The Hierarchy of Hyperlogics

Norine Coenen, Bernd Finkbeiner, Christopher Hahn, Jana Hofmann

Reactive Systems Group, Saarland University, Germany

The 34th Annual ACM/IEEE Symposium on Logic in Computer Science

Vancouver, Canada, June 2019

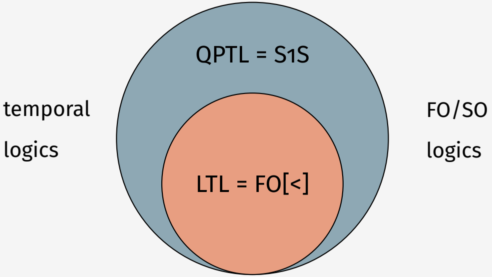
Reactive Systems Group

OSARES

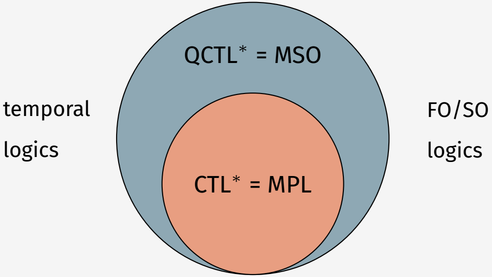
UNIVERSITÄT DES SAARLANDES

Logics for Trace Properties

Linear-Time Logics



Branching-Time Logics



Logics for Hyperproperties

Linear-Time Hyperlogics

temporal logic

HyperLTL

FO[<,E]

HyperQPTL

S1S[E]

FO/SO logic

Branching-Time Hyperlogics

temporal logic

HyperCTL*

MPL[E]

HyperQCTL*

MSO[E]

FO/SO logic

How do temporal and FO/SO hyperlogics relate w.r.t. expressiveness?

Satisfiability beyond HyperLTL?

Are Trace Properties Enough?

MELTDOWN

SPECTRE

side channels

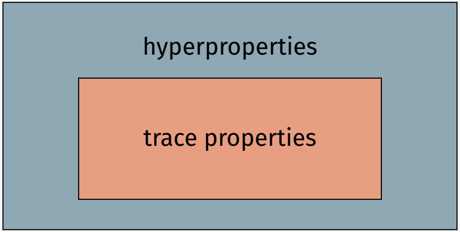
trace properties

Many proccessors are vulnerable even though proven correct.

The attacks compare multiple executions traces.

Hyperproperties

A trace property is a set of traces.

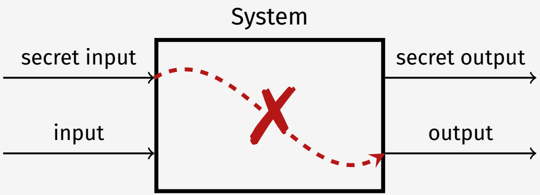


A hyperproperty[1] is a set of sets of traces.

Hyperproperties relate multiple execution traces.

[1] Clarkson, Schneider. *Hyperproperties.* Journal of Computer Security, 2010.

Hyperproperties in Information-Flow Control



- Trace equality: Do all execution traces agree on the value of a?
- Observational determinism: Does the system appear deterministic to low-security users?
- Uniform termination, noninterference, strong secrecy...

Two Paths to Hyperlogics

Temporal Hyperlogics

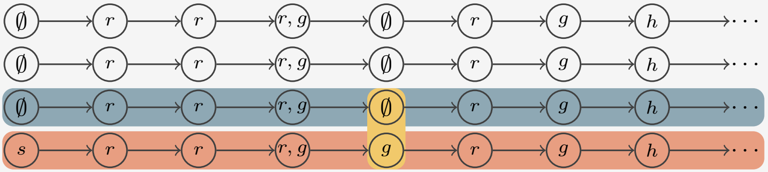
temporal logic	+	trace quantifiers / path quantifiers	=	temporal hyperlogic
LTL, CTL*, QPTL				*HyperLTL, HyperCTL*, HyperQPTL*

First-Order/Second-Order Hyperlogics

monadic FO/SO logic	+	equal-level predicate E	=	FO/SO hyperlogic
FO[<], MPL, S1S				*FO[<,E], MPL[E], S1S[E]*

