Prophecies all the Way: Game-based Model-Checking for HyperQPTL beyond ∀*∃*

Joint work with Sarah Winter (IRIF, Paris)

Martin Zimmermann

Aalborg University

CONCUR 2025, Aarhus, Denmark

Prophecies all the Way: Game-based Model-Checking for HyperLTL beyond $\forall^*\exists^*$

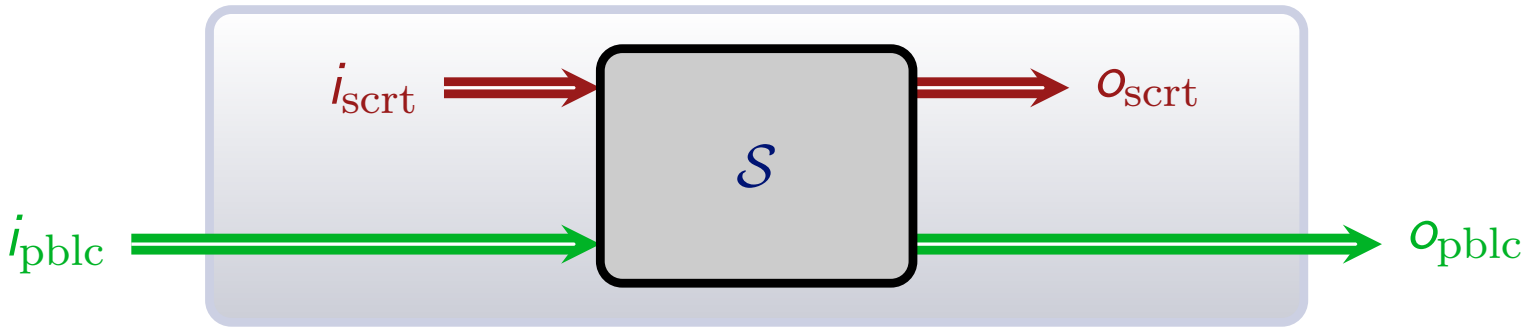
Joint work with Sarah Winter (IRIF, Paris)

Martin Zimmermann

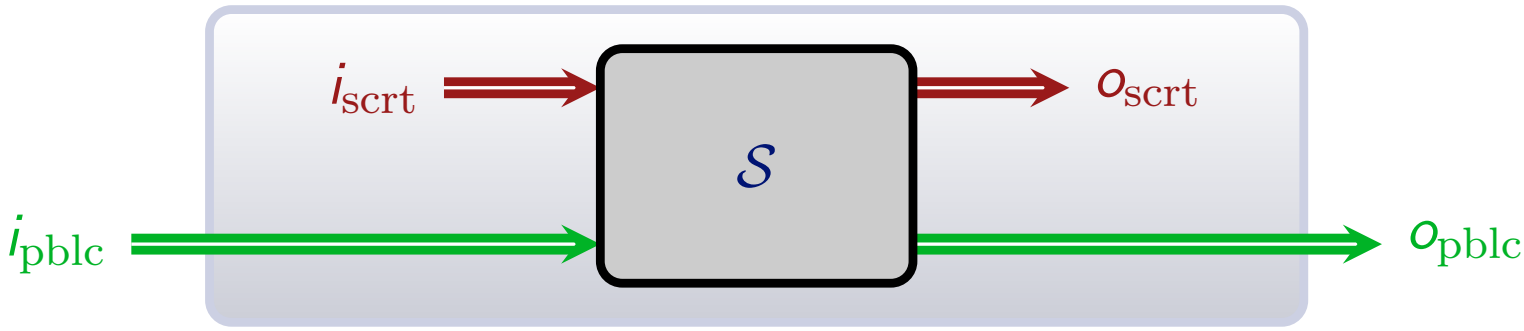
Aalborg University

CONCUR 2025, Aarhus, Denmark

Reactive Systems

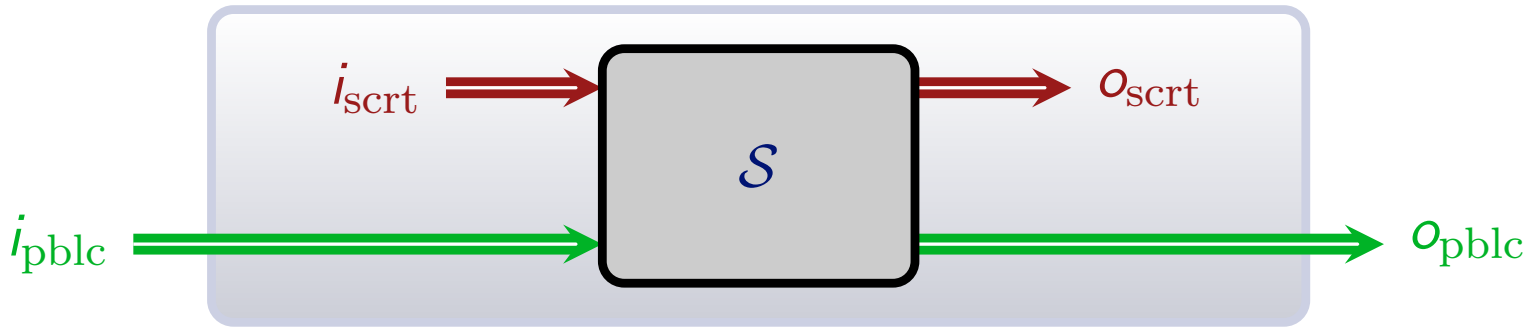


Reactive Systems



Trace-based view on $\mathcal{S}$: observe execution traces, i.e., infinite sequences over 2^{AP} for some set AP of atomic propositions.

