Deciding Hyperproperties Combined with Functional Specifications

Raven Beutner[1], David Carral[3], Bernd Finkbeiner[1], **Jana Hofmann**[1], Markus Krötzsch[2]

[1] *CISPA Helmholtz Center for Information Security, Germany*

[2] *TU Dresden, Germany*

[3] *LIRMM, Inria, University of Montpellier, CNRS, France*

LICS 2022

Haifa, 2 August 2022

Overview

Overview

- Hyperproperties describe many information flow policies like noninterference

Overview

- Hyperproperties describe many information flow policies like noninterference
- HyperLTL satisfiability is highly undecidable for $\forall^*\exists^*$ formulas

Overview

- Hyperproperties describe many information flow policies like noninterference
- HyperLTL satisfiability is highly undecidable for $\forall^*\exists^*$ formulas
- 2 new perspectives: temporal safety/liveness + split in functional property and hyperproperty

Overview

- Hyperproperties describe many information flow policies like noninterference
- HyperLTL satisfiability is highly undecidable for $\forall^*\exists^*$ formulas
- 2 new perspectives: temporal safety/liveness + split in functional property and hyperproperty

		no LTL spec.	with LTL spec.
Temporal Safety	complete fragment	coRE [Thm. 3.7]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\bigcirc^*$	NEXP [Thm. 3.12]	NEXP [Thm. 3.12]
	$\forall^*\exists^*.\square$	NEXP [Lem. 3.13]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\square(\bigcirc^*)$	coRE [Lem. 3.10]	Σ_1^1 [Thm. 3.11]
Temporal Liveness	complete fragment	Σ_1^1 [Thm. 4.2]	Σ_1^1 [Thm. 4.2]
	$\forall\exists^*$. det-liveness	trivial [Prop. 4.15]	Σ_1^1 [Cor. 4.16]
	$\forall\exists^*.\Diamond(\bigcirc^*)$	NP [Lem. 4.4]	dec. [Thm. 4.6]
	$\forall^*\exists^*.\Diamond\wedge\cdots\wedge\Diamond$	NP [Lem. 4.4]	Σ_1^1 [Thm. 4.12]

Overview

- Hyperproperties describe many information flow policies like noninterference
- HyperLTL satisfiability is highly undecidable for $\forall^*\exists^*$ formulas
- 2 new perspectives: temporal safety/liveness + split in functional property and hyperproperty

		no LTL spec.	with LTL spec.
Temporal Safety	complete fragment	coRE [Thm. 3.7]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\bigcirc^*$	NEXP [Thm. 3.12]	NEXP [Thm. 3.12]
	$\forall^*\exists^*.\square$	NEXP [Lem. 3.13]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\square(\bigcirc^*)$	coRE [Lem. 3.10]	Σ_1^1 [Thm. 3.11]
Temporal Liveness	complete fragment	Σ_1^1 [Thm. 4.2]	Σ_1^1 [Thm. 4.2]
	$\forall\exists^*$. det-liveness	trivial [Prop. 4.15]	Σ_1^1 [Cor. 4.16]
	$\forall\exists^*.\Diamond(\bigcirc^*)$	NP [Lem. 4.4]	dec. [Thm. 4.6]
	$\forall^*\exists^*.\Diamond\wedge\cdots\wedge\Diamond$	NP [Lem. 4.4]	Σ_1^1 [Thm. 4.12]

Overview

- Hyperproperties describe many information flow policies like noninterference
- HyperLTL satisfiability is highly undecidable for $\forall^*\exists^*$ formulas
- 2 new perspectives: temporal safety/liveness + split in functional property and hyperproperty

		no LTL spec.	with LTL spec.
Temporal Safety	complete fragment	coRE [Thm. 3.7]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\bigcirc^*$	NEXP [Thm. 3.12]	NEXP [Thm. 3.12]
	$\forall^*\exists^*.\square$	NEXP [Lem. 3.13]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\square(\bigcirc^*)$	coRE [Lem. 3.10]	Σ_1^1 [Thm. 3.11]
Temporal Liveness	complete fragment	Σ_1^1 [Thm. 4.2]	Σ_1^1 [Thm. 4.2]
	$\forall\exists^*$. det-liveness	trivial [Prop. 4.15]	Σ_1^1 [Cor. 4.16]
	$\forall\exists^*.\Diamond(\bigcirc^*)$	NP [Lem. 4.4]	dec. [Thm. 4.6]
	$\forall^*\exists^*.\Diamond\wedge\cdots\wedge\Diamond$	NP [Lem. 4.4]	Σ_1^1 [Thm. 4.12]

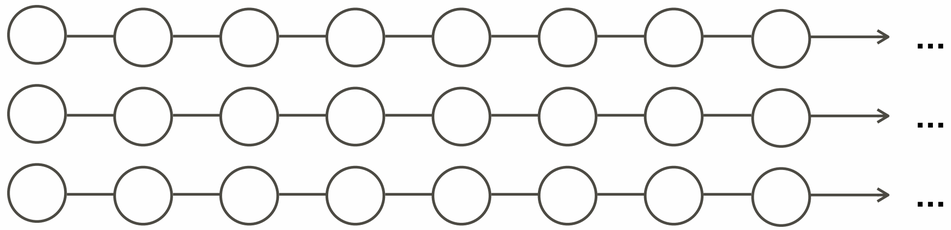
- Sound algorithm for largest models for $\forall\exists^*$ HyperLTL

Hyperproperties

Hyperproperties

Trace property P: set of traces

system

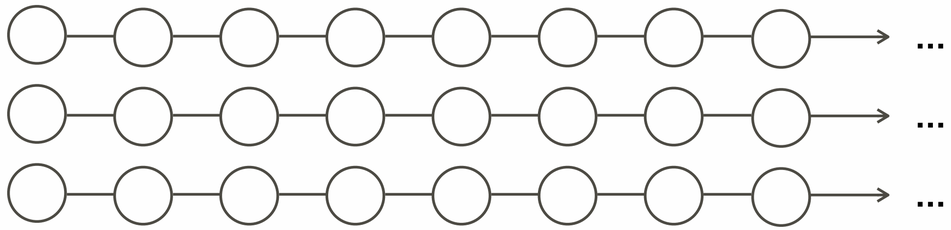


trace $\in$ P?

Hyperproperties

Trace property P: set of traces

system



trace ∈ P?

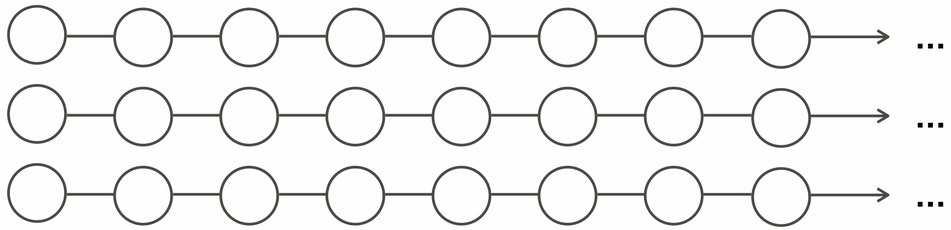
✔

Hyperproperties

Trace property P: set of traces

system

trace $\in$ P?

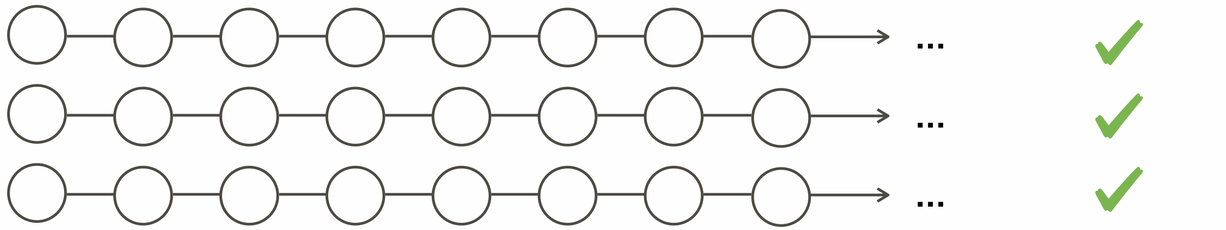


Hyperproperties

Trace property P: set of traces

system

trace $\in$ P?

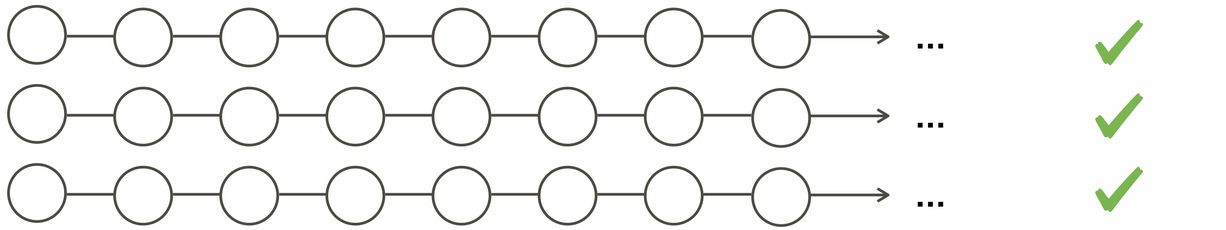


Hyperproperties

Trace property P: set of traces

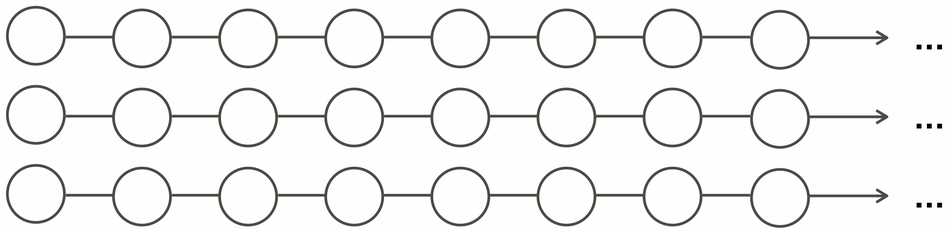
system

trace ∈ P?



