Deciding Hyperproperties

Bernd Finkbeiner and Christopher Hahn
Reactive Systems Group
Saarland University, Germany

Highlights of Logic, Games and Automata
Brussels, 06.-09. September 2016

Information Leakage

- Heartbleed - 4.5m patient information leaked
- Goto Fail - encryption of >300m devices broken
- Shellshock - web servers attackable for 22 years

HyperLTL - A Logic for Information-flow Control

[Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez, '14]

- Observational Determinism: "Program appears deterministic to low security users."

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

- Generalized Noninterference: "... additionally low-security outputs may not be altered by injection of high-security inputs."

  $\forall \pi . \forall \pi' . \exists \pi'' .\ \square(HighI_\pi = HighI_{\pi''}) \wedge \square(O_{\pi'} = O_{\pi''})$

HyperLTL - An Extension of LTL

LTL

- logical connectives: $\vee$, $\neg$
  temporal connectives:
  $\Box$ - *globally*
  $\bigcirc$ - *next*
- $\Box a$ is satisfied by $\{a\}^\omega$ as well as $\{a, b\}^\omega$
- $\Box a \wedge \Box \neg a$ is unsatisfiable.
- defines a set of computation traces (trace property)

HyperLTL

- LTL + explicit trace quantifiers:
  Observational Determinism:
  $\forall \pi. \forall \pi'. \quad \Box(I_\pi = I_{\pi'}) \rightarrow \Box(O_\pi = O_{\pi'})$
- $\exists \pi. \exists \pi'. \ \Box a_\pi \wedge \Box \neg a_{\pi'}$
  is satisfiable by $\{\{a\}^\omega, \{b\}^\omega\}$.
- defines a set of sets of computation traces (hyperproperty)

Satisfiability of HyperLTL

Definition (HyperLTL-SAT)

Let φ be an HyperLTL formula. HyperLTL-SAT is the problem to decide whether there exists a non-empty trace set T satisfying φ.

Example (Application)

Two versions of Observational Determinism:

- $\forall \pi . \forall \pi' . \square(I_\pi = I_{\pi'}) \rightarrow \square(O_\pi = O_{\pi'})$
- $\forall \pi . \forall \pi' . (I_\pi = I_{\pi'}) \rightarrow \square(O_\pi = O_{\pi'})$

Which variation is stronger?

Challenge

LTL Satisfiability Solving

- Translate LTL formula into Büchi automaton
- Check the automaton for emptiness
- PSPACE-complete

HyperLTL Satisfiability Solving

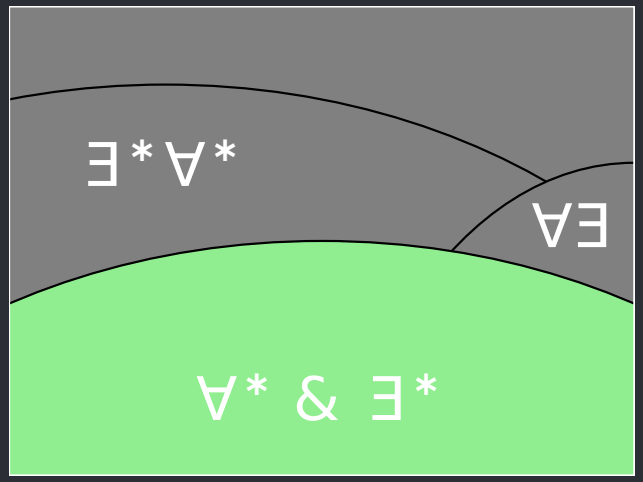
- A Hyperproperty is not necessarily ω-regular
- Standard automata approach cannot be applied

Key Results

[Finkbeiner, H., ’16]

- HyperLTL-SAT is PSPACE-complete for alternation-free formulas
- HyperLTL-SAT is EXPSPACE-complete for $\exists^* \forall^*$ formulas
- HyperLTL-SAT is undecidable for $\forall\exists$ formulas

Outline - Solving HyperLTL-SAT



1. Alternation-free fragments (∀* & ∃*)
2. Alternation starting with existential quantifier (∃*∀*)
3. Alternation starting with universal quantifier (∀*∃*)

Existential Fragment

Theorem

$\exists^*$ *HyperLTL-SAT is* ***PSPACE-complete****.*

Example

$\exists \pi_0 \exists \pi_1.\ \square a_{\pi_0} \wedge \square b_{\pi_0} \wedge \square c_{\pi_0} \wedge \square a_{\pi_1} \wedge \square \neg c_{\pi_1}$

Idea:

Replace indexed atomic propositions with fresh atomic propositions.

$$\square a_0 \wedge \square b_0 \wedge \square c_0 \wedge \square a_1 \wedge \square \neg c_1$$

Existential Fragment

Theorem

$\exists^*$ *HyperLTL-SAT is* ***PSPACE-complete****.*

Example

$\exists \pi_0 \exists \pi_1.\ \square a_{\pi_0} \wedge \square b_{\pi_0} \wedge \square c_{\pi_0} \wedge \square a_{\pi_1} \wedge \square \neg c_{\pi_1}$

Idea:

Replace indexed atomic propositions with fresh atomic propositions.

$$\square a_0 \wedge \square b_0 \wedge \square c_0 \wedge \square a_1 \wedge \square \neg c_1$$

$$t : \{a_0, b_0, c_0, a_1\}^\omega$$

Existential Fragment

Theorem

$\exists^*$ *HyperLTL-SAT is* ***PSPACE-complete****.*

Example

$\exists \pi_0 \exists \pi_1.\ \square a_{\pi_0} \wedge \square b_{\pi_0} \wedge \square c_{\pi_0} \wedge \square a_{\pi_1} \wedge \square \neg c_{\pi_1}$

Idea:

Replace indexed atomic propositions with fresh atomic propositions.

$$\square a_0 \wedge \square b_0 \wedge \square c_0 \wedge \square a_1 \wedge \square \neg c_1$$

$$t : \{a_0, b_0, c_0, a_1\}^\omega$$

$$T = \{\{a, b, c\}^\omega, \{a\}^\omega\}$$

Universal Fragment

Theorem

$\forall^*$ *HyperLTL-SAT is* ***PSPACE-complete****.*

Example

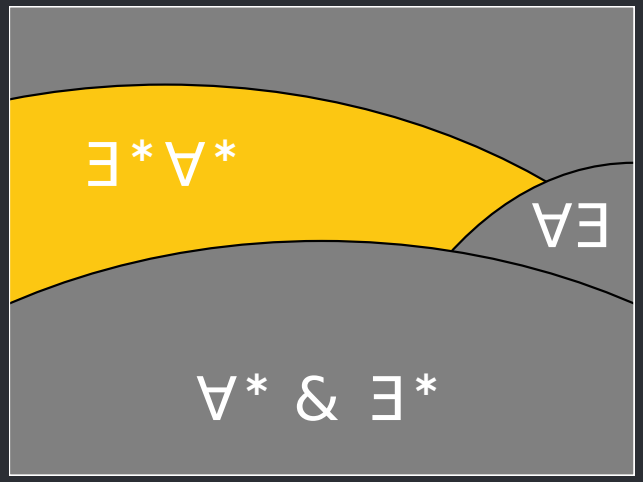
$$\forall\pi\forall\pi'.\,\square b_\pi \wedge \square\neg b_{\pi'} \quad \equiv \quad \square b \wedge \square\neg b$$

$$\Downarrow \quad \Downarrow \qquad\qquad\qquad\qquad \Downarrow$$

$$t \quad t \qquad\qquad\qquad\qquad \text{unsatisfiable}$$

Idea: Discard indexes from indexed propositions

Outline - Solving HyperLTL-SAT



1. Alternation-free fragments (∀* & ∃*)
2. Alternation starting with existential quantifier (∃*∀*)
3. Alternation starting with universal quantifier (∀*∃*)

$\exists^*\forall^*$ HyperLTL-SAT

Lemma

For every $\exists\pi_1 \ldots \exists\pi_n \forall\pi'_1 \ldots \forall\pi'_m . \varphi$ HyperLTL formula, there exists an equisatisfiable $\exists^*$ HyperLTL formula.

Example

$\exists\pi_0 \exists\pi_1 \forall\pi'_0 \forall\pi'_1 . (\square a_{\pi'_0} \wedge \square b_{\pi'_1}) \wedge (\square c_{\pi_0} \wedge \square d_{\pi_1})$

Idea: Unroll universal quantifiers

$$\begin{aligned}\exists\pi_0 \exists\pi_1 . \; & (\square a_{\pi_0} \wedge \square b_{\pi_0}) \wedge (\square c_{\pi_0} \wedge \square d_{\pi_1}) \\ & \wedge (\square a_{\pi_1} \wedge \square b_{\pi_0}) \wedge (\square c_{\pi_0} \wedge \square d_{\pi_1}) \\ & \wedge (\square a_{\pi_0} \wedge \square b_{\pi_1}) \wedge (\square c_{\pi_0} \wedge \square d_{\pi_1}) \\ & \wedge (\square a_{\pi_1} \wedge \square b_{\pi_1}) \wedge (\square c_{\pi_0} \wedge \square d_{\pi_1})\end{aligned}$$