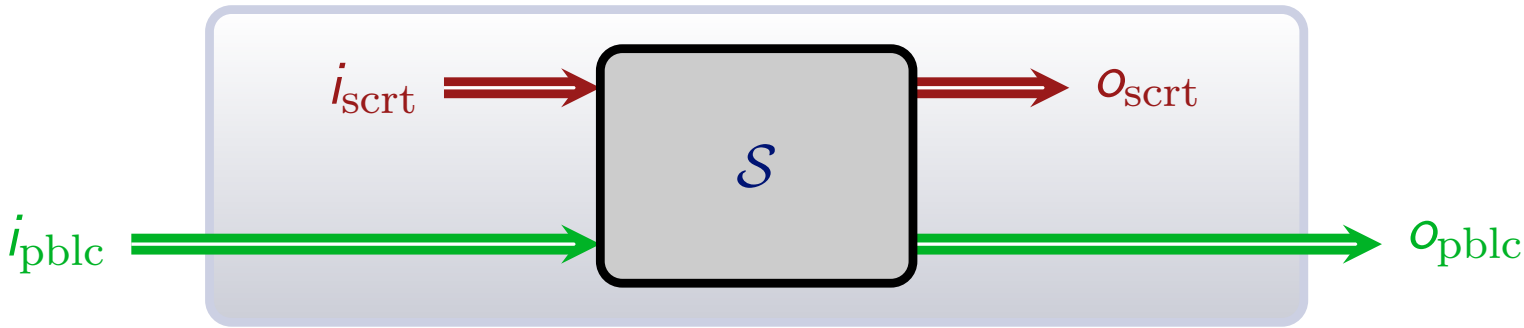
Reactive Systems



Trace-based view on $\mathcal{S}$: observe execution traces, i.e., infinite sequences over 2^{AP} for some set AP of atomic propositions.

$\{\texttt{init}, \texttt{i}_{\text{pblc}}\}$

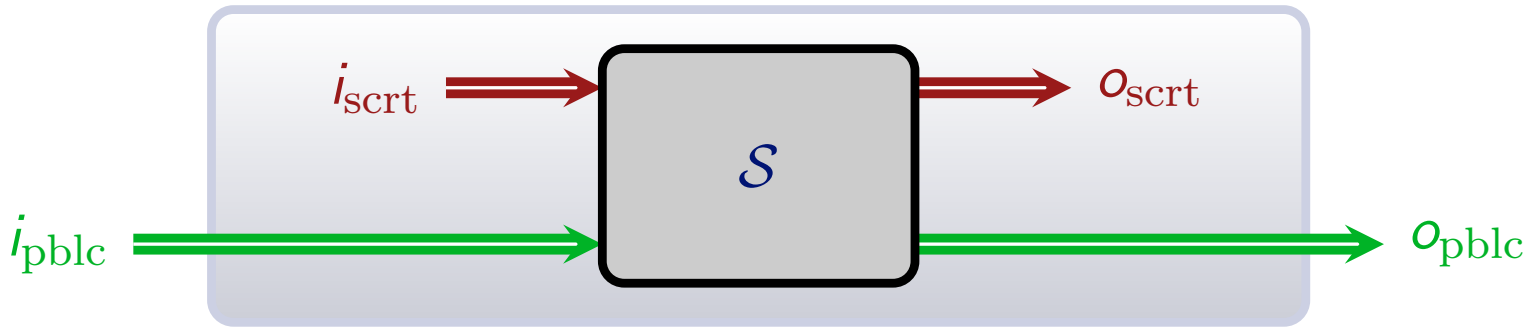
Reactive Systems



Trace-based view on $\mathcal{S}$: observe execution traces, i.e., infinite sequences over 2^{AP} for some set AP of atomic propositions.

$\{\texttt{init}, \texttt{i}_{\text{pblc}}\}$ $\{\texttt{i}_{\text{scrt}}\}$

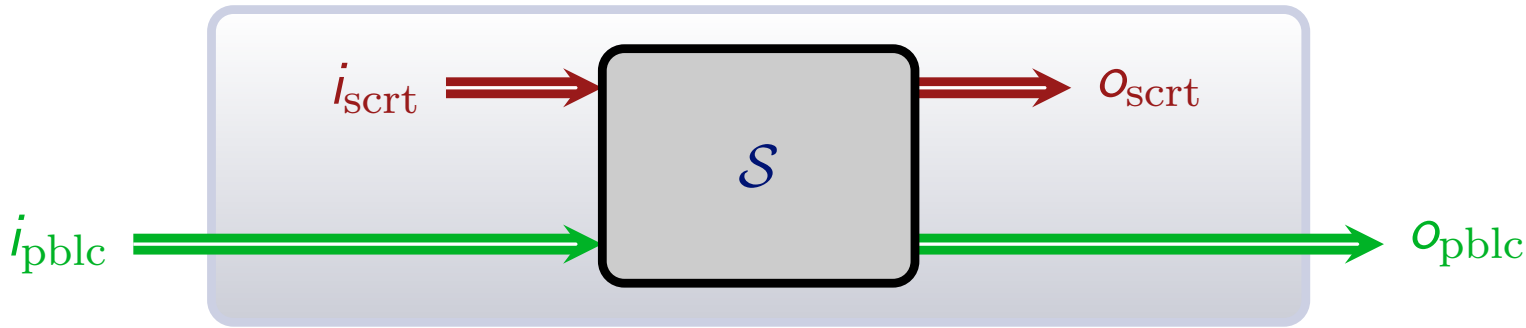
Reactive Systems



Trace-based view on $\mathcal{S}$: observe execution traces, i.e., infinite sequences over 2^{AP} for some set AP of atomic propositions.