Hyperproperty[1] H: set of sets of traces

system



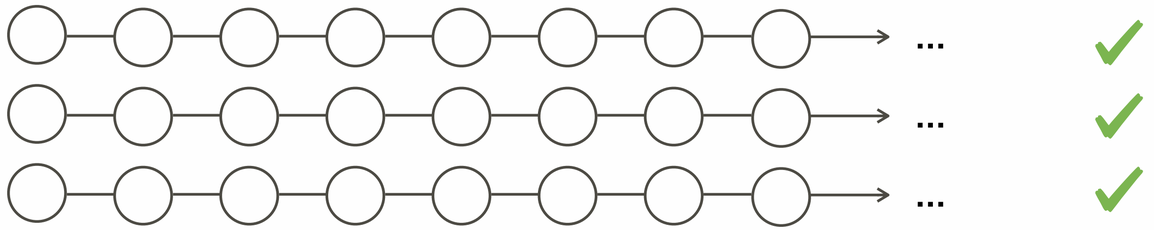
[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Hyperproperties

Trace property P: set of traces

system

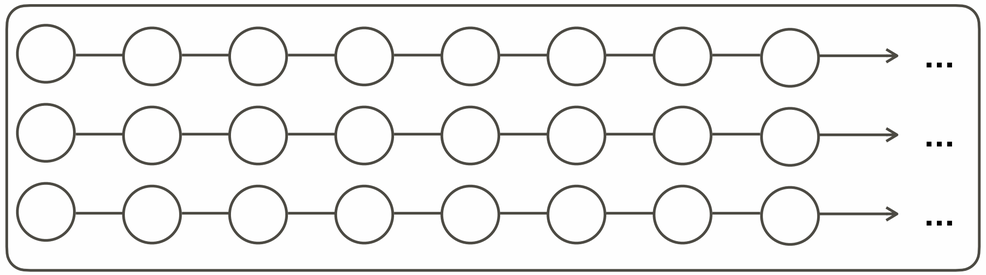
trace ∈ P?



Hyperproperty[1] H: set of sets of traces

system

system ∈ H?



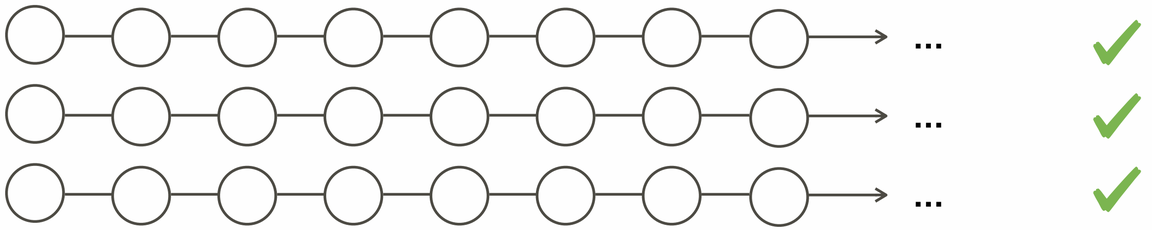
[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Hyperproperties

Trace property P: set of traces

system

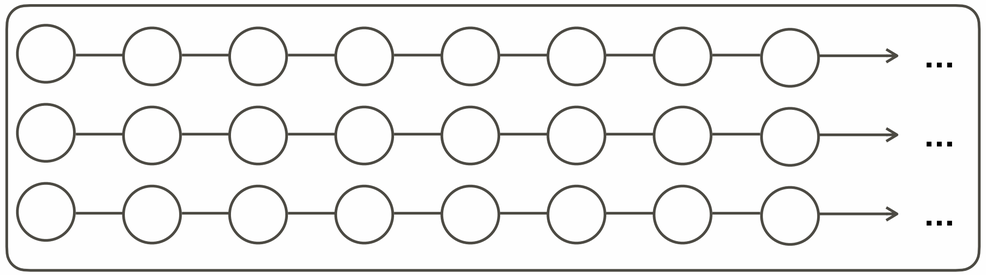
trace ∈ P?



Hyperproperty[1] H: set of sets of traces

system

system ∈ H?

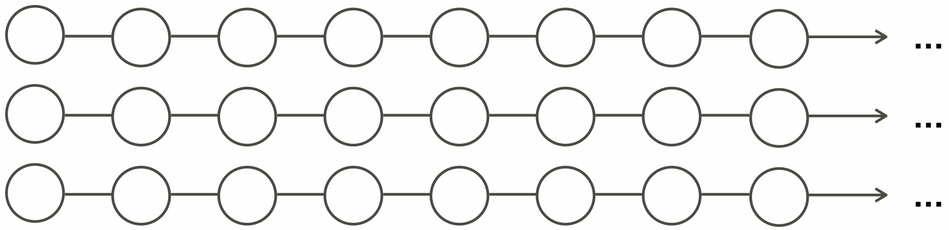


[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Hyperproperties

Trace property P: set of traces

system



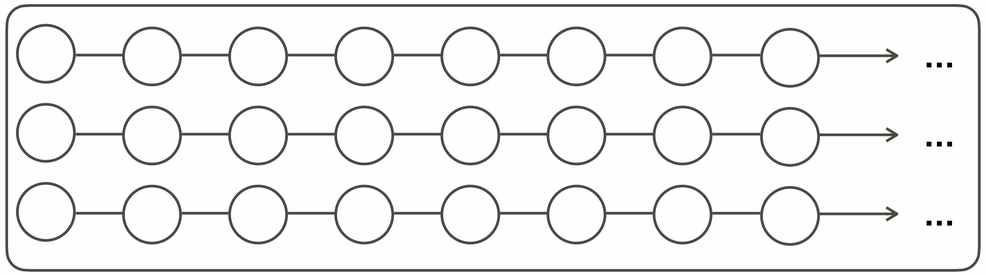
trace ∈ P?

✓ ✓ ✓

Functional properties: safety, liveness, …

Hyperproperty[1] H: set of sets of traces

system



system ∈ H?

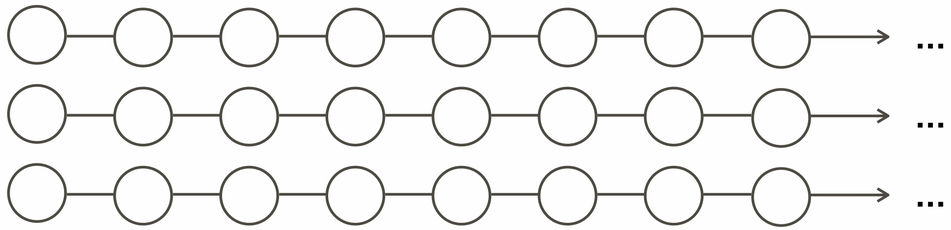
✓

[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Hyperproperties

Trace property P: set of traces

system



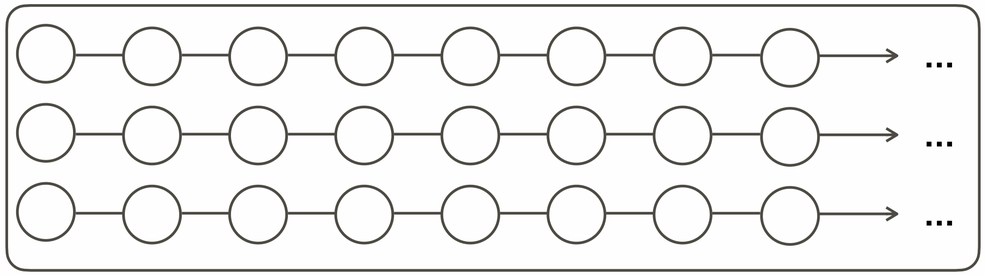
trace ∈ P?

✓ ✓ ✓

Functional properties: safety, liveness, …

Hyperproperty[1] H: set of sets of traces

system



system ∈ H?

✓

Relational properties:

"Does a change of inputs lead to a change of outputs?"

[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

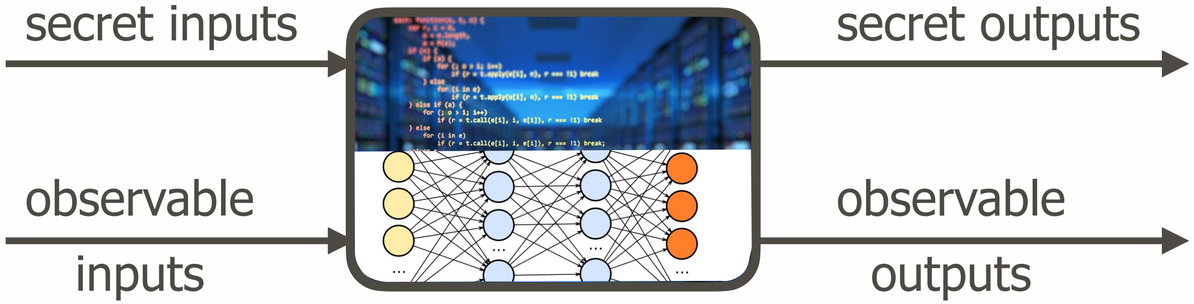
HyperLTL[1]

"Secret inputs do not interfere with publicly observable inputs and outputs."