Temporal Logics for Hyperproperties

HyperLTL = LTL + prenex trace quantifiers



Input: r = request

Outputs: g = grant

h = halt

s = grant secret

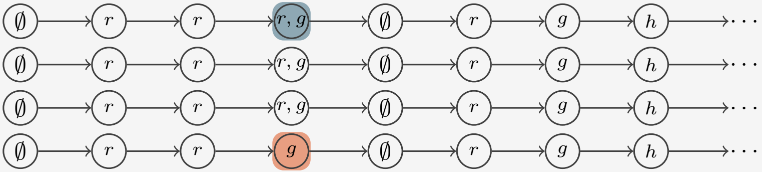
- Observational determinism:

$$\forall \pi . \forall \pi' . \Box (lowIn_{\pi} = lowIn_{\pi'}) \rightarrow \Box (lowOut_{\pi} = lowOut_{\pi'})$$

✗

FO/SO Hyperlogics

FO[<,E] = monadic FO logic of order + equal-level predicate E



Input: r = request

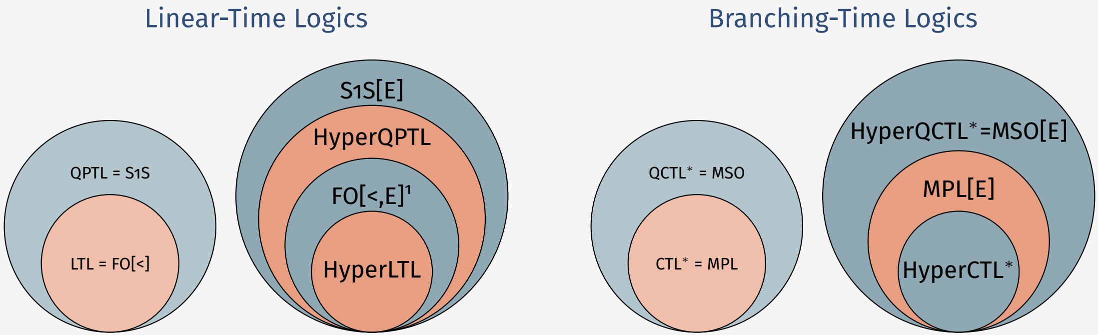
Outputs: g = grant

h = halt

s = grant secret

- **Trace equality:** $\forall x. \forall y. E(x, y) \rightarrow \left(\bigwedge_{P \in Pred} P(x) \leftrightarrow P(y) \right)$ ✗

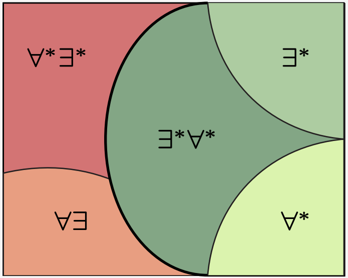
Expressiveness of Hyperlogics



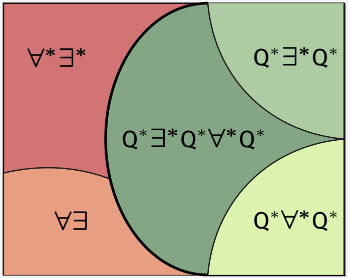
[1] Finkbeiner, Zimmermann. *The First-Order Logic of Hyperproperties*. STACS 2017.

Satisfiability of Hyperlogics

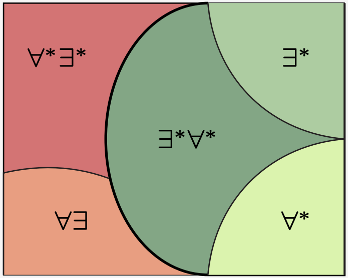
HyperLTL:[1]



HyperQPTL:



HyperCTL*:



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

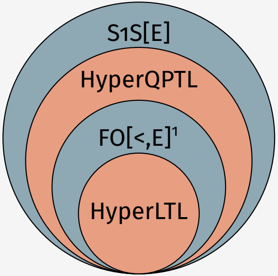
Overview

1. Linear-time hyperlogics

   Expressiveness results

   - The limits of HyperLTL
   - The more expressive HyperQPTL
   - The power of the equal-level predicate