$\{\texttt{init}, \texttt{i}_{\text{pblc}}\}$ $\{\texttt{i}_{\text{scrt}}\}$ $\{\texttt{i}_{\text{pblc}}\}$

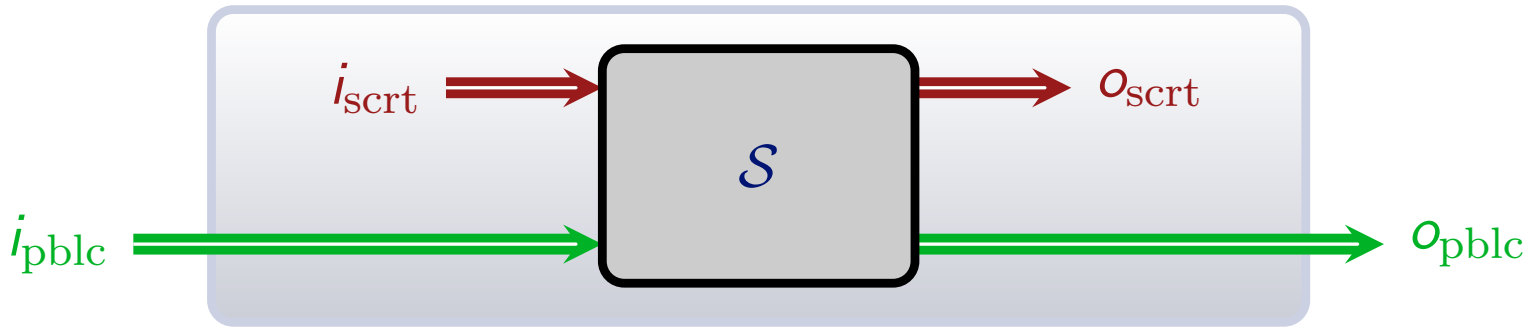
Reactive Systems



Trace-based view on $\mathcal{S}$: observe execution traces, i.e., infinite sequences over 2^{AP} for some set AP of atomic propositions.

$\{\texttt{init}, \texttt{i}_{\text{pblc}}\}$ $\{\texttt{i}_{\text{scrt}}\}$ $\{\texttt{i}_{\text{pblc}}\}$ $\{\texttt{i}_{\text{scrt}}, \texttt{o}_{\text{pblc}}, \texttt{term}\}$

Reactive Systems



Trace-based view on $\mathcal{S}$: observe execution traces, i.e., infinite sequences over 2^{AP} for some set AP of atomic propositions.

$\{\texttt{init}, \texttt{i}_{\text{pblc}}\}$ $\{\texttt{i}_{\text{scrt}}\}$ $\{\texttt{i}_{\text{pblc}}\}$ $\{\texttt{i}_{\text{scrt}}, \texttt{o}_{\text{pblc}}, \texttt{term}\}$ $\emptyset \cdots$

Trace Properties vs. Hyperproperties

- $\mathcal{S}$ terminates

Trace Properties vs. Hyperproperties

- $\mathcal{S}$ terminates
- $\mathcal{S}$ terminates within a uniform time bound

Trace Properties vs. Hyperproperties

- $\mathcal{S}$ terminates
- $\mathcal{S}$ terminates within a uniform time bound
- Noninterference: for all traces t, t' of $\mathcal{S}$, if t and t' coincide on their projection to their public inputs, then they also coincide on their projection to the public outputs.

Trace Properties vs. Hyperproperties

- $\mathcal{S}$ terminates
- $\mathcal{S}$ terminates within a uniform time bound
- Noninterference: for all traces t, t' of $\mathcal{S}$, if t and t' coincide on their projection to their public inputs, then they also coincide on their projection to the public outputs.
- Noninterference for nondeterministic systems: for all traces t, t' of $\mathcal{S}$ there exists a trace t'' of $\mathcal{S}$ such that t'' and t coincide on their projection to public inputs and outputs and t'' and t' coincide on their projection to secret inputs.

Trace Properties vs. Hyperproperties

- $\mathcal{S}$ terminates
- $\mathcal{S}$ terminates within a uniform time bound
- Noninterference: for all traces t, t' of $\mathcal{S}$, if t and t' coincide on their projection to their public inputs, then they also coincide on their projection to the public outputs.
- Noninterference for nondeterministic systems: for all traces t, t' of $\mathcal{S}$ there exists a trace t'' of $\mathcal{S}$ such that t'' and t coincide on their projection to public inputs and outputs and t'' and t' coincide on their projection to secret inputs.

Remark

- The first property can be checked by inspecting each trace in isolation (a trace property), but
- the other three properties can only be checked by reasoning about multiple traces simultaneously (hyperproperties).