Complexity of $\exists^*\forall^*$ HyperLTL-SAT

Theorem

$\exists^*\forall^*$ *HyperLTL-SAT is EXPSPACE-complete.*

- Let n be the number of existential quantifier and m be the number of universal quantifier.
  Unrolling results in formula of size $O(n^m)$.
- Hardness follows from an encoding of an EXPSPACE-bounded Turing machine in this fragment.

$\exists^* \forall^b$ HyperLTL-SAT

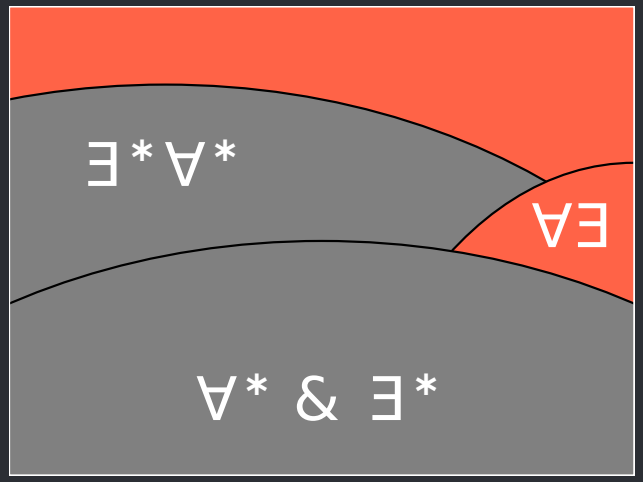
Theorem

Bounded $\exists^* \forall^b$ HyperLTL-SAT is PSPACE-complete.

Observation: In practice, many properties of interest quantify universally over pairs of traces

$\forall \pi. \forall \pi'. \square(I_\pi = I_{\pi'}) \rightarrow \square(O_\pi = O_{\pi'})$
$\forall \pi. \forall \pi'. (I_\pi = I_{\pi'}) \rightarrow \square(O_\pi = O_{\pi'})$
$\forall \pi. \forall \pi'. \exists \pi''. \ \square(HighI_\pi = HighI_{\pi''}) \wedge \square(O_{\pi'} = O_{\pi''})$

Outline - Solving HyperLTL-SAT



1. Alternation-free fragments (∀* & ∃*)
2. Alternation starting with existential quantifier (∃*∀*)
3. Alternation starting with universal quantifier (∀*∃*)

The Power of ∀∃ - Encoding of PCP

Theorem

∀∃ *HyperLTL-SAT is undecidable.*

- Can give a ∀∃ HyperLTL formula, which is only satisfied by an infinite trace set:

$$\forall \pi \exists \pi' .\ a_{\pi'} \tag{1}$$
$$\wedge\ \square(a_\pi \rightarrow \bigcirc \square \neg a_\pi) \tag{2}$$
$$\wedge\ \square(a_\pi \rightarrow \bigcirc a_{\pi'}) \tag{3}$$

- Encoding of Posts Correspondence Problem (PCP) in this fragment.

Summary & Conclusion

$\exists^*$	$\forall^*$	$\exists^*\forall^*$	*b*-Bounded $\exists^*\forall^*$	$\forall\exists$
PSpace-complete	**PSpace-complete**	**EXPSpace-complete**	**PSpace-complete**	undecidable

- Satisfiability of alternation-free formulas is decidable
- Implication and equivalence of alternation-free formulas are decidable
- Full logic is undecidable:
  HyperLTL is much more powerful than LTL

Christopher Hahn: hahn@react.uni-saarland.de

Appendix

Bibliography

[Clarkson, Schneider, ’10] Clarkson, M. R., and F. B. Schneider. "Hyperproperties." Journal of Computer Security 18.6 (2010): 1157-1210.

[Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez, ’14] Clarkson, M. R., Finkbeiner, B., Koleini, M., Micinski, K. K., Rabe, M. N., & Sánchez, C. (2014, April). Temporal logics for hyperproperties. In International Conference on Principles of Security and Trust (pp. 265-284).

[Finkbeiner, H., ’16] Bernd Finkbeiner and Christopher Hahn. Deciding hyperproperties. In International Conference on Concurrency Theory (2016).

Picture: http://russia-insider.com/sites/insider/files/20110226_bbd001_0.jpg

HyperLTL Syntax

Syntax

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

$\varphi ::= a_\pi \mid \neg\varphi \mid \varphi \vee \varphi \mid \bigcirc \varphi \mid \varphi\ \mathcal{U}\ \varphi$

- Quantifier Prefix with arbitrary alternation
- Then quantifier-free LTL formula with trace variables
- $\wedge$, $\rightarrow$, $\leftrightarrow$, $\square$, $\diamond$ derived in the usual way
- $X_\pi = X_{\pi'}$ syntactic sugar for $\bigwedge_{x \in X} (x_\pi \leftrightarrow x_{\pi'})$

Example

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

$\forall \pi.\ \forall \pi'.\ \square(on_\pi \leftrightarrow on_{\pi'})$

HyperLTL Semantics

Semantics w.r.t. Trace Environment $\Pi : \mathit{Var} \mapsto \mathit{TR}$

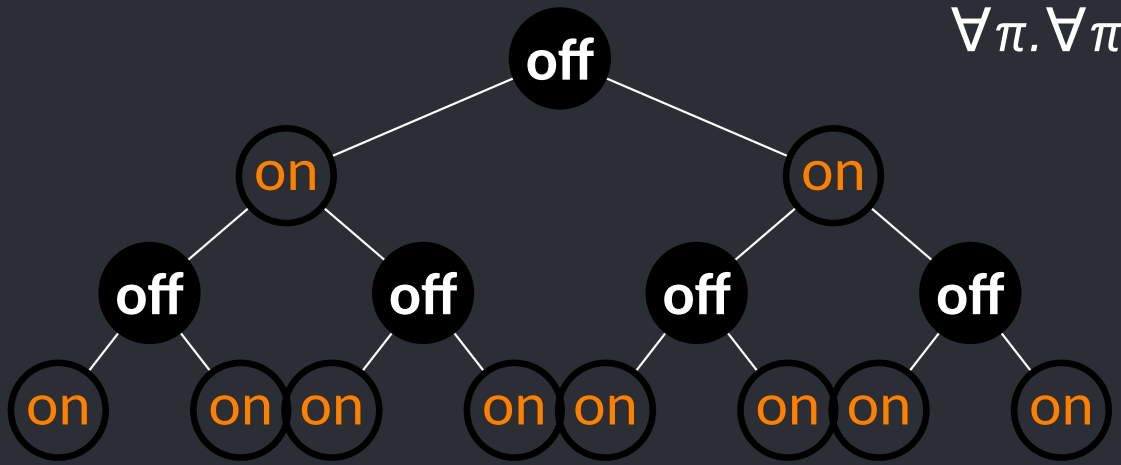
$\Pi \models_T \forall \pi.\varphi \quad \mathit{iff} \quad \mathit{for\ all}\ t \in T, s.t. \Pi[\pi \mapsto t] \models_T \varphi$

$\Pi \models_T a_\pi \quad \mathit{iff} \quad a \in \Pi(\pi)[0]$

$\Pi \models_T \square\varphi \quad \mathit{iff} \quad \forall i \geq 0 : \Pi[i, \infty] \models_T \varphi$

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

$\forall \pi. \forall \pi'.\ \square(on_\pi \longleftrightarrow on_{\pi'})$



HyperLTL Semantics

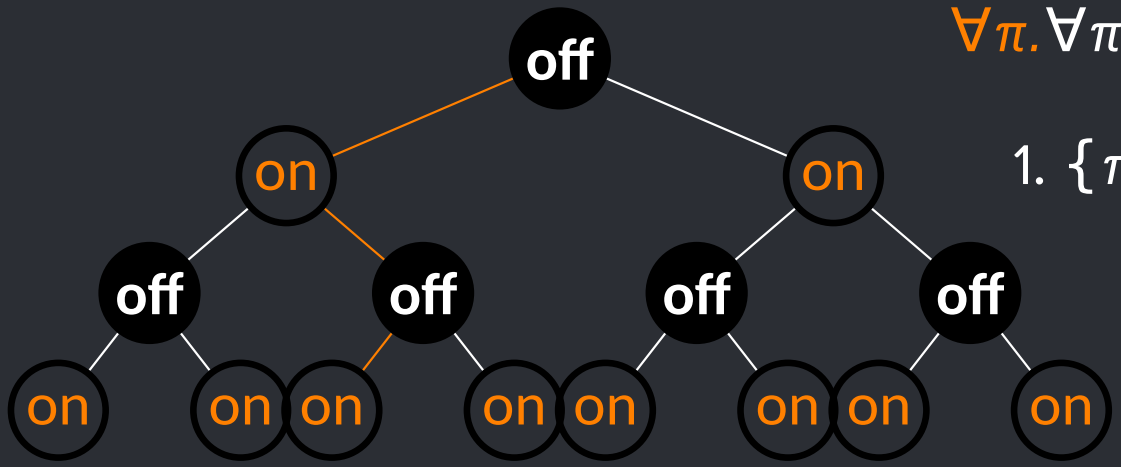
Semantics w.r.t. Trace Environment $\Pi : \mathit{Var} \mapsto \mathit{TR}$