[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

HyperLTL[1]

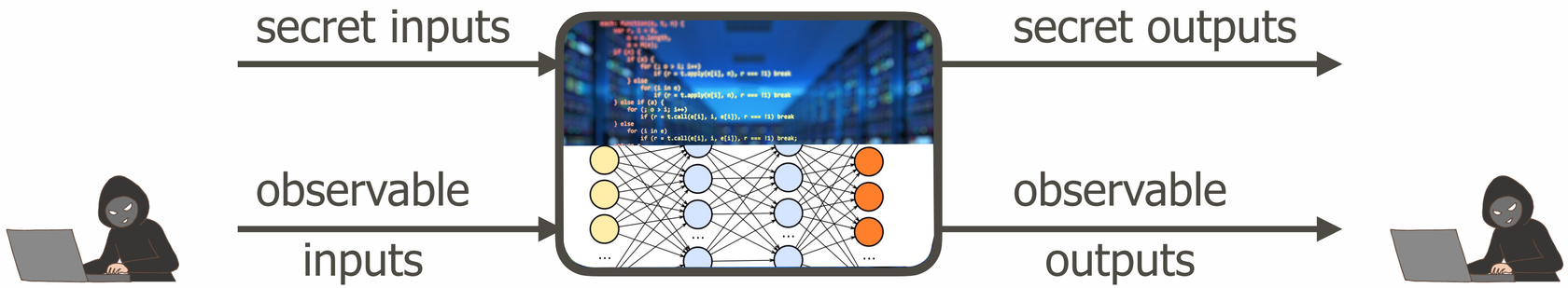
"Secret inputs do not interfere with publicly observable inputs and outputs."



[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

HyperLTL[1]

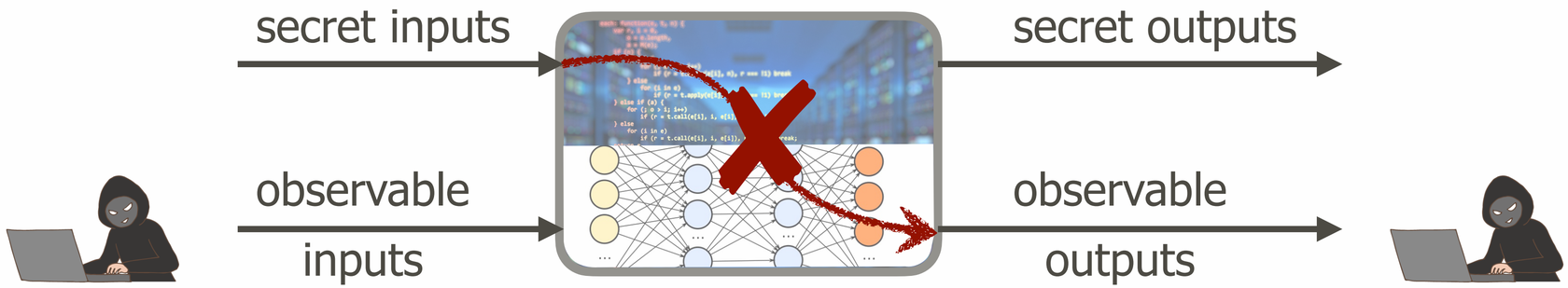
"Secret inputs do not interfere with publicly observable inputs and outputs."



[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

HyperLTL[1]

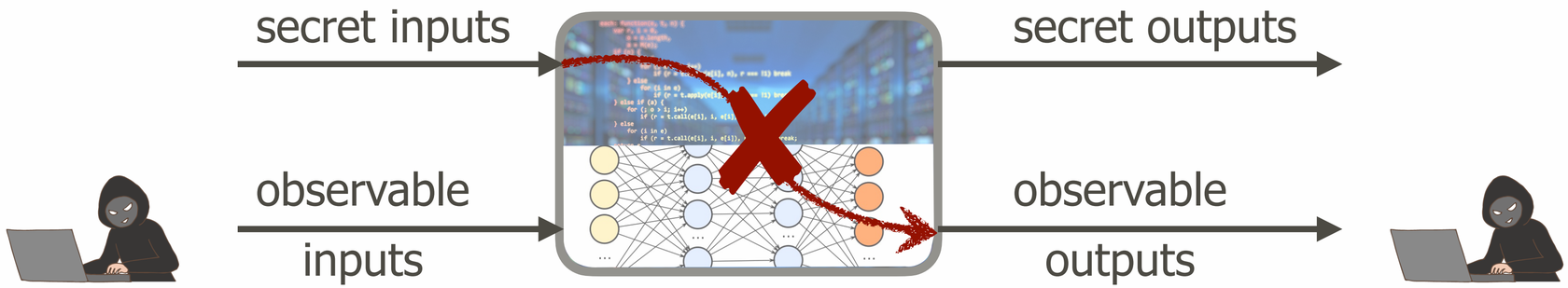
"Secret inputs do not interfere with publicly observable inputs and outputs."



[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

HyperLTL[1]

"Secret inputs do not interfere with publicly observable inputs and outputs."

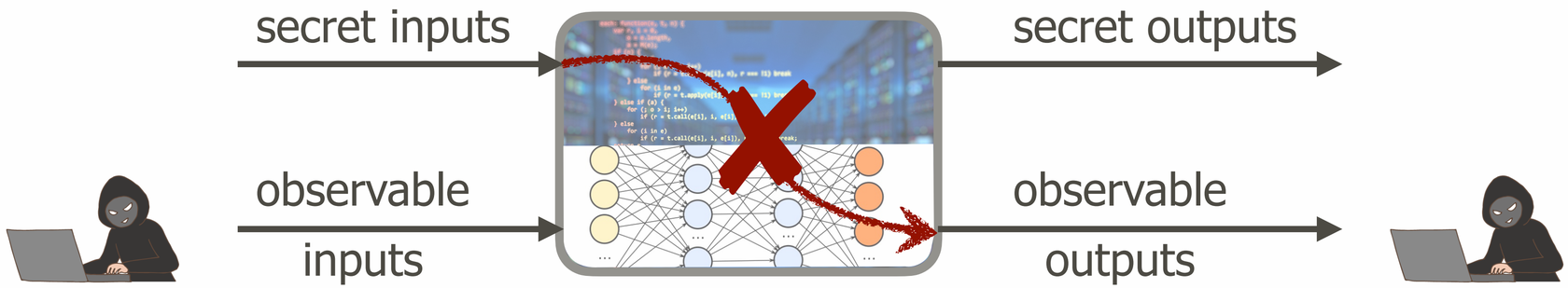


Generalized noninterference in HyperLTL:

[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

HyperLTL[1]

"Secret inputs do not interfere with publicly observable inputs and outputs."



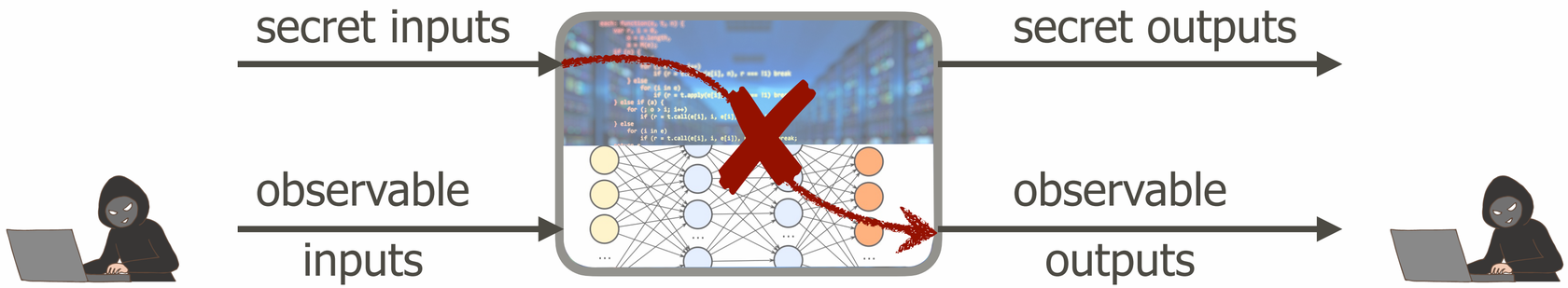
Generalized noninterference in HyperLTL:

$$\forall\pi.\forall\pi'.\exists\pi''.\ \square(secretIn_{\pi} = secretIn_{\pi''}) \wedge$$
$$\square(observableOut_{\pi'} = observableOut_{\pi''})$$

[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

HyperLTL[1]

"Secret inputs do not interfere with publicly observable inputs and outputs."



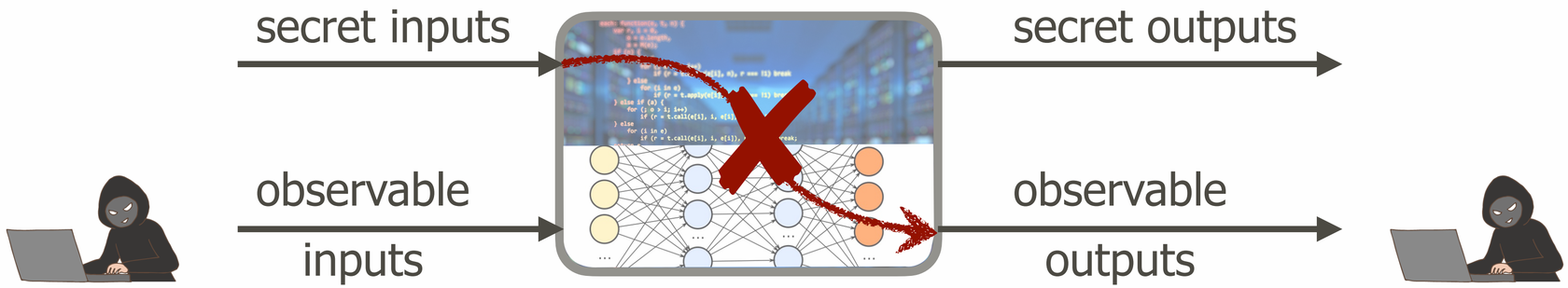
Generalized noninterference in HyperLTL:

$$\forall\pi.\forall\pi'.\exists\pi''.\ \square(secretIn_{\pi} = secretIn_{\pi''}) \wedge \underbrace{\square(observableOut_{\pi'} = observableOut_{\pi''})}_{\text{LTL with indexed atomic propositions}}$$

[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

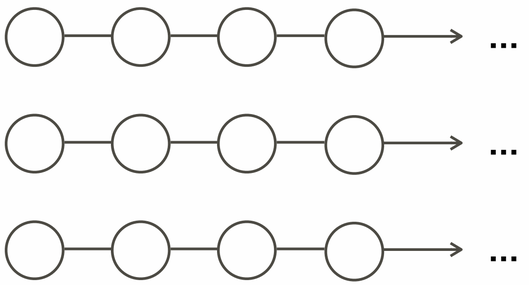
HyperLTL[1]

"Secret inputs do not interfere with publicly observable inputs and outputs."



Generalized noninterference in HyperLTL:

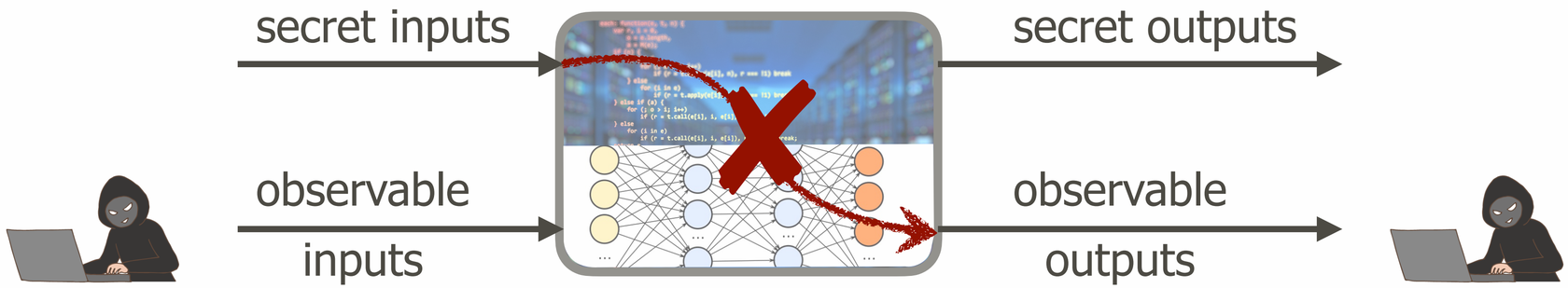
$$\forall\pi.\forall\pi'.\exists\pi''.\ \square(secretIn_{\pi} = secretIn_{\pi''}) \wedge \underbrace{\square(observableOut_{\pi'} = observableOut_{\pi''})}_{\text{LTL with indexed atomic propositions}}$$



[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

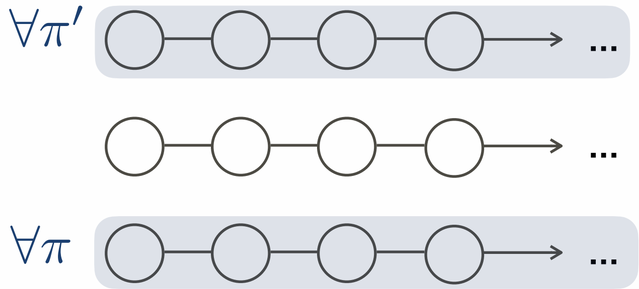
HyperLTL[1]

"Secret inputs do not interfere with publicly observable inputs and outputs."



Generalized noninterference in HyperLTL:

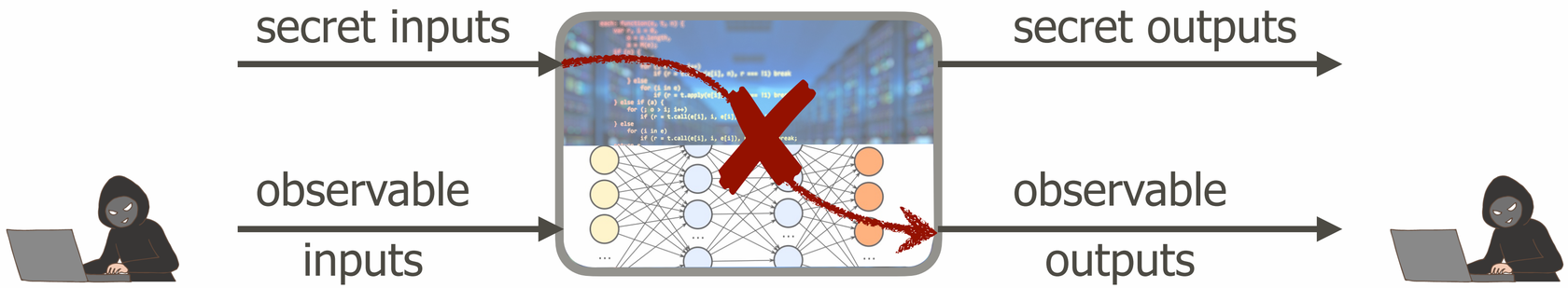
$$\forall\pi.\forall\pi'.\exists\pi''.\ \square(secretIn_{\pi} = secretIn_{\pi''}) \wedge \underbrace{\square(observableOut_{\pi'} = observableOut_{\pi''})}_{\text{LTL with indexed atomic propositions}}$$



[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

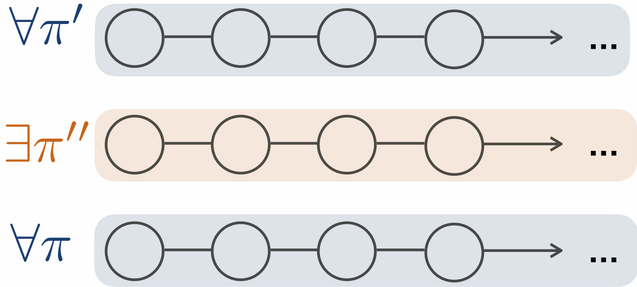
HyperLTL[1]

"Secret inputs do not interfere with publicly observable inputs and outputs."



Generalized noninterference in HyperLTL:

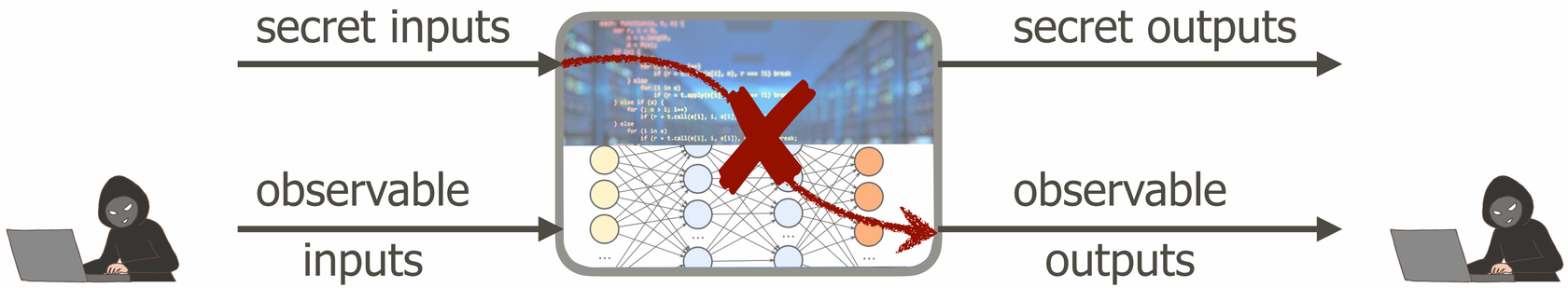
$$\forall\pi.\forall\pi'.\exists\pi''.\ \square(secretIn_{\pi} = secretIn_{\pi''}) \wedge \underbrace{\square(observableOut_{\pi'} = observableOut_{\pi''})}_{\text{LTL with indexed atomic propositions}}$$



[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

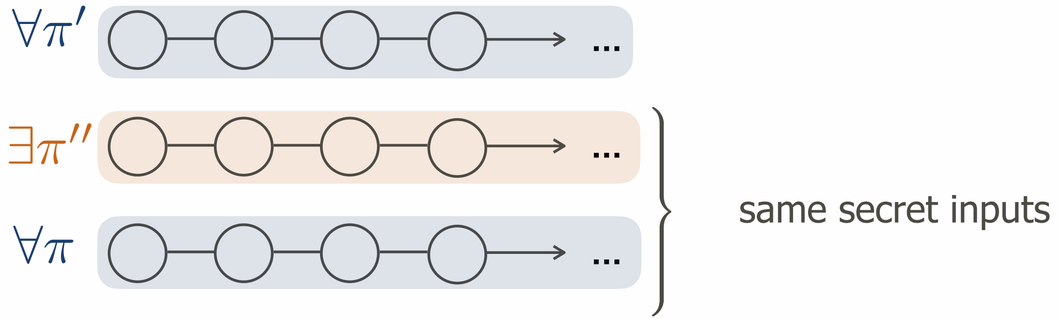
HyperLTL[1]

"Secret inputs do not interfere with publicly observable inputs and outputs."



Generalized noninterference in HyperLTL:

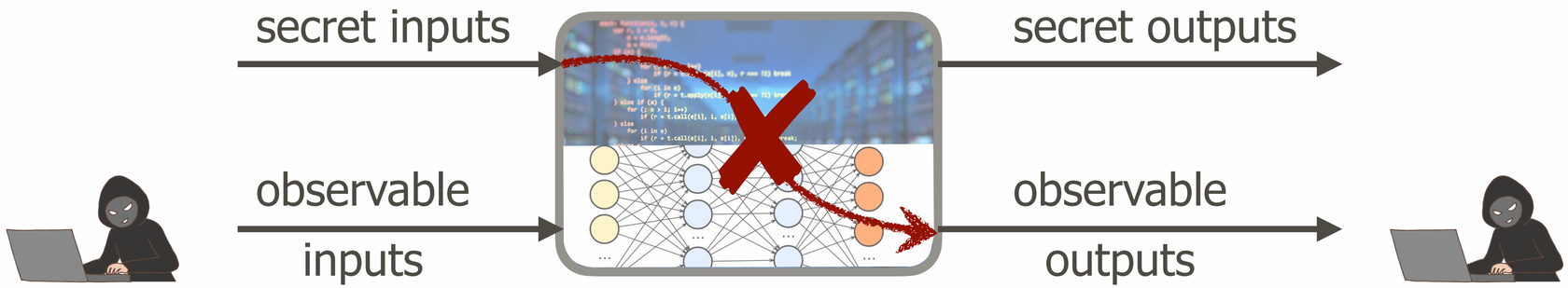
$$\forall\pi.\forall\pi'.\exists\pi''.\ \square(secretIn_\pi = secretIn_{\pi''}) \wedge \underbrace{\square(observableOut_{\pi'} = observableOut_{\pi''})}_{\text{LTL with indexed atomic propositions}}$$



[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

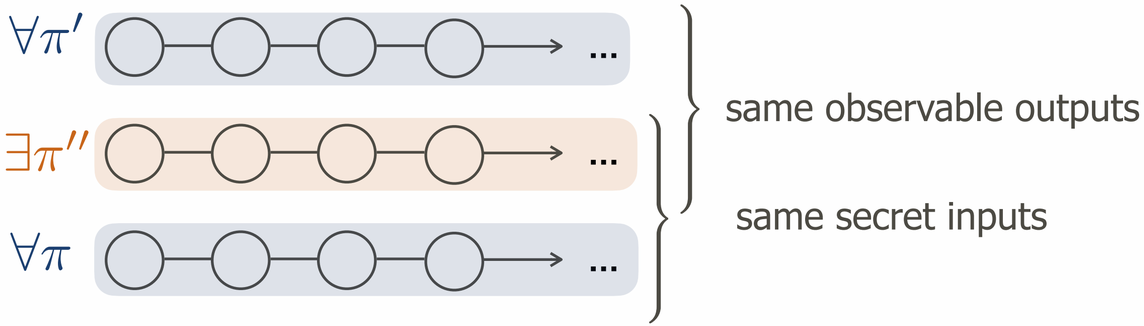
HyperLTL[1]

"Secret inputs do not interfere with publicly observable inputs and outputs."



Generalized noninterference in HyperLTL:

$$\forall\pi.\forall\pi'.\exists\pi''.\ \square(secretIn_\pi = secretIn_{\pi''}) \wedge \underbrace{\square(observableOut_{\pi'} = observableOut_{\pi''})}_{\text{LTL with indexed atomic propositions}}$$



[1] Clarkson, Finkbeiner, Koleini, Micinski, Rabe, Sánchez. *Temporal Logics for Hyperproperties*. POST 2014.

Satisfiability of ∀*∃* HyperLTL

Satisfiability of ∀* ∃* HyperLTL

$\exists^*\forall^*$: decidable[1], in general: highly undecidable (in Σ^1_1)[2]

[2] Mascle, Zimmermann. *The Keys to Decidable HyperLTL Satisfiability: Small Models or Very Simple Formulas*. CSL 2020.

[1] Finkbeiner, Hahn. *Deciding Hyperproperties*. CONCUR 2016.

Satisfiability of ∀* ∃* HyperLTL

$\exists^*\forall^*$: decidable[1], in general: highly undecidable (in Σ^1_1)[2]

Formulas with $\forall\exists$ quantifier alternation get undecidable very quickly. Undecidable are:

[2] Mascle, Zimmermann. *The Keys to Decidable HyperLTL Satisfiability: Small Models or Very Simple Formulas*. CSL 2020.

[1] Finkbeiner, Hahn. *Deciding Hyperproperties*. CONCUR 2016.

Satisfiability of ∀* ∃* HyperLTL

$\exists^*\forall^*$: decidable[1], in general: highly undecidable (in Σ^1_1)[2]

Formulas with $\forall\exists$ quantifier alternation get undecidable very quickly. Undecidable are:

$\forall^2\exists^*$ + only $\Box$ and $\Diamond$, not nested[2]

[2] Mascle, Zimmermann. *The Keys to Decidable HyperLTL Satisfiability: Small Models or Very Simple Formulas*. CSL 2020.

[1] Finkbeiner, Hahn. *Deciding Hyperproperties*. CONCUR 2016.

Satisfiability of ∀* ∃* HyperLTL

$\exists^*\forall^*$: decidable[1], in general: highly undecidable (in Σ^1_1)[2]

Formulas with $\forall\exists$ quantifier alternation get undecidable very quickly. Undecidable are:

$\forall^2\exists^*$ + only $\square$ and $\diamond$, not nested[2]

$\forall\exists^*$ + arbitrary temporal operators[1]

[2] Mascle, Zimmermann. *The Keys to Decidable HyperLTL Satisfiability: Small Models or Very Simple Formulas*. CSL 2020.

[1] Finkbeiner, Hahn. *Deciding Hyperproperties*. CONCUR 2016.

Satisfiability of ∀* ∃* HyperLTL

$\exists^*\forall^*$: decidable[1], in general: highly undecidable (in Σ^1_1)[2]

Formulas with $\forall\exists$ quantifier alternation get undecidable very quickly. Undecidable are:

$\forall^2\exists^*$ + only $\square$ and $\diamond$, not nested[2]

$\forall\exists^*$ + arbitrary temporal operators[1]

⇨ No easier fragments obtained from syntactic restrictions

[2] Mascle, Zimmermann. *The Keys to Decidable HyperLTL Satisfiability: Small Models or Very Simple Formulas*. CSL 2020.

[1] Finkbeiner, Hahn. *Deciding Hyperproperties*. CONCUR 2016.

Satisfiability of ∀* ∃* HyperLTL

$\exists^*\forall^*$: decidable[1], in general: highly undecidable (in Σ^1_1)[2]

Formulas with $\forall\exists$ quantifier alternation get undecidable very quickly. Undecidable are:

$\forall^2\exists^*$ + only $\Box$ and $\Diamond$, not nested[2]

$\forall\exists^*$ + arbitrary temporal operators[1]

⇨ No easier fragments obtained from syntactic restrictions

⇨ New perspectives on HyperLTL satisfiability?

[2] Mascle, Zimmermann. *The Keys to Decidable HyperLTL Satisfiability: Small Models or Very Simple Formulas*. CSL 2020.

[1] Finkbeiner, Hahn. *Deciding Hyperproperties*. CONCUR 2016.

New Perspectives on ∀*∃* HyperLTL SAT

2 approaches:

New Perspectives on ∀*∃* HyperLTL SAT

2 approaches:

1) Split hyperproperty into trace property + hyperproperty

New Perspectives on ∀* ∃* HyperLTL SAT

2 approaches:

1) Split hyperproperty into trace property + hyperproperty

Trace property in LTL: describes functional behavior

New Perspectives on ∀*∃* HyperLTL SAT

2 approaches:

1) Split hyperproperty into trace property + hyperproperty

Trace property in LTL: describes functional behavior

Hyperproperty in HyperLTL: simple relational property

New Perspectives on ∀* ∃* HyperLTL SAT

2 approaches:

1) Split hyperproperty into trace property + hyperproperty

Trace property in LTL: describes functional behavior, e.g.: safety properties

Hyperproperty in HyperLTL: simple relational property, e.g.: privacy properties

New Perspectives on ∀* ∃* HyperLTL SAT

2 approaches:

1) Split hyperproperty into trace property + hyperproperty

Trace property in LTL: describes functional behavior, e.g.: safety properties

Hyperproperty in HyperLTL: simple relational property, e.g.: privacy properties

2) Semantic notion of temporal safety and temporal liveness

New Perspectives on ∀* ∃* HyperLTL SAT

2 approaches:

1) Split hyperproperty into trace property + hyperproperty

Trace property in LTL: describes functional behavior, e.g.: safety properties

Hyperproperty in HyperLTL: simple relational property, e.g.: privacy properties

2) Semantic notion of temporal safety and temporal liveness

Idea: especially safety properties have algorithmic advantages

New Perspectives on $\forall^* \exists^*$ HyperLTL SAT

2 approaches:

1) Split hyperproperty into trace property + hyperproperty

Trace property in LTL: describes functional behavior, e.g.: safety properties

Hyperproperty in HyperLTL: simple relational property, e.g.: privacy properties

2) Semantic notion of temporal safety and temporal liveness

Idea: especially safety properties have algorithmic advantages

$\forall^* \exists^* . \psi$ — safety / liveness LTL formula

Temporal Safety vs Hypersafety

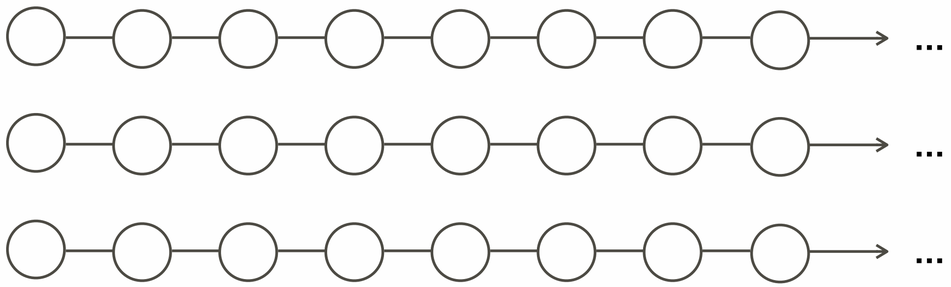
Temporal Safety vs Hypersafety

Hypersafety[1]: Does every counterexample have a "finite reason" for being a counterexample?

[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Temporal Safety vs Hypersafety

Hypersafety[1]: Does every counterexample have a "finite reason" for being a counterexample?



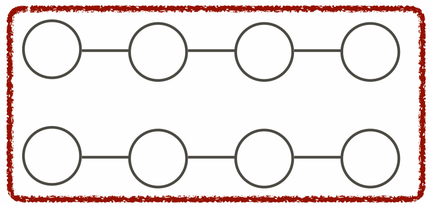
trace set $\in$ H?

✗

[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Temporal Safety vs Hypersafety

Hypersafety[1]: Does every counterexample have a "finite reason" for being a counterexample?

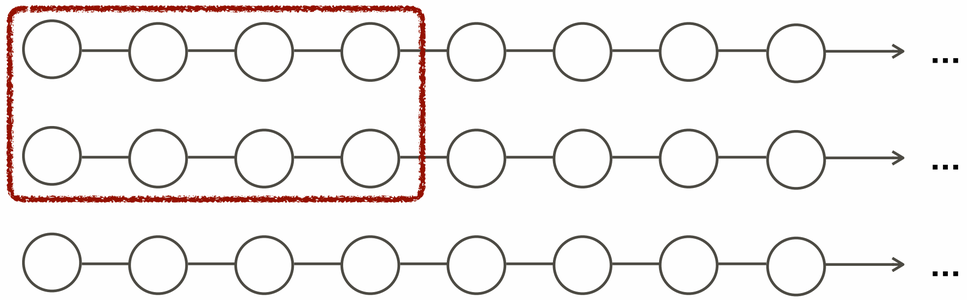


trace set $\in$ H?

[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Temporal Safety vs Hypersafety

Hypersafety[1]: Does every counterexample have a "finite reason" for being a counterexample?



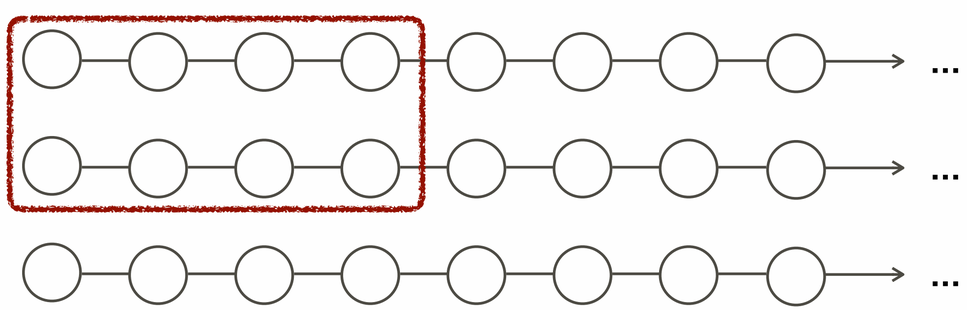
trace set $\in$ H?

✗

[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Temporal Safety vs Hypersafety

Hypersafety[1]: Does every counterexample have a "finite reason" for being a counterexample?



trace set $\in$ H?

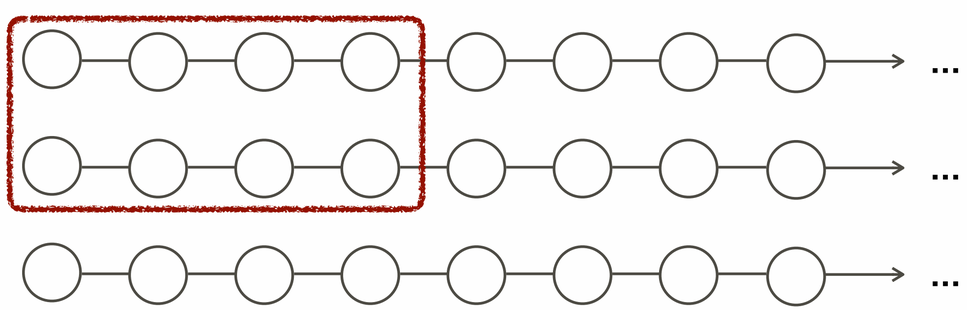
✗

Hypersafety does not help with satisfiability:

[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Temporal Safety vs Hypersafety

Hypersafety[1]: Does every counterexample have a "finite reason" for being a counterexample?



trace set $\in$ H?

✗

Hypersafety does not help with satisfiability:

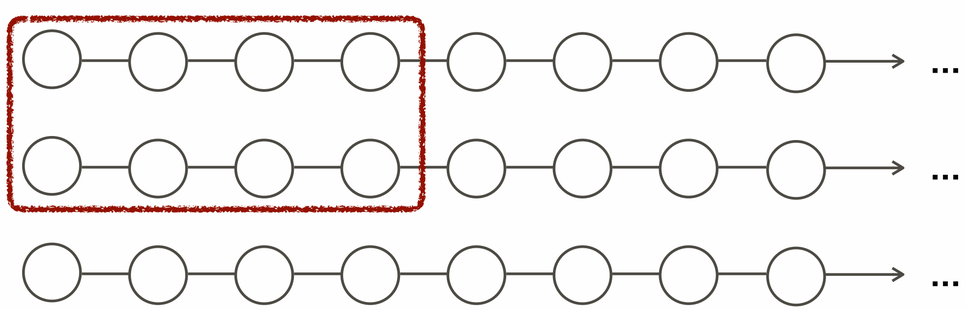
1) Whether HyperLTL formula φ is hypersafety is highly undecidable (in Π^1_1)[2]

[2] Finkbeiner, Haas, Torfah. *Canonical Representations of k-Safety Hyperproperties*. CSF 2019.

[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Temporal Safety vs Hypersafety

Hypersafety[1]: Does every counterexample have a "finite reason" for being a counterexample?



trace set $\in$ H?

✗

Hypersafety does not help with satisfiability:

1) Whether HyperLTL formula φ is hypersafety is highly undecidable (in Π^1_1)[2]

2) If we know that φ is hypersafety, deciding SAT is in PSPACE - no harder than LTL

[2] Finkbeiner, Haas, Torfah. *Canonical Representations of k-Safety Hyperproperties*. CSF 2019.

[1] Clarkson, Schneider. *Hyperproperties*. CSF 2008.

Results

		no LTL spec.	with LTL spec.
Temporal Safety	complete fragment	coRE [Thm. 3.7]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\bigcirc^*$	NEXP [Thm. 3.12]	NEXP [Thm. 3.12]
	$\forall^*\exists^*.\square$	NEXP [Lem. 3.13]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\square(\bigcirc^*)$	coRE [Lem. 3.10]	Σ_1^1 [Thm. 3.11]
Temporal Liveness	complete fragment	Σ_1^1 [Thm. 4.2]	Σ_1^1 [Thm. 4.2]
	$\forall\exists^*.$ det-liveness	trivial [Prop. 4.15]	Σ_1^1 [Cor. 4.16]
	$\forall\exists^*.\Diamond(\bigcirc^*)$	NP [Lem. 4.4]	dec. [Thm. 4.6]
	$\forall^*\exists^*.\Diamond\wedge\cdots\wedge\Diamond$	NP [Lem. 4.4]	Σ_1^1 [Thm. 4.12]

Results

Reduction to first-order logic

		no LTL spec.	with LTL spec.
Temporal Safety	complete fragment	coRE [Thm. 3.7]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\bigcirc^*$	NEXP [Thm. 3.12]	NEXP [Thm. 3.12]
	$\forall^*\exists^*.\Box$	NEXP [Lem. 3.13]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\Box(\bigcirc^*)$	coRE [Lem. 3.10]	Σ_1^1 [Thm. 3.11]
Temporal Liveness	complete fragment	Σ_1^1 [Thm. 4.2]	Σ_1^1 [Thm. 4.2]
	$\forall\exists^*.$ det-liveness	trivial [Prop. 4.15]	Σ_1^1 [Cor. 4.16]
	$\forall\exists^*.\Diamond(\bigcirc^*)$	NP [Lem. 4.4]	dec. [Thm. 4.6]
	$\forall^*\exists^*.\Diamond\wedge\cdots\wedge\Diamond$	NP [Lem. 4.4]	Σ_1^1 [Thm. 4.12]

Results

Reduction to first-order logic

⇨ FO SAT solving becomes applicable

		no LTL spec.	with LTL spec.
Temporal Safety	complete fragment	coRE [Thm. 3.7]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\bigcirc^*$	NEXP [Thm. 3.12]	NEXP [Thm. 3.12]
	$\forall^*\exists^*.\Box$	NEXP [Lem. 3.13]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\Box(\bigcirc^*)$	coRE [Lem. 3.10]	Σ_1^1 [Thm. 3.11]
Temporal Liveness	complete fragment	Σ_1^1 [Thm. 4.2]	Σ_1^1 [Thm. 4.2]
	$\forall\exists^*.$ det-liveness	trivial [Prop. 4.15]	Σ_1^1 [Cor. 4.16]
	$\forall\exists^*.\Diamond(\bigcirc^*)$	NP [Lem. 4.4]	dec. [Thm. 4.6]
	$\forall^*\exists^*.\Diamond\wedge\cdots\wedge\Diamond$	NP [Lem. 4.4]	Σ_1^1 [Thm. 4.12]

Results

Reduction to first-order logic

⇨ FO SAT solving becomes applicable

		no LTL spec.	with LTL spec.
Temporal Safety	complete fragment	coRE [Thm. 3.7]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\bigcirc^*$	NEXP [Thm. 3.12]	NEXP [Thm. 3.12]
	$\forall^*\exists^*.\Box$	NEXP [Lem. 3.13]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\Box(\bigcirc^*)$	coRE [Lem. 3.10]	Σ_1^1 [Thm. 3.11]
Temporal Liveness	complete fragment	Σ_1^1 [Thm. 4.2]	Σ_1^1 [Thm. 4.2]
	$\forall\exists^*.$ det-liveness	trivial [Prop. 4.15]	Σ_1^1 [Cor. 4.16]
	$\forall\exists^*.\Diamond(\bigcirc^*)$	NP [Lem. 4.4]	dec. [Thm. 4.6]
	$\forall^*\exists^*.\Diamond\wedge\cdots\wedge\Diamond$	NP [Lem. 4.4]	Σ_1^1 [Thm. 4.12]

First decidability result for formulas that can enforce models with infinitely many traces

Finding Largest Models for ∀∃* HyperLTL

1: **procedure** FINDMODEL($\mathcal{A}$)
2: **if** $\mathcal{L}(\mathcal{A}^{\forall}) = \emptyset$ **then**
3: **return** UNSAT;
4: **if** $\mathcal{L}(\mathcal{A}^{\exists}) \subseteq \mathcal{L}(\mathcal{A}^{\forall})$ **then**
5: **return** SAT, model: $\mathcal{L}(\mathcal{A}^{\forall})$;
6: $\mathcal{A}_{\text{new}} := \mathcal{A} \cap \mathcal{A}^{\forall}_{\pi'}$;
7: FINDMODEL($\mathcal{A}_{\text{new}}$);

Finding Largest Models for $\forall\exists^*$ HyperLTL

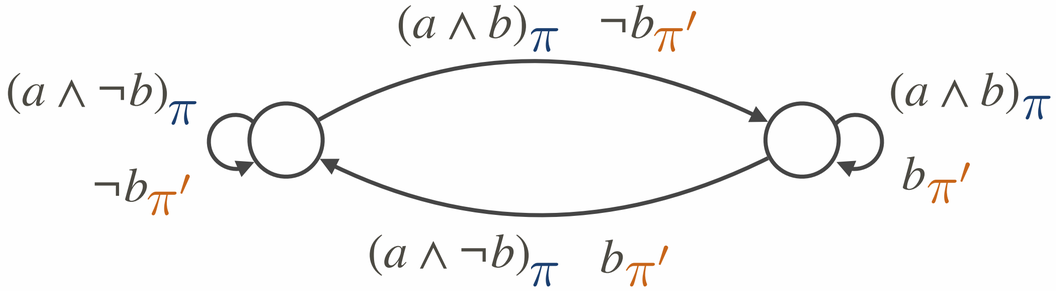
```
1: procedure FINDMODEL(𝒜)
2:     if ℒ(𝒜^∀) = ∅ then
3:         return UNSAT;
4:     if ℒ(𝒜^∃) ⊆ ℒ(𝒜^∀) then
5:         return SAT, model: ℒ(𝒜^∀);
6:     𝒜_new := 𝒜 ∩ 𝒜^∀_π';
7:     FINDMODEL(𝒜_new);
```

$\forall\pi.\exists\pi'.\ \square(a_\pi \wedge (b_\pi \leftrightarrow \bigcirc b_{\pi'}))$

Finding Largest Models for $\forall\exists^*$ HyperLTL

```
1: procedure FINDMODEL(A)
2:     if L(A^∀) = ∅ then
3:         return UNSAT;
4:     if L(A^∃) ⊆ L(A^∀) then
5:         return SAT, model: L(A^∀);
6:     A_new := A ∩ A^∀_π';
7:     FINDMODEL(A_new);
```

$\forall\pi.\exists\pi'.\ \underbrace{\square(a_\pi \wedge (b_\pi \leftrightarrow \bigcirc b_{\pi'}))}$

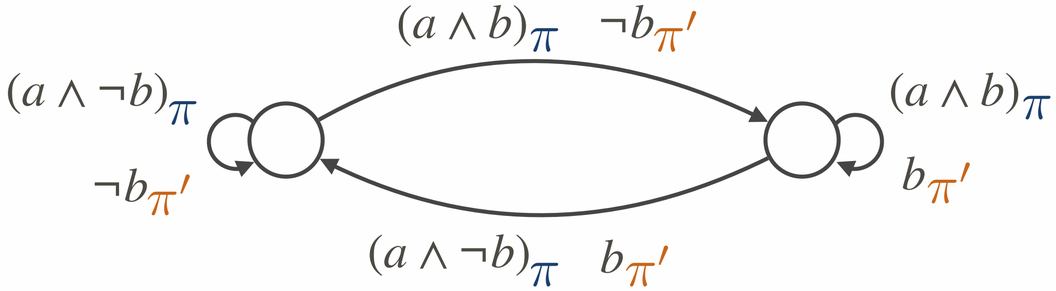


Finding Largest Models for ∀∃* HyperLTL

```
1: procedure FINDMODEL(𝒜)
2:     if ℒ(𝒜^∀) = ∅ then
3:         return UNSAT;
4:     if ℒ(𝒜^∃) ⊆ ℒ(𝒜^∀) then
5:         return SAT, model: ℒ(𝒜^∀);
6:     𝒜_new := 𝒜 ∩ 𝒜^∀_π′;
7:     FINDMODEL(𝒜_new);
```

Remove ∀-traces that produce wrong ∃-traces

$\forall\pi.\exists\pi'.\ \underbrace{\square(a_\pi \wedge (b_\pi \leftrightarrow \bigcirc b_{\pi'}))}$

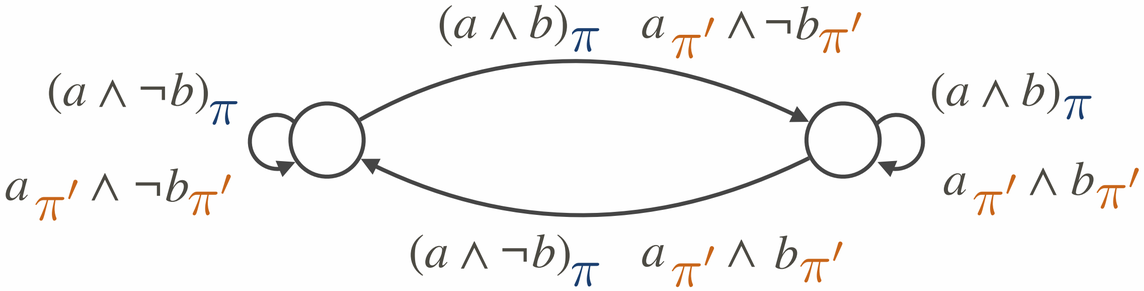


Finding Largest Models for $\forall\exists^*$ HyperLTL

```
1: procedure FINDMODEL(𝒜)
2:     if ℒ(𝒜^∀) = ∅ then
3:         return UNSAT;
4:     if ℒ(𝒜^∃) ⊆ ℒ(𝒜^∀) then
5:         return SAT, model: ℒ(𝒜^∀);
6:     𝒜_new := 𝒜 ∩ 𝒜^∀_π';
7:     FINDMODEL(𝒜_new);
```

Remove ∀-traces that produce wrong ∃-traces

$$\forall\pi.\exists\pi'.\ \underbrace{\square(a_\pi \wedge (b_\pi \leftrightarrow \bigcirc b_{\pi'}))}$$

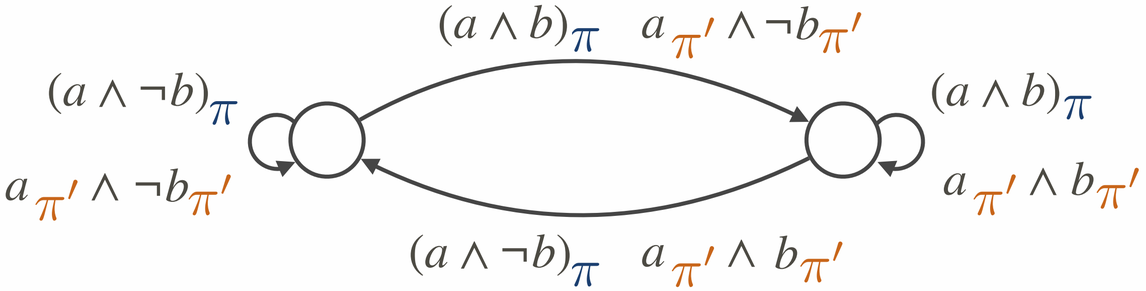


Finding Largest Models for ∀∃* HyperLTL

```
1: procedure FINDMODEL(𝒜)
2:     if ℒ(𝒜^∀) = ∅ then
3:         return UNSAT;
4:     if ℒ(𝒜^∃) ⊆ ℒ(𝒜^∀) then
5:         return SAT, model: ℒ(𝒜^∀);
6:     𝒜_new := 𝒜 ∩ 𝒜^∀_π';
7:     FINDMODEL(𝒜_new);
```

Remove ∀-traces that produce wrong ∃-traces

$\forall \pi . \exists \pi' . \underbrace{\square(a_\pi \wedge (b_\pi \leftrightarrow \bigcirc b_{\pi'}))}$



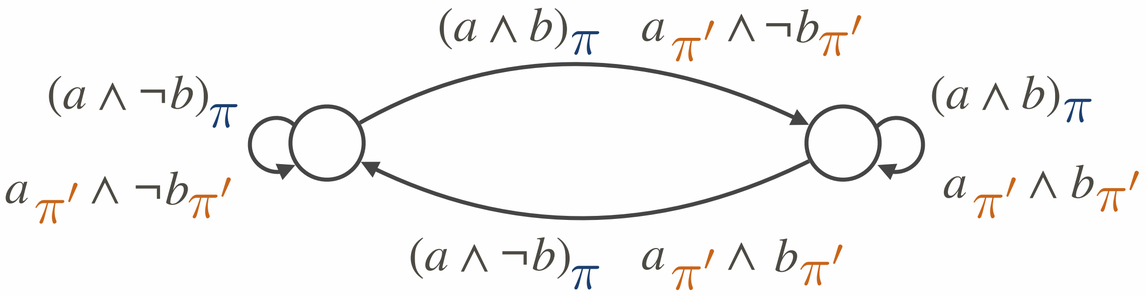
- Finds largest models

Finding Largest Models for ∀∃* HyperLTL

```
1: procedure FINDMODEL(𝒜)
2:     if ℒ(𝒜^∀) = ∅ then
3:         return UNSAT;
4:     if ℒ(𝒜^∃) ⊆ ℒ(𝒜^∀) then
5:         return SAT, model: ℒ(𝒜^∀);
6:     𝒜_new := 𝒜 ∩ 𝒜^∀_π';
7:     FINDMODEL(𝒜_new);
```

Remove ∀-traces that produce wrong ∃-traces

$\forall\pi.\exists\pi'.\ \underbrace{\square(a_\pi \wedge (b_\pi \leftrightarrow \bigcirc b_{\pi'}))}$



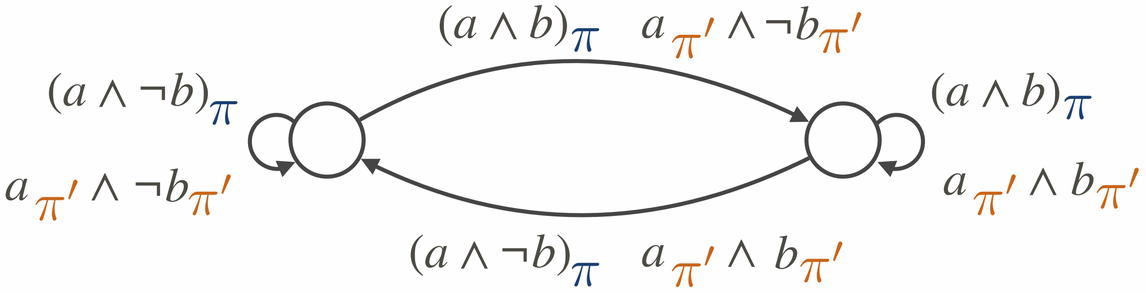
- Finds largest models
- Sound but necessarily not complete

Finding Largest Models for ∀∃* HyperLTL

```
1: procedure FINDMODEL(𝒜)
2:     if ℒ(𝒜^∀) = ∅ then
3:         return UNSAT;
4:     if ℒ(𝒜^∃) ⊆ ℒ(𝒜^∀) then
5:         return SAT, model: ℒ(𝒜^∀);
6:     𝒜_new := 𝒜 ∩ 𝒜^∀_π';
7:     FINDMODEL(𝒜_new);
```

Remove ∀-traces that produce wrong ∃-traces

$\forall\pi.\exists\pi'.\ \underbrace{\square(a_\pi \wedge (b_\pi \leftrightarrow \bigcirc b_{\pi'}))}$



- Finds largest models
- Sound but necessarily not complete
- Evaluation: finds models that MGHyper[1] does not find, can show unsatisfiability

[1] Finkbeiner, Hahn, Hans. MGHyper: *Checking Satisfiability of HyperLTL formulas beyond the ∃*∀* Fragment*. ATVA 2018.

Conclusion

Conclusion

- Syntactic fragments of $\forall^*\exists^*$ HyperLTL do not make satisfiability easier

Conclusion

- Syntactic fragments of $\forall^*\exists^*$ HyperLTL do not make satisfiability easier
- 2 new perspectives: temporal safety/liveness + split in functional property and hyperproperty

		no LTL spec.	with LTL spec.
Temporal Safety	complete fragment	coRE [Thm. 3.7]	Σ^1_1 [Thm. 3.11]
	$\forall^*\exists^*.\bigcirc^*$	NEXP [Thm. 3.12]	NEXP [Thm. 3.12]
	$\forall^*\exists^*.\square$	NEXP [Lem. 3.13]	Σ^1_1 [Thm. 3.11]
	$\forall^*\exists^*.\square(\bigcirc^*)$	coRE [Lem. 3.10]	Σ^1_1 [Thm. 3.11]
Temporal Liveness	complete fragment	Σ^1_1 [Thm. 4.2]	Σ^1_1 [Thm. 4.2]
	$\forall\exists^*$. det-liveness	trivial [Prop. 4.15]	Σ^1_1 [Cor. 4.16]
	$\forall\exists^*.\Diamond(\bigcirc^*)$	NP [Lem. 4.4]	dec. [Thm. 4.6]
	$\forall^*\exists^*.\Diamond\wedge\cdots\wedge\Diamond$	NP [Lem. 4.4]	Σ^1_1 [Thm. 4.12]

Conclusion

- Syntactic fragments of $\forall^*\exists^*$ HyperLTL do not make satisfiability easier
- 2 new perspectives: temporal safety/liveness + split in functional property and hyperproperty

		no LTL spec.	with LTL spec.
Temporal Safety	complete fragment	coRE [Thm. 3.7]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\bigcirc^*$	NEXP [Thm. 3.12]	NEXP [Thm. 3.12]
	$\forall^*\exists^*.\square$	NEXP [Lem. 3.13]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\square(\bigcirc^*)$	coRE [Lem. 3.10]	Σ_1^1 [Thm. 3.11]
Temporal Liveness	complete fragment	Σ_1^1 [Thm. 4.2]	Σ_1^1 [Thm. 4.2]
	$\forall\exists^*$. det-liveness	trivial [Prop. 4.15]	Σ_1^1 [Cor. 4.16]
	$\forall\exists^*.\Diamond(\bigcirc^*)$	NP [Lem. 4.4]	dec. [Thm. 4.6]
	$\forall^*\exists^*.\Diamond\wedge\cdots\wedge\Diamond$	NP [Lem. 4.4]	Σ_1^1 [Thm. 4.12]

- Sound algorithm for largest models for $\forall\exists^*$ HyperLTL

Conclusion

- Syntactic fragments of $\forall^*\exists^*$ HyperLTL do not make satisfiability easier
- 2 new perspectives: temporal safety/liveness + split in functional property and hyperproperty

		no LTL spec.	with LTL spec.
Temporal Safety	complete fragment	coRE [Thm. 3.7]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\bigcirc^*$	NEXP [Thm. 3.12]	NEXP [Thm. 3.12]
	$\forall^*\exists^*.\square$	NEXP [Lem. 3.13]	Σ_1^1 [Thm. 3.11]
	$\forall^*\exists^*.\square(\bigcirc^*)$	coRE [Lem. 3.10]	Σ_1^1 [Thm. 3.11]
Temporal Liveness	complete fragment	Σ_1^1 [Thm. 4.2]	Σ_1^1 [Thm. 4.2]
	$\forall\exists^*$. det-liveness	trivial [Prop. 4.15]	Σ_1^1 [Cor. 4.16]
	$\forall\exists^*.\Diamond(\bigcirc^*)$	NP [Lem. 4.4]	dec. [Thm. 4.6]
	$\forall^*\exists^*.\Diamond\wedge\cdots\wedge\Diamond$	NP [Lem. 4.4]	Σ_1^1 [Thm. 4.12]

New $\forall^*\exists^*$ decidability results beyond purely syntactic restrictions

Fixpoint algorithm for SAT and UNSAT

- Sound algorithm for largest models for $\forall\exists^*$ HyperLTL