LTL in One Slide

Syntax

$$\varphi ::= a \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \varphi\,\mathsf{U}\,\varphi \qquad \text{where } a \in \mathrm{AP}$$

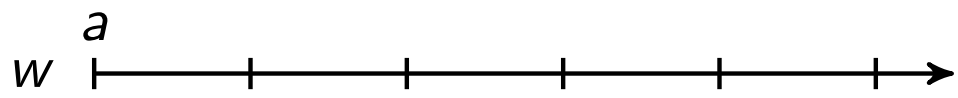
LTL in One Slide

Syntax

$$\varphi ::= a \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \varphi\,\mathsf{U}\,\varphi \qquad \text{where } a \in \mathrm{AP}$$

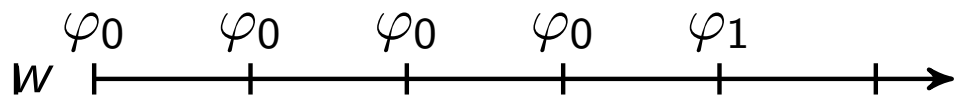
Semantics

- $w \models a$:



- $w \models \mathsf{X}\,\varphi$:

- $w \models \varphi_0\,\mathsf{U}\,\varphi_1$:



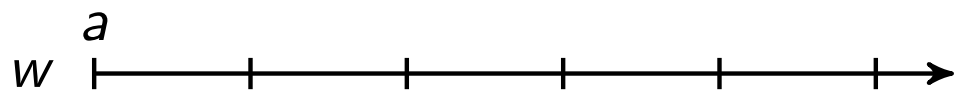
LTL in One Slide

Syntax

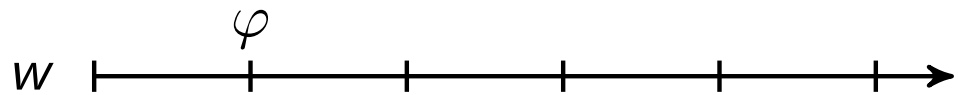
$$\varphi ::= a \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \varphi\,\mathsf{U}\,\varphi \qquad \text{where } a \in \mathrm{AP}$$

Semantics

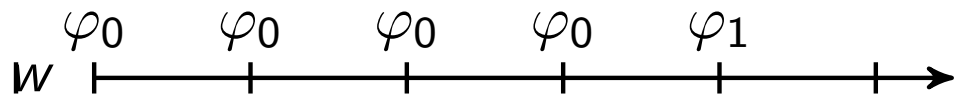
- $w \models a$:



- $w \models \mathsf{X}\,\varphi$:



- $w \models \varphi_0\,\mathsf{U}\,\varphi_1$:



Syntactic Sugar

- $\mathsf{F}\,\psi = \mathsf{tt}\,\mathsf{U}\,\psi$
- $\mathsf{G}\,\psi = \neg\,\mathsf{F}\,\neg\psi$

HyperLTL

HyperLTL = LTL + trace quantification

$$\varphi ::= \exists \pi.\ \varphi \mid \forall \pi.\ \varphi \mid \psi$$
$$\psi ::= a_\pi \mid \neg \psi \mid \psi \vee \psi \mid \mathsf{X}\, \psi \mid \psi\, \mathsf{U}\, \psi$$

where $a \in \mathrm{AP}$ and $\pi \in \mathcal{V}$ (trace variables).

HyperLTL

HyperLTL = LTL + trace quantification

$$\varphi ::= \exists \pi.\ \varphi \mid \forall \pi.\ \varphi \mid \psi$$
$$\psi ::= a_\pi \mid \neg\psi \mid \psi \vee \psi \mid \mathsf{X}\,\psi \mid \psi\,\mathsf{U}\,\psi$$

where $a \in \mathrm{AP}$ and $\pi \in \mathcal{V}$ (trace variables).

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

HyperLTL

HyperLTL = LTL + trace quantification

$$\varphi ::= \exists \pi.\ \varphi \mid \forall \pi.\ \varphi \mid \psi$$
$$\psi ::= a_\pi \mid \neg\psi \mid \psi \vee \psi \mid \mathsf{X}\,\psi \mid \psi\,\mathsf{U}\,\psi$$

where $a \in \mathrm{AP}$ and $\pi \in \mathcal{V}$ (trace variables).

- Prenex normal form, but
- closed under boolean combinations.
- Time passes synchronously on the quantified traces.

Examples

- Noninterference:

$$\forall\pi\forall\pi'.\ \mathsf{G}((i_{\mathsf{pblc}})_\pi \leftrightarrow (i_{\mathsf{pblc}})_{\pi'}) \rightarrow \mathsf{G}((o_{\mathsf{pblc}})_\pi \leftrightarrow (o_{\mathsf{pblc}})_{\pi'})$$

Examples

- Noninterference:

$$\forall\pi\forall\pi'.\ \mathsf{G}((i_{\mathsf{pblc}})_\pi \leftrightarrow (i_{\mathsf{pblc}})_{\pi'}) \rightarrow \mathsf{G}((o_{\mathsf{pblc}})_\pi \leftrightarrow (o_{\mathsf{pblc}})_{\pi'})$$