   Deciding HyperQPTL



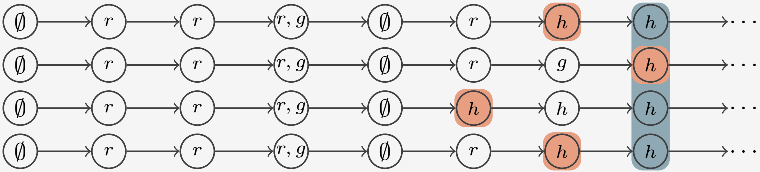
2. Branching-time hyperlogics

   Deciding HyperCTL*

The Limits of HyperLTL

- Promptness: "There is a bound up to which *all* traces fulfill a."

  Example: Uniform termination: "The system terminates within a bounded number of steps."



r = request
g = grant
h = halt

- Promptness is not expressible in HyperLTL.[1]

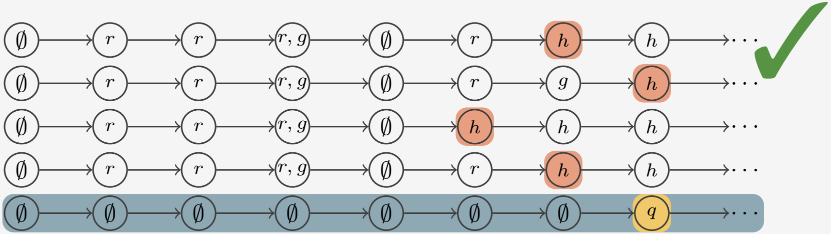
[1] Bozzelli, Maubert, Pinchinat. *Unifying Hyper and Epistemic Temporal Logics.* FoSSaCS 2015

HyperQPTL

HyperQPTL = QPTL + prenex trace quantifiers

= HyperLTL + quantification over propositional variables

- Uniform termination: $\exists q.\forall \pi.\Diamond q \wedge (\neg q\,\mathcal{U}\,h_\pi)$



r = request

g = grant

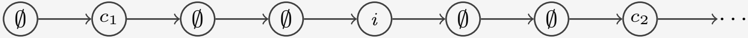
h = halt

The Power of the Equal-Level Predicate

The S1S[E] model checking problem is undecidable.

Proof by reduction from the halting problem of 2-counter machines:

- Decide if $T \vDash \varphi$, where:
- Encode each possible configuration $c = (instr, c_1, c_2)$ as a trace:



  for $c = (4, 1, 7)$

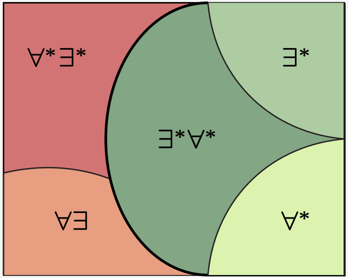
  T is the set of all encoded configurations.

- φ existentially quantifies a set of configurations that encodes a halting computation.

HyperQPTL Satisfiability

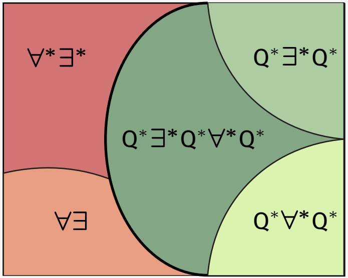
In general undecidable, decidable fragments:

HyperLTL:[1]



Decidability proofs: reduction to LTL

HyperQPTL:



Decidability proofs: reduction to QPTL

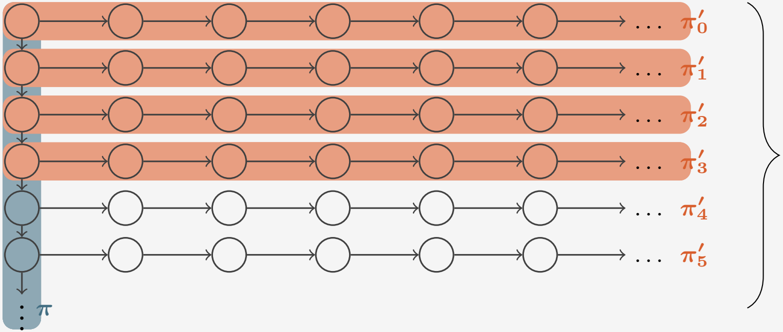
Propositional quantification does not change the decidability of the satisfiability problem.

