T}_{\alpha+1} = \mathcal{T}_\alpha \cup \bigcup \left\{ f(\bar{x}) \mid \bar{x} \text{ inputs from } \mathcal{T}_\alpha, f \text{ a skolem function} \right\}
\]

\[
\mathcal{T}_\alpha = \bigcup_{\alpha' < \alpha} \mathcal{T}_{\alpha'} \quad \text{for limit ordinals}
\]
Matching upper bound

Every satisfiable HyperCTL* formula \( \varphi \) has a model of cardinality at most \( \mathfrak{c} \).

- Start with an arbitrary model \( \mathcal{T} \) of \( \varphi \), and Skolem functions witnessing satisfaction.
- Saturation procedure:

\[
\mathcal{T}_0 = \{ s_0 \rightarrow s_1 \rightarrow \cdots \}
\]

\[
\mathcal{T}_{\alpha+1} = \mathcal{T}_\alpha \cup \bigcup f(\bar{x})
\]

\( \bar{x} \) inputs from \( \mathcal{T}_\alpha \)

\( f \) a skolem function

\[
\mathcal{T}_\alpha = \bigcup_{\alpha' < \alpha} \mathcal{T}_{\alpha'} \quad \text{for limit ordinals}
\]
Matching upper bound

Every satisfiable HyperCTL* formula $\varphi$ has a model of cardinality at most $\mathfrak{c}$.

- Start with an arbitrary model $\mathcal{T}$ of $\varphi$, and Skolem functions witnessing satisfaction.
- Saturation procedure:

\[ \mathcal{T}_0 = \{ s_0 \rightarrow s_1 \rightarrow \cdots \} \]

\[ \mathcal{T}_{\alpha+1} = \mathcal{T}_\alpha \cup \bigcup_{\bar{x} \text{ inputs from } \mathcal{T}_\alpha} f(\bar{x}) \]

\[ \mathcal{T}_\alpha = \bigcup_{\alpha' < \alpha} \mathcal{T}_{\alpha'} \quad \text{for limit ordinals} \]
Matching upper bound

Every satisfiable HyperCTL* formula $\varphi$ has a model of cardinality at most $c$.

- Start with an arbitrary model $T$ of $\varphi$, and Skolem functions witnessing satisfaction.
- Saturation procedure:

$$T_0 = \{s_0 \rightarrow s_1 \rightarrow \cdots\}$$

$$T_{\alpha+1} = T_\alpha \cup \bigcup f(\bar{x})$$

\[\bar{x}\text{ inputs from } T_\alpha\]
\[f\text{ a skolem function}\]

$$T_\alpha = \bigcup_{\alpha' < \alpha} T_{\alpha'}$$ for limit ordinals

- Fixpoint at the first uncountable ordinal: $T_{\omega_1} = T_{\omega_1+1}$. 


Matching upper bound

Every satisfiable HyperCTL* formula \( \varphi \) has a model of cardinality at most \( c \).

- Start with an arbitrary model \( \mathcal{T} \) of \( \varphi \), and Skolem functions witnessing satisfaction.
- Saturation procedure:
  \[
  \mathcal{T}_0 = \{ s_0 \rightarrow s_1 \rightarrow \cdots \}
  \]
  \[
  \mathcal{T}_{\alpha+1} = \mathcal{T}_\alpha \cup \bigcup f(\bar{x})
  \]
  \( \bar{x} \) inputs from \( \mathcal{T}_\alpha \)
  \( f \) a skolem function
  \[
  \mathcal{T}_\alpha = \bigcup_{\alpha' < \alpha} \mathcal{T}_{\alpha'} \quad \text{for limit ordinals}
  \]
- Fixpoint at the first uncountable ordinal: \( \mathcal{T}_{\omega_1} = \mathcal{T}_{\omega_1+1} \).
- \( \mathcal{T}_{\omega_1} \) contains at most \( c \) vertices, and \( \mathcal{T}_{\omega_1} \models \varphi \).
Size of Minimal Models

**Theorem**

- Every satisfiable HyperCTL* formula has a model of cardinality at most $\mathfrak{c} = |2^\mathbb{N}|$.
- There is a satisfiable HyperCTL* formula that does not have any model of cardinality less than $\mathfrak{c}$. 
HyperCTL* Satisfiability

**Theorem**

HyperCTL* satisfiability is $\Sigma^2_1$-complete.

$\exists x_1, \ldots, \exists x_n \cdot \Phi_0(x, x_1, \ldots, x_n)$

- third-order variables
- second-order arithmetic
Theorem

HyperCTL* satisfiability is $\Sigma^2_1$-complete.

Upper bound

- Every satisfiable HyperCTL* formula has a model of cardinality at most $c$
  - set of states $= 2^\mathbb{N}$
  - transitions $= \text{subset of } 2^\mathbb{N} \times 2^\mathbb{N}$
  - labeling function $2^\mathbb{N} \rightarrow \mathbb{N}$
HyperCTL* Satisfiability

Theorem

HyperCTL* satisfiability is $\Sigma^2_1$-complete.

Upper bound

- Every satisfiable HyperCTL* formula has a model of cardinality at most $c$
  - set of states $= 2^\mathbb{N}$
  - transitions $= \text{subset of } 2^\mathbb{N} \times 2^\mathbb{N}$
  - labeling function $2^\mathbb{N} \rightarrow \mathbb{N}$
- Use third-order quantifiers to express the existence of a model
HyperCTL* Satisfiability

**Theorem**

HyperCTL* satisfiability is $\Sigma^2_1$-complete.

**Lower bound**

From an existential third-order arithmetic formula $\Phi(x)$ and $n$, construct $\psi$ such that $\mathbb{N} \models \Phi(n)$ iff $\psi$ is satisfiable:
Theorem

HyperCTL* satisfiability is $\Sigma^2_1$-complete.

Lower bound

From an existential third-order arithmetic formula $\Phi(x)$ and $n$, construct $\psi$ such that $\mathbb{N} \models \Phi(n)$ iff $\psi$ is satisfiable:

- $n \in \mathbb{N} \rightsquigarrow$ path with 1 in $n$-th position only;
HyperCTL* Satisfiability

**Theorem**

HyperCTL* satisfiability is $\Sigma^2_1$-complete.

$$\exists x_1, \ldots, \exists x_n. \Phi_0(x, x_1, \ldots, x_n)$$

**Lower bound**

From an existential third-order arithmetic formula $\Phi(x)$ and $n$, construct $\psi$ such that $\mathbb{N} \models \Phi(n)$ iff $\psi$ is satisfiable:

- $n \in \mathbb{N} \rightsquigarrow$ path with 1 in $n$-th position only;
- $A \subseteq \mathbb{N} \rightsquigarrow$ path labeled by characteristic sequence of $A$;
- First and second-order quantifiers $\rightsquigarrow$ path quantifiers.
**HyperCTL* Satisfiability**

**Theorem**

HyperCTL* satisfiability is $\Sigma^2_1$-complete.

$$\exists x_1, \ldots, \exists x_n \cdot \Phi_0(x, x_1, \ldots, x_n)$$

**Lower bound**

From an existential third-order arithmetic formula $\Phi(x)$ and $n$, construct $\psi$ such that $\mathbb{N} \models \Phi(n)$ iff $\psi$ is satisfiable:

- $n \in \mathbb{N} \rightsquigarrow$ path with 1 in $n$-th position only;
- $A \subseteq \mathbb{N} \rightsquigarrow$ path labeled by characteristic sequence of $A$;
- First and second-order quantifiers $\rightsquigarrow$ path quantifiers
- One atomic proposition $p_i$ for each third-order $x_i$.

Existential third-order quantifiers $\rightsquigarrow$ satisfiability
Conclusion

• HyperLTL satisfiability problem is $\Sigma_1^1$-complete, thus highly undecidable.

• HyperCTL$^*$ satisfiability problem is $\Sigma_2^1$-complete, which is infinitely higher than $\Sigma_1^1$ in the hierarchy.

• First (and optimal) bound on the minimal size of models for HyperCTL$^*$:
  - every satisfiable formula has a model with at most $c$ many states
  - there is satisfiable formula that requires $c$ many states.
Conclusion

- HyperLTL satisfiability problem is $\Sigma_1^1$-complete, thus highly undecidable
Conclusion

- HyperLTL satisfiability problem is $\Sigma^1_1$-complete, thus highly undecidable
- HyperCTL* satisfiability problem is $\Sigma^2_1$-complete, which is infinitely higher than $\Sigma^1_1$ in the hierarchy
• HyperLTL satisfiability problem is $\Sigma^1_1$-complete, thus highly undecidable

• HyperCTL* satisfiability problem is $\Sigma^2_1$-complete, which is infinitely higher than $\Sigma^1_1$ in the hierarchy

• First (and optimal) bound on the minimal size of models for HyperCTL*:
  • every satisfiable formula has a model with at most $c$ many states
  • there is satisfiable formula that requires $c$ many states
Conclusion

Other results:

• HyperLTL satisfiability is still $\Sigma_1^1$-complete when restricted to ultimately periodic traces.

• HyperCTL$^*$ satisfiability restricted to countable or finitely branching transition systems is equivalent to the problem of evaluating a second-order arithmetic formula.

• deciding if a HyperLTL formula is equivalent to one with $n$ quantifier alternations is exactly as hard unsatisfiability, i.e. $\Pi_1^1$-complete.
Other results:

- HyperLTL satisfiability is still $\Sigma^1_1$-complete when restricted to ultimately periodic traces.
Other results:

- HyperLTL satisfiability is still $\Sigma^1_1$-complete when restricted to ultimately periodic traces.
- HyperCTL* satisfiability restricted to countable or finitely branching transition systems is equivalent to the problem of evaluating a second-order arithmetic formula.
Other results:

- HyperLTL satisfiability is still $\Sigma^1_1$-complete when restricted to ultimately periodic traces.
- HyperCTL* satisfiability restricted to countable or finitely branching transition systems is equivalent to the problem of evaluating a second-order arithmetic formula.
- deciding if a HyperLTL formula is equivalent to one with $n$ quantifier alternations is exactly as hard unsatisfiability, i.e. $\Pi^1_1$-complete.