- Noninterference for nondeterministic systems:

$$\begin{aligned}\forall\pi\forall\pi'\exists\pi''.\ &\mathsf{G}((i_{\mathsf{pblc}})_\pi \leftrightarrow (i_{\mathsf{pblc}})_{\pi''}) \wedge \\ &\mathsf{G}((o_{\mathsf{pblc}})_\pi \leftrightarrow (o_{\mathsf{pblc}})_{\pi''}) \wedge \\ &\mathsf{G}((i_{\mathsf{scrt}})_{\pi'} \leftrightarrow (i_{\mathsf{scrt}})_{\pi''})\end{aligned}$$

Examples

- Noninterference:

$$\forall\pi\forall\pi'.\ \mathsf{G}((i_{\mathsf{pblc}})_\pi \leftrightarrow (i_{\mathsf{pblc}})_{\pi'}) \rightarrow \mathsf{G}((o_{\mathsf{pblc}})_\pi \leftrightarrow (o_{\mathsf{pblc}})_{\pi'})$$

- Noninterference for nondeterministic systems:

$$\begin{aligned}\forall\pi\forall\pi'\exists\pi''.\ &\mathsf{G}((i_{\mathsf{pblc}})_\pi \leftrightarrow (i_{\mathsf{pblc}})_{\pi''}) \wedge \\ &\mathsf{G}((o_{\mathsf{pblc}})_\pi \leftrightarrow (o_{\mathsf{pblc}})_{\pi''}) \wedge \\ &\mathsf{G}((i_{\mathsf{scrt}})_{\pi'} \leftrightarrow (i_{\mathsf{scrt}})_{\pi''})\end{aligned}$$

- $\mathcal{S}$ terminates within a uniform time bound. Not expressible in HyperLTL.

Model-Checking

The HyperLTL model-checking problem:

Given a finite transition system $\mathcal{S}$ and φ, does $\mathrm{Traces}(\mathcal{S}) \models \varphi$?

Model-Checking

The HyperLTL model-checking problem:

Given a finite transition system $\mathcal{S}$ and φ, does $\mathrm{Traces}(\mathcal{S}) \models \varphi$?

Recall: The LTL model-checking problem is PSPACE-complete.

Model-Checking

The HyperLTL model-checking problem:

Given a finite transition system $\mathcal{S}$ and φ, does $\mathrm{Traces}(\mathcal{S}) \models \varphi$?

Recall: The LTL model-checking problem is PSPACE-complete.

Theorem (Clarkson et al. '14, Rabe '16)

The HyperLTL model-checking problem is TOWER-complete (in the number of quantifier alternations).

Model-Checking

The HyperLTL model-checking problem:

Given a finite transition system $\mathcal{S}$ and φ, does $\mathrm{Traces}(\mathcal{S}) \models \varphi$?

Recall: The LTL model-checking problem is PSPACE-complete.

Theorem (Clarkson et al. '14, Rabe '16)

The HyperLTL model-checking problem is TOWER-complete (in the number of quantifier alternations).

Bottleneck of the "classical" algorithm: complementation of Büchi automata.

Model-Checking via Automata

Proof:

- Given φ, we replace every $\forall\pi.$ by $\neg\exists\pi.\neg.$

Model-Checking via Automata

Proof:

- Given φ, we replace every $\forall\pi.$ by $\neg\exists\pi.\neg$.
- We construct, by induction over the quantifier prefix, non-determinstic Büchi automata accepting exactly the variable assignments satisfying the subformulas of φ.
- Then, we obtain an automaton $\mathcal{A}$ with $L(\mathcal{A}) \neq \emptyset$ iff $\mathrm{Traces}(\mathcal{S}) \models \varphi$.

Model-Checking via Automata

Proof:

- Given φ, we replace every $\forall\pi.$ by $\neg\exists\pi.\neg$.
- We construct, by induction over the quantifier prefix, non-determinstic Büchi automata accepting exactly the variable assignments satisfying the subformulas of φ.
- Then, we obtain an automaton $\mathcal{A}$ with $L(\mathcal{A}) \neq \emptyset$ iff $\mathrm{Traces}(\mathcal{S}) \models \varphi$.
  - Induction start: build automaton for the LTL formula obtained from $\neg\psi$ by replacing a_{π_j} by a_j.

Model-Checking via Automata

Proof:

- Given φ, we replace every $\forall\pi.$ by $\neg\exists\pi.\neg$.
- We construct, by induction over the quantifier prefix, non-determinstic Büchi automata accepting exactly the variable assignments satisfying the subformulas of φ.
- Then, we obtain an automaton $\mathcal{A}$ with $L(\mathcal{A}) \neq \emptyset$ iff $\mathrm{Traces}(\mathcal{S}) \models \varphi$.
  - Induction start: build automaton for the LTL formula obtained from $\neg\psi$ by replacing a_{π_j} by a_j.
  - For $\exists\pi_j\theta$ restrict automaton for θ in dimension j to traces of $\mathcal{S}$ (involves product with $\mathcal{S}$).

Model-Checking via Automata

Proof:

- Given φ, we replace every $\forall\pi.$ by $\neg\exists\pi.\neg$.
- We construct, by induction over the quantifier prefix, non-determinstic Büchi automata accepting exactly the variable assignments satisfying the subformulas of φ.
- Then, we obtain an automaton $\mathcal{A}$ with $L(\mathcal{A}) \neq \emptyset$ iff $\mathrm{Traces}(\mathcal{S}) \models \varphi$.
  - Induction start: build automaton for the LTL formula obtained from $\neg\psi$ by replacing a_{π_j} by a_j.
  - For $\exists\pi_j\theta$ restrict automaton for θ in dimension j to traces of $\mathcal{S}$ (involves product with $\mathcal{S}$).
  - For $\neg\theta$ complement automaton for θ.

Games to the Rescue

Consider a formula of the form $\forall\pi_0\exists\pi_1.\ \psi$ with quantifier-free ψ.

- Its semantics can be captured by a two-player perfect-information zero-sum game:

Games to the Rescue

Consider a formula of the form $\forall\pi_0\exists\pi_1.\ \psi$ with quantifier-free ψ.

- Its semantics can be captured by a two-player perfect-information zero-sum game:
  - Challenger picks a trace t_0 for π_0.

Games to the Rescue

Consider a formula of the form $\forall\pi_0\exists\pi_1.\ \psi$ with quantifier-free ψ.

- Its semantics can be captured by a two-player perfect-information zero-sum game:
  - Challenger picks a trace t_0 for π_0.
  - Then, Prover picks a trace t_1 for π_1.

Games to the Rescue

Consider a formula of the form $\forall\pi_0\exists\pi_1.\ \psi$ with quantifier-free ψ.

- Its semantics can be captured by a two-player perfect-information zero-sum game:
  - Challenger picks a trace t_0 for π_0.
  - Then, Prover picks a trace t_1 for π_1.
  - Prover wins if the assignment

$$\{\pi_0 \mapsto t_0, \pi_1 \mapsto t_1\}$$

satisfies ψ.

Games to the Rescue

Consider a formula of the form $\forall\pi_0\exists\pi_1.\ \psi$ with quantifier-free ψ.

- Its semantics can be captured by a two-player perfect-information zero-sum game:
  - Challenger picks a trace t_0 for π_0.
  - Then, Prover picks a trace t_1 for π_1.
  - Prover wins if the assignment

$$\{\pi_0 \mapsto t_0, \pi_1 \mapsto t_1\}$$

    satisfies ψ.

Remark

- The game is sound and complete,
- but it is in general not algorithmically solvable.

Sequential Games..

.. are algorithmically more appealing..

π_0

π_1

Sequential Games..

.. are algorithmically more appealing..

π_0 t_0^0

π_1

Sequential Games..

.. are algorithmically more appealing..

π_0 t_0^0

π_1 t_1^0

Sequential Games..

.. are algorithmically more appealing..

π_0 t_0^0 t_0^1

π_1 t_1^0

Sequential Games..

.. are algorithmically more appealing..

π_0 t_0^0 t_0^1

π_1 t_1^0 t_1^1

Sequential Games..

.. are algorithmically more appealing..

π_0 t_0^0 t_0^1 t_0^2

π_1 t_1^0 t_1^1

Sequential Games..

.. are algorithmically more appealing..

π_0 t_0^0 t_0^1 t_0^2

π_1 t_1^0 t_1^1 t_1^2

Sequential Games..

.. are algorithmically more appealing..

π_0 t_0^0 t_0^1 t_0^2 t_0^3

π_1 t_1^0 t_1^1 t_1^2

Sequential Games..

.. are algorithmically more appealing..

π_0 t_0^0 t_0^1 t_0^2 t_0^3

π_1 t_1^0 t_1^1 t_1^2 t_1^3

Sequential Games..

.. are algorithmically more appealing..

π_0 t_0^0 t_0^1 t_0^2 t_0^3 t_0^4

π_1 t_1^0 t_1^1 t_1^2 t_1^3

Sequential Games..

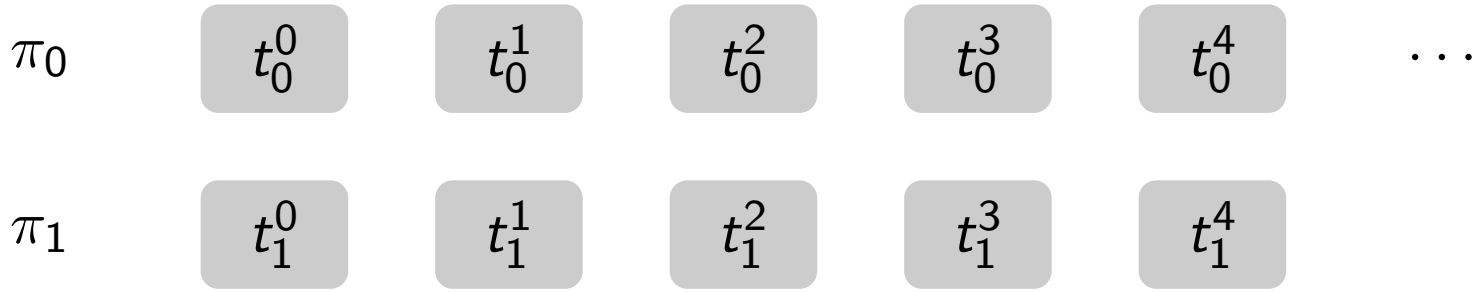
.. are algorithmically more appealing..

π_0 t_0^0 t_0^1 t_0^2 t_0^3 t_0^4

π_1 t_1^0 t_1^1 t_1^2 t_1^3 t_1^4

Sequential Games..

.. are algorithmically more appealing..



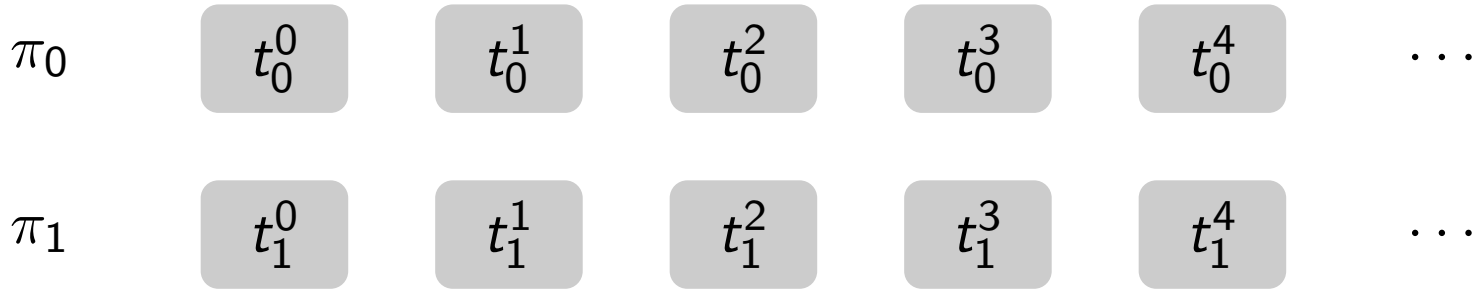
Sequential Games..

.. are algorithmically more appealing..

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4	$\cdots$
π_1	t_1^0	t_1^1	t_1^2	t_1^3	t_1^4	$\cdots$

Sequential Games..

.. are algorithmically more appealing..



.. and sound, but not complete.

Sequential Games..

.. are algorithmically more appealing..

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4	$\cdots$
π_1	t_1^0	t_1^1	t_1^2	t_1^3	t_1^4	$\cdots$

.. and sound, but not complete.

Example

Consider the formula

$$\varphi = \forall\pi\exists\pi'.\ (\mathsf{F}\, a_\pi) \leftrightarrow a_{\pi'}.$$

$(2^{\{a\}})^\omega$ satisfies φ, but Prover loses the sequential game.

Prophecies

- Intuitively, Prover is at a disadvantage, because she does not get access to the full trace t when making her first move.

Prophecies

- Intuitively, Prover is at a disadvantage, because she does not get access to the full trace t when making her first move.
- But she does not need to, limited information is sufficient: In the example, knowing whether t contains an a or not.

Prophecies

- Intuitively, Prover is at a disadvantage, because she does not get access to the full trace t when making her first move.
- But she does not need to, limited information is sufficient: In the example, knowing whether t contains an a or not.
- A prophecy is an ω-language, Challenger has to specify in each round whether the suffix of the trace he picks is in the prophecy or not. If he cheats, he loses.

Prophecies

- Intuitively, Prover is at a disadvantage, because she does not get access to the full trace t when making her first move.
- But she does not need to, limited information is sufficient: In the example, knowing whether t contains an a or not.
- A prophecy is an ω-language, Challenger has to specify in each round whether the suffix of the trace he picks is in the prophecy or not. If he cheats, he loses.

Theorem [Beutner and Finkbeiner '22]
For every transition system $\mathcal{S}$ and every $\forall^*\exists^*$ HyperLTL formula φ there is a finite set of prophecies such that $\mathcal{S} \models \varphi$ if and only if Prover wins the induced game with prophecies (which is a finite parity game).

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0

π_1

π_2

π_3

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0 t_0^0

π_1

π_2

π_3

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0 t_0^0

π_1 t_1^0

π_2

π_3

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0
π_1	t_1^0
π_2	t_2^0
π_3	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0
π_1	t_1^0
π_2	t_2^0
π_3	t_3^0

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1
π_1	t_1^0	
π_2	t_2^0	
π_3	t_3^0	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1
π_1	t_1^0	t_1^1
π_2	t_2^0	
π_3	t_3^0	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1
π_1	t_1^0	t_1^1
π_2	t_2^0	t_2^1
π_3	t_3^0	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1
π_1	t_1^0	t_1^1
π_2	t_2^0	t_2^1
π_3	t_3^0	t_3^1

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2
π_1	t_1^0	t_1^1	
π_2	t_2^0	t_2^1	
π_3	t_3^0	t_3^1	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2
π_1	t_1^0	t_1^1	t_1^2
π_2	t_2^0	t_2^1	
π_3	t_3^0	t_3^1	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2
π_1	t_1^0	t_1^1	t_1^2
π_2	t_2^0	t_2^1	t_2^2
π_3	t_3^0	t_3^1	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2
π_1	t_1^0	t_1^1	t_1^2
π_2	t_2^0	t_2^1	t_2^2
π_3	t_3^0	t_3^1	t_3^2

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3
π_1	t_1^0	t_1^1	t_1^2	
π_2	t_2^0	t_2^1	t_2^2	
π_3	t_3^0	t_3^1	t_3^2	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3
π_1	t_1^0	t_1^1	t_1^2	t_1^3
π_2	t_2^0	t_2^1	t_2^2	
π_3	t_3^0	t_3^1	t_3^2	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3
π_1	t_1^0	t_1^1	t_1^2	t_1^3
π_2	t_2^0	t_2^1	t_2^2	t_2^3
π_3	t_3^0	t_3^1	t_3^2	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3
π_1	t_1^0	t_1^1	t_1^2	t_1^3
π_2	t_2^0	t_2^1	t_2^2	t_2^3
π_3	t_3^0	t_3^1	t_3^2	t_3^3

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4
π_1	t_1^0	t_1^1	t_1^2	t_1^3	
π_2	t_2^0	t_2^1	t_2^2	t_2^3	
π_3	t_3^0	t_3^1	t_3^2	t_3^3	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4
π_1	t_1^0	t_1^1	t_1^2	t_1^3	t_1^4
π_2	t_2^0	t_2^1	t_2^2	t_2^3	
π_3	t_3^0	t_3^1	t_3^2	t_3^3	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4
π_1	t_1^0	t_1^1	t_1^2	t_1^3	t_1^4
π_2	t_2^0	t_2^1	t_2^2	t_2^3	t_2^4
π_3	t_3^0	t_3^1	t_3^2	t_3^3	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4
π_1	t_1^0	t_1^1	t_1^2	t_1^3	t_1^4
π_2	t_2^0	t_2^1	t_2^2	t_2^3	t_2^4
π_3	t_3^0	t_3^1	t_3^2	t_3^3	t_3^4

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4	$\cdots$
π_1	t_1^0	t_1^1	t_1^2	t_1^3	t_1^4	
π_2	t_2^0	t_2^1	t_2^2	t_2^3	t_2^4	
π_3	t_3^0	t_3^1	t_3^2	t_3^3	t_3^4	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4	$\cdots$
π_1	t_1^0	t_1^1	t_1^2	t_1^3	t_1^4	$\cdots$
π_2	t_2^0	t_2^1	t_2^2	t_2^3	t_2^4	
π_3	t_3^0	t_3^1	t_3^2	t_3^3	t_3^4	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4	$\cdots$
π_1	t_1^0	t_1^1	t_1^2	t_1^3	t_1^4	$\cdots$
π_2	t_2^0	t_2^1	t_2^2	t_2^3	t_2^4	$\cdots$
π_3	t_3^0	t_3^1	t_3^2	t_3^3	t_3^4	

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4	$\cdots$
π_1	t_1^0	t_1^1	t_1^2	t_1^3	t_1^4	$\cdots$
π_2	t_2^0	t_2^1	t_2^2	t_2^3	t_2^4	$\cdots$
π_3	t_3^0	t_3^1	t_3^2	t_3^3	t_3^4	$\cdots$

What About General Quantifier Prefixes?

Consider a formula of the form $\forall\pi_0\exists\pi_1\forall\pi_2\exists\pi_3.\ \psi$ with quantifier-free ψ.

π_0	t_0^0	t_0^1	t_0^2	t_0^3	t_0^4	$\cdots$
π_1	t_1^0	t_1^1	t_1^2	t_1^3	t_1^4	$\cdots$
π_2	t_2^0	t_2^1	t_2^2	t_2^3	t_2^4	$\cdots$
π_3	t_3^0	t_3^1	t_3^2	t_3^3	t_3^4	$\cdots$

We need imperfect information to ensure that π_1 does not depend on π_2 and π_3.

Our Main Result

Theorem

For every transition system $\mathcal{S}$ and every HyperLTL formula φ there is a finite set of prophecies such that $\mathcal{S} \models \varphi$ if and only if Prover wins the induced game with prophecies (which is a finite imperfect-information parity game).

Our Main Result

Theorem

For every transition system $\mathcal{S}$ and every HyperLTL formula φ there is a finite set of prophecies such that $\mathcal{S} \models \varphi$ if and only if Prover wins the induced game with prophecies (which is a finite imperfect-information parity game).

- A set of prophecies for each existentially quantified variable with consistency requirements between prophecies.
- Careful setup so that prophecies do not "leak" information.

Our Main Result

Theorem

For every transition system $\mathcal{S}$ and every HyperLTL formula φ there is a finite set of prophecies such that $\mathcal{S} \models \varphi$ if and only if Prover wins the induced game with prophecies (which is a finite imperfect-information parity game).

- A set of prophecies for each existentially quantified variable with consistency requirements between prophecies.
- Careful setup so that prophecies do not "leak" information.
- As many players as variable alternations.

Our Main Result

Theorem

For every transition system $\mathcal{S}$ and every HyperLTL formula φ there is a finite set of prophecies such that $\mathcal{S} \models \varphi$ if and only if Prover wins the induced game with prophecies (which is a finite imperfect-information parity game).

- A set of prophecies for each existentially quantified variable with consistency requirements between prophecies.
- Careful setup so that prophecies do not "leak" information.
- As many players as variable alternations.
- The tower bounding the runtime of our algorithm is "taller" than the lower bound for HyperLTL model-checking.

Our Main Result

Theorem

For every transition system $\mathcal{S}$ and every HyperLTL formula φ there is a finite set of prophecies such that $\mathcal{S} \models \varphi$ if and only if Prover wins the induced game with prophecies (which is a finite imperfect-information parity game).

- A set of prophecies for each existentially quantified variable with consistency requirements between prophecies.
- Careful setup so that prophecies do not "leak" information.
- As many players as variable alternations.
- The tower bounding the runtime of our algorithm is "taller" than the lower bound for HyperLTL model-checking.
- Also, we complement Büchi automata.