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

HyperCTL* Satisfiability

HyperCTL* = CTL* + (non-prenex) path quantifiers

The interesting case: $\exists\pi.\Box(\exists\pi'.\varphi)$



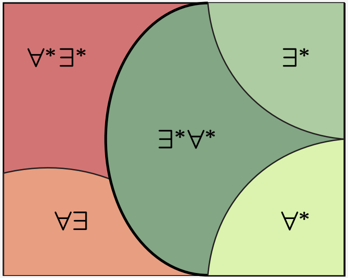
Does this lead to undecidability?

(It feels like ∃∀∃ after all...)

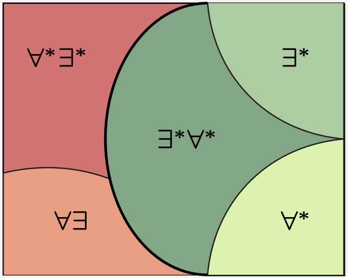
No!

HyperCTL* Satisfiability

HyperLTL:[1]



HyperCTL*:



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

Proving HyperCTL* Decidability

Roadmap

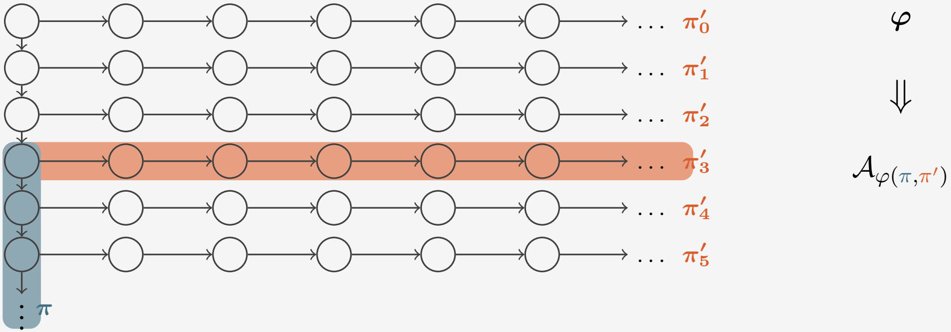
- Interesting case: $\exists^*$ fragment
- Exemplary proof for $\psi \coloneqq \exists\pi. \Box(\exists\pi'.\varphi)$

1. Label model with automaton states.
2. Define a cutting operation to cut out superfluous parts of the model.
3. Create a bounded representation of the model.

Decidability of the $\exists^*$ Fragment

Proof for: $\psi := \exists\pi.\square(\exists\pi'.\varphi)$

1. Label model with automaton states.

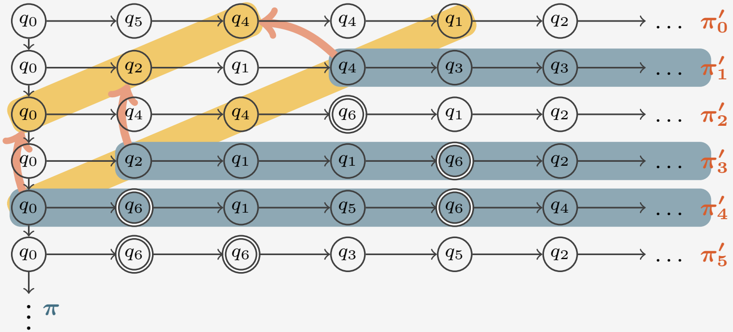


- Assumption: $\mathcal{A}_{\varphi(\pi,\pi')}$ accepts each $(p[i,\infty], p_i)$.