$\Pi \models_T \forall \pi. \varphi$ *iff* *for all* $t \in T$, *s.t.* $\Pi[\pi \mapsto t] \models_T \varphi$

$\Pi \models_T a_\pi$ *iff* $a \in \Pi(\pi)[0]$

$\Pi \models_T \square \varphi$ *iff* $\forall i \geq 0 : \Pi[i, \infty] \models_T \varphi$

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



$\forall \pi. \forall \pi'. \square(on_\pi \leftrightarrow on_{\pi'})$

1. $\{\pi \mapsto t\} \models_M \forall \pi'. \square(\ldots)$

HyperLTL Semantics

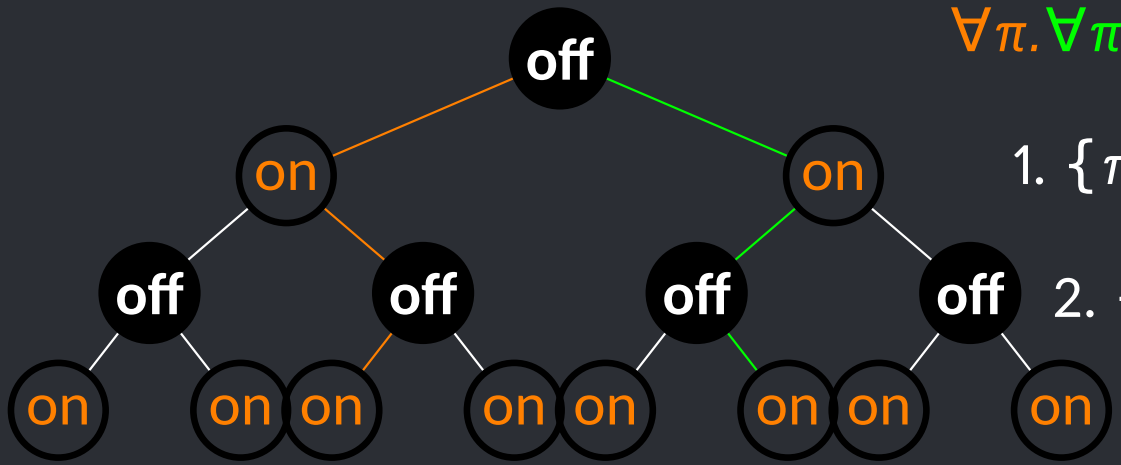
Semantics w.r.t. Trace Environment $\Pi : \mathit{Var} \mapsto \mathit{TR}$

$\Pi \models_T \forall \pi.\varphi \quad \mathit{iff} \quad \mathit{for\ all}\ t \in T, s.t. \Pi[\pi \mapsto t] \models_T \varphi$

$\Pi \models_T a_\pi \quad \mathit{iff} \quad a \in \Pi(\pi)[0]$

$\Pi \models_T \Box\varphi \quad \mathit{iff} \quad \forall i \geq 0 : \Pi[i, \infty] \models_T \varphi$

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



$\forall \pi.\forall \pi'.\ \Box(on_\pi \longleftrightarrow on_{\pi'})$

1. $\{\pi \mapsto t\} \models_M \forall \pi'.\ \Box(\ldots)$
2. $\{\pi \mapsto t, \pi' \mapsto t'\} \models_T \Box(\ldots)$

HyperLTL Semantics

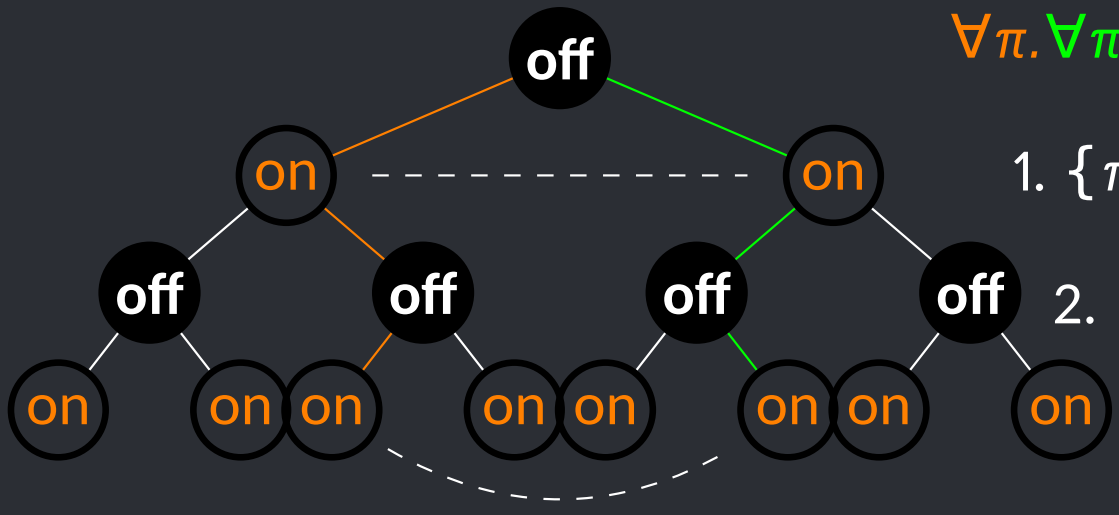
Semantics w.r.t. Trace Environment $\Pi : \mathit{Var} \mapsto \mathit{TR}$

$\Pi \models_T \forall \pi.\varphi$ *iff* *for all* $t \in T$, *s.t.* $\Pi[\pi \mapsto t] \models_T \varphi$

$\Pi \models_T a_\pi$ *iff* $a \in \Pi(\pi)[0]$

$\Pi \models_T \square \varphi$ *iff* $\forall i \geq 0 : \Pi[i, \infty] \models_T \varphi$

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



$\forall \pi.\forall \pi'.\ \square(on_\pi \leftrightarrow on_{\pi'})$

1. $\{\pi \mapsto t\} \models_M \forall \pi'.\ \square(\ldots)$

2. $\{\pi \mapsto t, \pi' \mapsto t'\} \models_T \square(\ldots)$

3. $\forall i \geq 0 :\ \{\pi \mapsto t[i, \infty],\ \pi' \mapsto t'[i, \infty]\} \models_T on_\pi \leftrightarrow on_{\pi'}$

Application: Implication Checking of Quantifier-alternation-free Hyperproperties

ψ implies φ?

Check the negation $\neg(\psi \rightarrow \varphi)$ for unsatisfiability.

Example

To determine whether the HyperLTL formula $\forall \pi_0 \ldots \forall \pi_n.\ \psi$ implies the HyperLTL formula $\forall \pi'_0 \ldots \forall \pi'_m.\ \varphi$, we check the $\exists^n \forall^m$ formula $\exists \pi_1 \ldots \pi_n \forall \pi'_0 \ldots \pi'_m.\ \psi \wedge \neg\varphi$ for unsatisfiability.

Theorem

Implication Checking of quantifier-alternation-free HyperLTL formulas is EXPSPACE-complete.

Encoding of Posts Correspondence Problem

Example PCP instance:

I	II	III
a	ab	bba
baa	aa	bb

Encoding of Posts Correspondence Problem

Example PCP solution:

III	II	III	I
bba	ab	bba	a
bb	aa	bb	baa

HyperLTL encoding:

1. exists a “solution”-trace π_s,
   where top matches bottom

Encoding of Posts Correspondence Problem

Example PCP solution:

III	II	III	I
bba	ab	bba	a
bb	aa	bb	baa

π_s

b b a a b b b a a # # ⋯

b b a a b b b a a # # ⋯

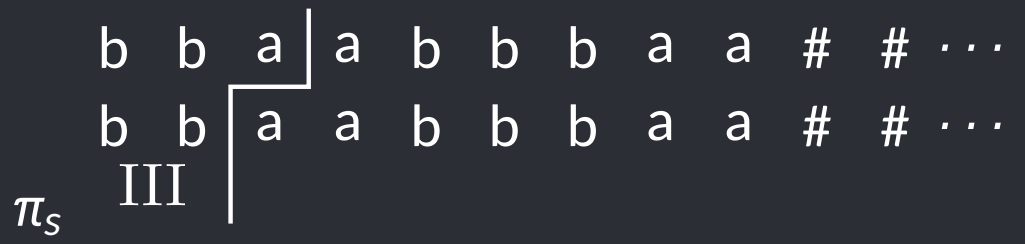
HyperLTL encoding:

1. exists a “solution”-trace π_s, where top matches bottom

Encoding of Posts Correspondence Problem

Example PCP solution:

III	II	III	I
bba	ab	bba	a
bb	aa	bb	baa



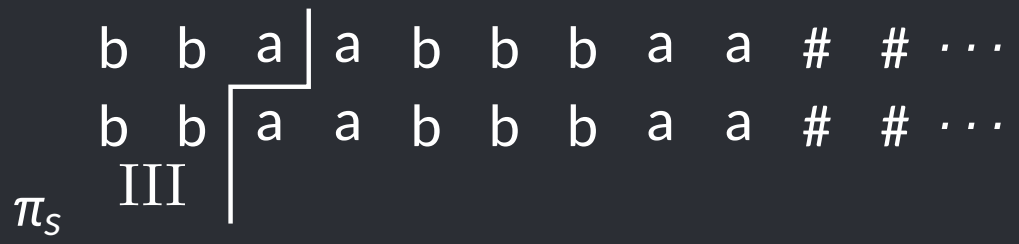
HyperLTL encoding:

1. exists a “solution”-trace π_s, where top matches bottom
2. every trace starts with a valid stone

Encoding of Posts Correspondence Problem

Example PCP solution:

III	II	III	I
bba	ab	bba	a
bb	aa	bb	baa



HyperLTL encoding:

1. exists a “solution”-trace π_s, where top matches bottom
2. every trace starts with a valid stone
3. for every trace, there exists another without the first stone

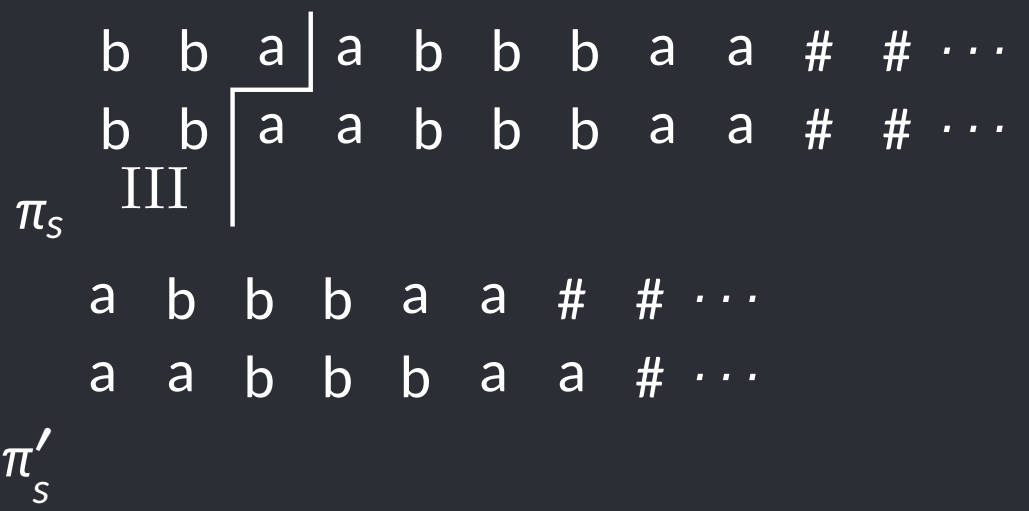
Encoding of Posts Correspondence Problem

Example PCP solution:

III	II	III	I
bba	ab	bba	a
bb	aa	bb	baa

HyperLTL encoding:

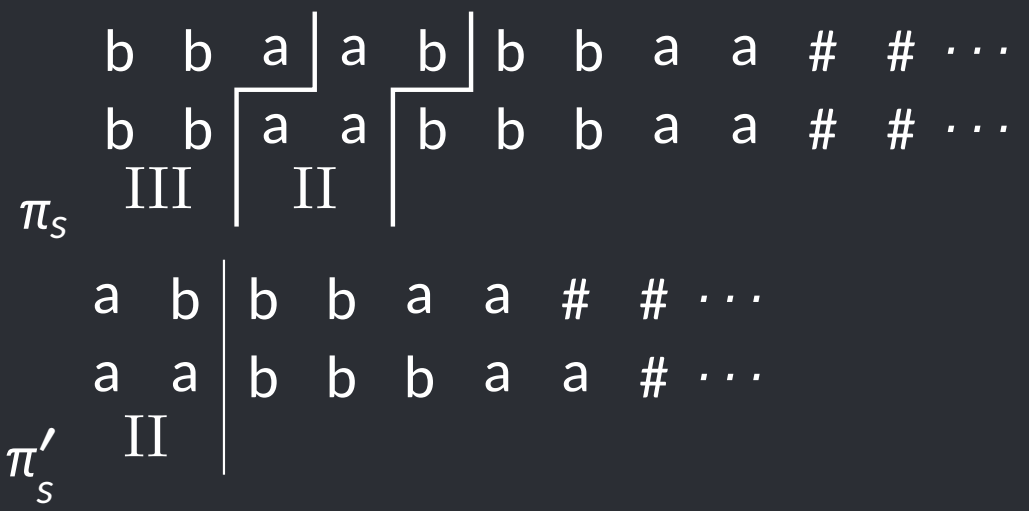
1. exists a “solution”-trace π_s, where top matches bottom
2. every trace starts with a valid stone
3. for every trace, there exists another without the first stone



Encoding of Posts Correspondence Problem

Example PCP solution:

III	II	III	I
bba	ab	bba	a
bb	aa	bb	baa



HyperLTL encoding:

1. exists a “solution”-trace π_s, where top matches bottom
2. every trace starts with a valid stone
3. for every trace, there exists another without the first stone

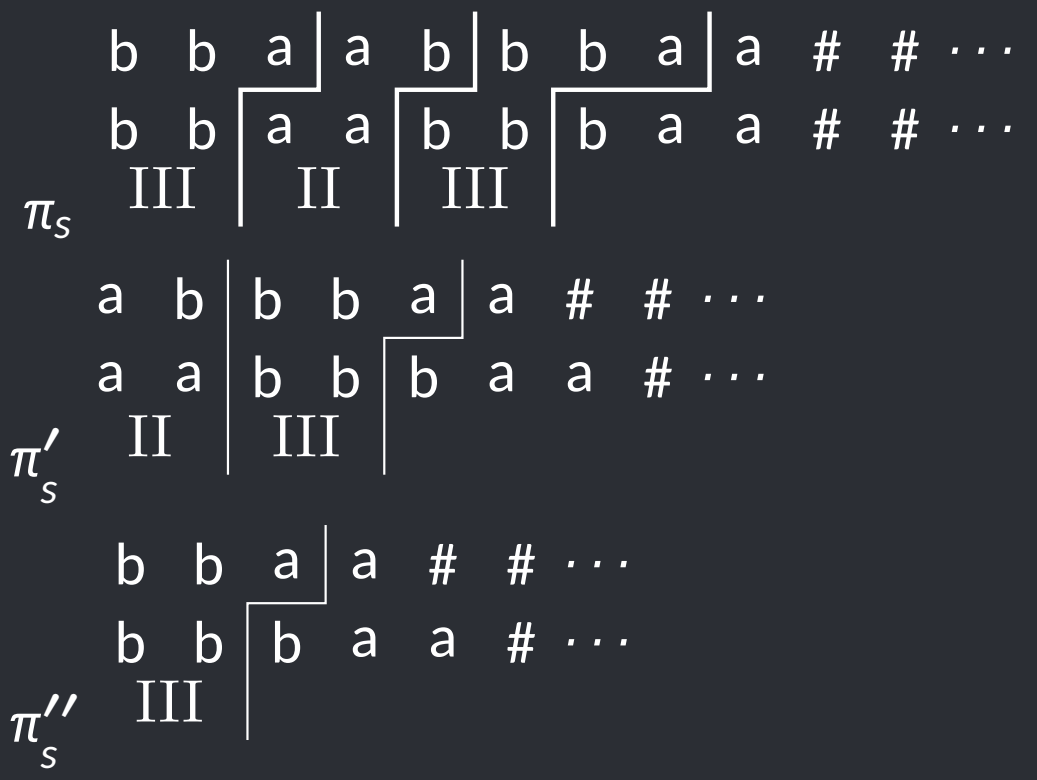
Encoding of Posts Correspondence Problem

Example PCP solution:

III	II	III	I
bba	ab	bba	a
bb	aa	bb	baa

HyperLTL encoding:

1. exists a “solution”-trace π_s, where top matches bottom
2. every trace starts with a valid stone
3. for every trace, there exists another without the first stone



Encoding of Posts Correspondence Problem

Example PCP solution:

III	II	III	I
bba	ab	bba	a
bb	aa	bb	baa

HyperLTL encoding:

1. exists a “solution”-trace π_s, where top matches bottom
2. every trace starts with a valid stone
3. for every trace, there exists another without the first stone