Decidability of the $\exists^*$ Fragment

Proof for: $\psi := \exists \pi . \Box (\exists \pi' . \varphi)$

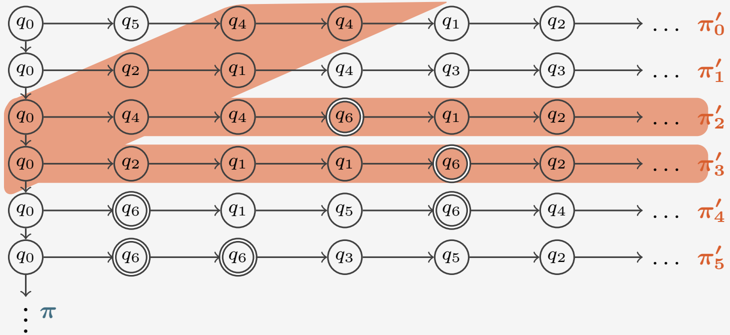
2. Define a cutting operation to cut out superfluous parts of the model.



Decidability of the $\exists^*$ Fragment

Proof for: $\psi := \exists\pi.\Box(\exists\pi'.\varphi)$

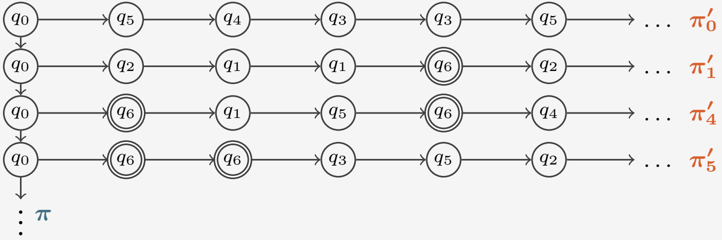
2. Define a cutting operation to cut out superfluous parts of the model.



Decidability of the $\exists^*$ Fragment

Proof for: $\psi \coloneqq \exists\pi.\square(\exists\pi'.\varphi)$

2. Define a cutting operation to cut out superfluous parts of the model.

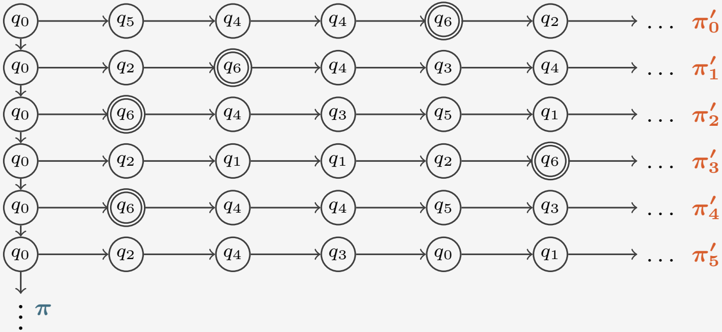


- Make sure the automaton run remains accepting.
- Do not cut accepting states.

Decidability of the ∃* Fragment

Proof for: $\psi := \exists\pi.\square(\exists\pi'.\varphi)$

3. Create a bounded representation of the model.

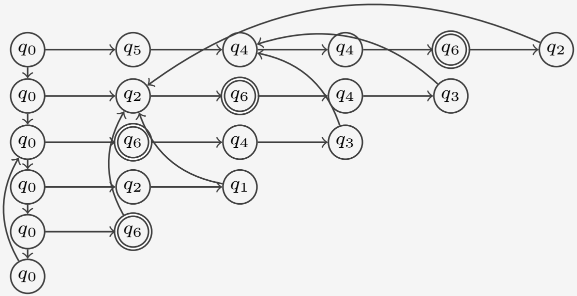


- Repeatedly cut out parts of the model until "enough" accepting sates are within a bound.

Decidability of the $\exists^*$ Fragment

Proof for: $\psi := \exists\pi.\Box(\exists\pi'.\varphi)$

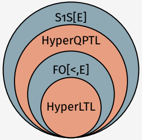
3. Create a bounded representation of the model.

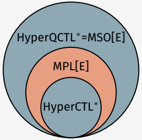


- Ensure: Accepting state on each loop.

Summary

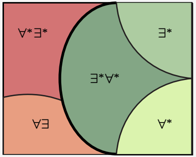
The expressiveness hierarchy of hyperlogics is different to the one for classic logics.



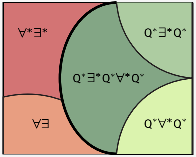


Mixing path quantifiers with propositional quantification and temporal operators does not affect the decidability of the satisfiability problem.

HyperLTL:



HyperQPTL:



HyperCTL*: