Bounded-Variable Fragments of Hybrid Logics

Volker Weber

Fachbereich Informatik, LS1
Universität Dortmund

joint work with Thomas Schwentick

Propositional Modal Logic

Syntax Given a set of proposition symbols $\{p, q, \ldots\}$ and a modality $\mathbf{F}$, the basic Modal Logic $\mathcal{ML}$ is defined as follows

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \mathbf{F}\varphi$$

- Abbreviations $\vee$, $\rightarrow$, $\leftrightarrow$ as usual; $\mathbf{G}\varphi = \neg\mathbf{F}\neg\varphi$

Propositional Modal Logic

Syntax Given a set of proposition symbols $\{p, q, \ldots\}$ and a modality $\mathbf{F}$, the basic Modal Logic $\mathcal{ML}$ is defined as follows

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \mathbf{F}\varphi$$

- Abbreviations $\vee$, $\rightarrow$, $\leftrightarrow$ as usual; $\mathbf{G}\varphi = \neg\mathbf{F}\neg\varphi$

Semantics via Kripke models $\mathcal{M} = (S, R, V)$

- S is a non-empty set of states
- R is a binary relation on S, the accessibility relation
- V is a valuation assigning subsets of S to proposition symbols

Propositional Modal Logic

Syntax Given a set of proposition symbols $\{p, q, \ldots\}$ and a modality $\mathbf{F}$, the basic Modal Logic $\mathcal{ML}$ is defined as follows

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \mathbf{F}\varphi$$

- Abbreviations $\vee$, $\rightarrow$, $\leftrightarrow$ as usual; $\mathbf{G}\varphi = \neg\mathbf{F}\neg\varphi$

Semantics via Kripke models $\mathcal{M} = (S, R, V)$

- S is a non-empty set of states
- R is a binary relation on S, the accessibility relation
- V is a valuation assigning subsets of S to proposition symbols

Temporal Logic $\mathcal{TL}$ past modality $\mathbf{P}$, $\mathbf{H}\varphi = \neg\mathbf{P}\neg\varphi$

Propositional Modal Logic

Syntax Given a set of proposition symbols $\{p, q, \ldots\}$ and a modality $\mathbf{F}$, the basic Modal Logic $\mathcal{ML}$ is defined as follows

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \mathbf{F}\varphi$$

- Abbreviations $\vee$, $\rightarrow$, $\leftrightarrow$ as usual; $\mathbf{G}\varphi = \neg\mathbf{F}\neg\varphi$

Semantics via Kripke models $\mathcal{M} = (S, R, V)$

- S is a non-empty set of states
- R is a binary relation on S, the accessibility relation
- V is a valuation assigning subsets of S to proposition symbols

Temporal Logic $\mathcal{TL}$ past modality $\mathbf{P}$, $\mathbf{H}\varphi = \neg\mathbf{P}\neg\varphi$

What $\mathcal{ML}$ and $\mathcal{TL}$ cannot do: Talk about individual states

Hybrid Logic

Nominals

- A second sort of atomic formula
- Each nominal is true at exactly one state in any model
- I.e., a nominal names a state
- Nominals correspond to constants in first-order logic

$\mathcal{HL} = \mathcal{ML}$ plus nominals, $\mathcal{HTL} = \mathcal{TL}$ plus nominals

The root of a tree

- Name the root by a nominal *root*

Hybrid Logic

Nominals

- A second sort of atomic formula
- Each nominal is true at exactly one state in any model
- I.e., a nominal names a state
- Nominals correspond to constants in first-order logic

$\mathcal{HL} = \mathcal{ML}$ plus nominals, $\mathcal{HTL} = \mathcal{TL}$ plus nominals

The @-Operator

- Jumps to a named state

The root of a tree

- Name the root by a nominal *root*
- Talk about the root: $@_{root}\varphi$

Hybrid Logic

The $\downarrow$-Operator

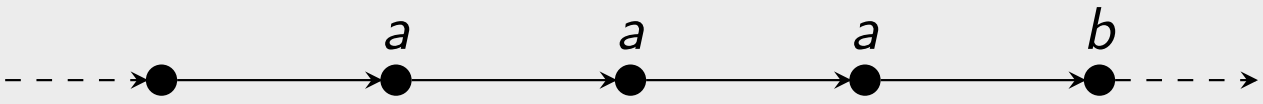
- Binds state variables to states
- $\downarrow x.\varphi \equiv$ name the current state x and evaluate φ
- $x \equiv$ check if the current state is named x
- Combines nicely with the @-Operator

The $\downarrow$-Operator

- Binds state variables to states
- $\downarrow x.\varphi \equiv$ name the current state x and evaluate φ
- $x \equiv$ check if the current state is named x
- Combines nicely with the @-Operator

Expressing UNTIL with $\downarrow$ and @

- $a\mathbf{U}b$



In all examples: Edges caused by transitivity are left out

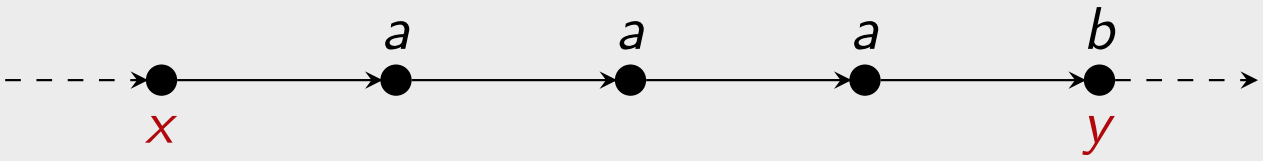
Hybrid Logic

The $\downarrow$-Operator

- Binds state variables to states
- $\downarrow x.\varphi \equiv$ name the current state x and evaluate φ
- $x \equiv$ check if the current state is named x
- Combines nicely with the @-Operator

Expressing **UNTIL** with $\downarrow$ and @

- $a\mathbf{U}b \quad \equiv \quad \downarrow x.\mathbf{F} \downarrow y.(b \wedge @_x\mathbf{G}(\mathbf{F}y \rightarrow a))$



In all examples: Edges caused by transitivity are left out

Complexity of Hybrid Satisfiability Problems

models	$\mathcal{HL}(\downarrow)$	$\mathcal{HL}(\downarrow, @)$	$\mathcal{HTL}(\downarrow)$	$\mathcal{HTL}(\downarrow, @)$
arbitrary	undec.	undec.	undec.	undec.
linear / $(\mathbb{N}, <)$	**NP**	non-elem.	non-elem.	non-elem.

- Areces, Blackburn, Marx 1999
- Franceschet, de Rijke, Schlingloff 2003
- Mundhenk, Schneider, Schwentick, W. 2005

Complexity of Hybrid Satisfiability Problems

models	$\mathcal{HL}(\downarrow)$	$\mathcal{HL}(\downarrow, @)$	$\mathcal{HTL}(\downarrow)$	$\mathcal{HTL}(\downarrow, @)$
arbitrary	undec.	undec.	undec.	undec.
linear / $(\mathbb{N}, <)$	**NP**	non-elem.	non-elem.	non-elem.

- Areces, Blackburn, Marx 1999
- Franceschet, de Rijke, Schlingloff 2003
- Mundhenk, Schneider, Schwentick, W. 2005

Are there decidable fragments / fragments of lower complexity?

- ten Cate, Franceschet 2005
  - Disallowed patterns, e.g. $\mathbf{G} \downarrow \mathbf{G}$
  - Models of bounded width
- Our approach
  - Bounded-variable fragments

Bounded-Variable Fragments

- Only k different variable symbols for a fixed k
- $\mathcal{HL}(\downarrow^k)$, $\mathcal{HL}(\downarrow^k, @)$, $\mathcal{HTL}(\downarrow^k)$, $\mathcal{HTL}(\downarrow^k, @)$

Bounded-Variable Fragments

- Only k different variable symbols for a fixed k
- $\mathcal{HL}(\downarrow^k)$, $\mathcal{HL}(\downarrow^k, @)$, $\mathcal{HTL}(\downarrow^k)$, $\mathcal{HTL}(\downarrow^k, @)$

Theorem

Model checking for $\mathcal{HTL}(\downarrow^k, @)$ can be done in polynomial time.

- **PSPACE**-c. for $\mathcal{HTL}(\downarrow, @)$ [Franceschet, de Rijke 2005]

Bounded-Variable Fragments

- Only k different variable symbols for a fixed k
- $\mathcal{HL}(\downarrow^k)$, $\mathcal{HL}(\downarrow^k, @)$, $\mathcal{HTL}(\downarrow^k)$, $\mathcal{HTL}(\downarrow^k, @)$

Theorem

Model checking for $\mathcal{HTL}(\downarrow^k, @)$ can be done in polynomial time.

- **PSPACE**-c. for $\mathcal{HTL}(\downarrow, @)$ [Franceschet, de Rijke 2005]

Theorem (Marx 2002, ten Cate, Franceschet 2005)

Satisfiability of $\mathcal{HL}(\downarrow^1)$ is undecidable.

Bounded-Variable Fragments

- Only k different variable symbols for a fixed k
- $\mathcal{HL}(\downarrow^k)$, $\mathcal{HL}(\downarrow^k, @)$, $\mathcal{HTL}(\downarrow^k)$, $\mathcal{HTL}(\downarrow^k, @)$

Theorem

Model checking for $\mathcal{HTL}(\downarrow^k, @)$ can be done in polynomial time.

- **PSPACE**-c. for $\mathcal{HTL}(\downarrow, @)$ [Franceschet, de Rijke 2005]

Theorem (Marx 2002, ten Cate, Franceschet 2005)

Satisfiability of $\mathcal{HL}(\downarrow^1)$ is undecidable.

What about restricted classes of models?

Here: Natural Numbers with Order $(\mathbb{N}, <)$

Bounded-Variable Fragments

- Only k different variable symbols for a fixed k
- $\cancel{\mathcal{HL}(\downarrow^k)}$, $\mathcal{HL}(\downarrow^k, @)$, $\mathcal{HTL}(\downarrow^k)$, $\cancel{\mathcal{HTL}(\downarrow^k, @)}$

Theorem

Model checking for $\mathcal{HTL}(\downarrow^k, @)$ can be done in polynomial time.

- **PSPACE**-c. for $\mathcal{HTL}(\downarrow, @)$ [Franceschet, de Rijke 2005]

Theorem (Marx 2002, ten Cate, Franceschet 2005)

Satisfiability of $\mathcal{HL}(\downarrow^1)$ is undecidable.

What about restricted classes of models?

Here: Natural Numbers with Order $(\mathbb{N}, <)$

Expressiveness

Theorem

$\mathcal{HL}(\downarrow^2, @)$ and $\mathcal{HTL}(\downarrow^1)$ are expressively complete over the natural numbers.

Proof: $\mathcal{LTL}$ is expressively complete [Gabbay et al. 1980, 1989]

$$a\mathbf{U}b = \downarrow x.\mathbf{F}(b \wedge \mathbf{H}(\mathbf{P}x \rightarrow a))$$

Expressiveness

Theorem

$\mathcal{HL}(\downarrow^2, @)$ and $\mathcal{HTL}(\downarrow^1)$ are expressively complete over the natural numbers.

Proof: $\mathcal{LTL}$ is expressively complete [Gabbay et al. 1980, 1989]

$$a\mathbf{U}b = \downarrow x.\mathbf{F}(b \wedge \mathbf{H}(\mathbf{P}x \rightarrow a))$$

Theorem

The inclusions $\mathcal{HL}(@) \subset \mathcal{HL}(\downarrow^1, @) \subset \mathcal{HL}(\downarrow^2, @)$ are strict with respect to expressivity.

Proof: Bisimulation arguments

Expressiveness

Theorem

$\mathcal{HL}(\downarrow^2, @)$ and $\mathcal{HTL}(\downarrow^1)$ are expressively complete over the natural numbers.

Proof: $\mathcal{LTL}$ is expressively complete [Gabbay et al. 1980, 1989]

$$a\mathbf{U}b = \downarrow x.\mathbf{F}(b \wedge \mathbf{H}(\mathbf{P}x \rightarrow a))$$

Theorem

The inclusions $\mathcal{HL}(@) \subset \mathcal{HL}(\downarrow^1, @) \subset \mathcal{HL}(\downarrow^2, @)$ are strict with respect to expressivity.

Proof: Bisimulation arguments

Theorem

$\mathcal{HL}(\downarrow^1, @)$ is expressively complete over the natural numbers with zero $(\mathbb{N}, <, 0)$.

The Two-Variable Fragment

Theorem

The satisfiability problem of $\mathcal{HL}(\downarrow^2, @)$ over the natural numbers has non-elementary complexity.

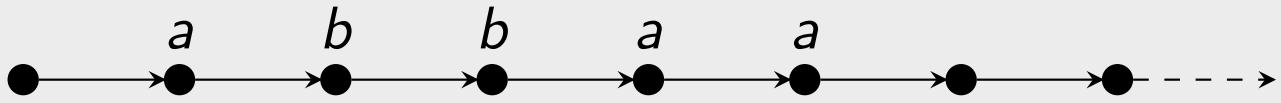
The Two-Variable Fragment

Theorem

The satisfiability problem of $\mathcal{HL}(\downarrow^2, @)$ over the natural numbers has non-elementary complexity.

Proof: Reduction from the non-emptiness problem for star-free expressions (union, concatenation, and negation).

The string *abbaa*



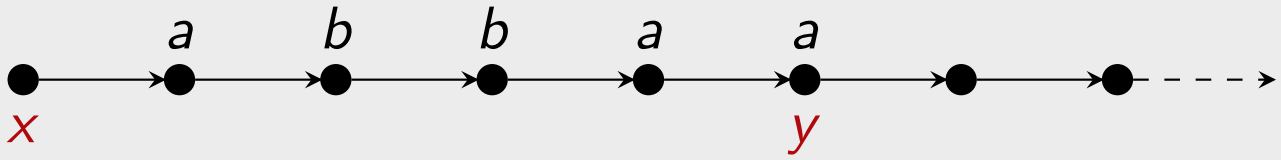
The Two-Variable Fragment

Theorem

The satisfiability problem of $\mathcal{HL}(\downarrow^2, @)$ over the natural numbers has non-elementary complexity.

Proof: Reduction from the non-emptiness problem for star-free expressions (union, concatenation, and negation).

The string $abb \cdot aa$ - Dealing with concatenation



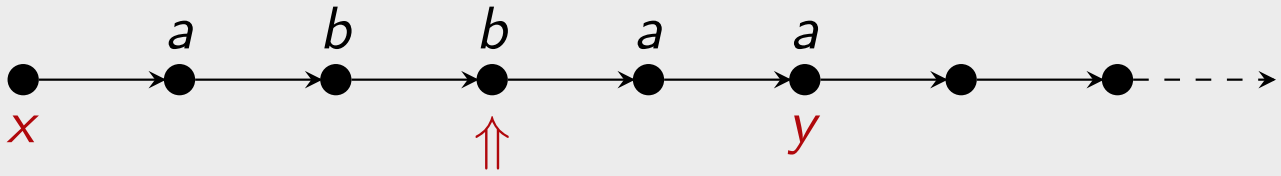
The Two-Variable Fragment

Theorem

The satisfiability problem of $\mathcal{HL}(\downarrow^2, @)$ over the natural numbers has non-elementary complexity.

Proof: Reduction from the non-emptiness problem for star-free expressions (union, concatenation, and negation).

The string $abb \cdot aa$ - Dealing with concatenation



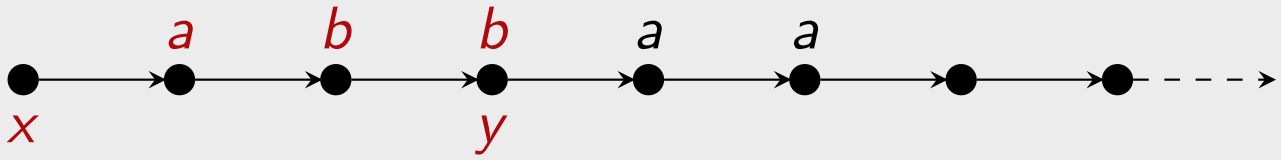
The Two-Variable Fragment

Theorem

The satisfiability problem of $\mathcal{HL}(\downarrow^2, @)$ over the natural numbers has non-elementary complexity.

Proof: Reduction from the non-emptiness problem for star-free expressions (union, concatenation, and negation).

The string $abb \cdot aa$ - Dealing with concatenation



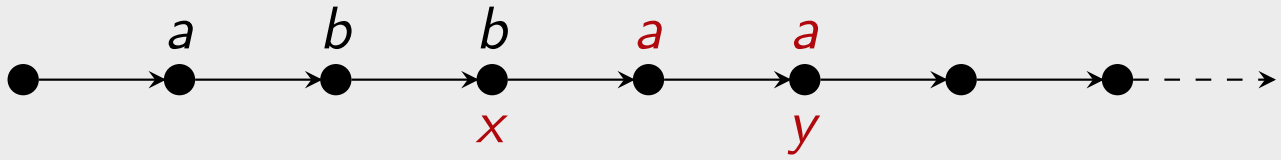
The Two-Variable Fragment

Theorem

The satisfiability problem of $\mathcal{HL}(\downarrow^2, @)$ over the natural numbers has non-elementary complexity.

Proof: Reduction from the non-emptiness problem for star-free expressions (union, concatenation, and negation).

The string $abb \cdot aa$ - Dealing with concatenation



The One-Variable Fragment

Theorem

The satisfiability problems of $\mathcal{HL}(\downarrow^1, @)$ and $\mathcal{HTL}(\downarrow^1)$ over the natural numbers are **EXPSPACE**-complete.

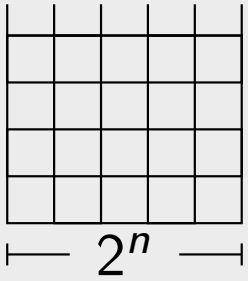
The One-Variable Fragment

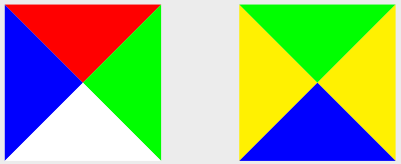
Theorem

The satisfiability problems of $\mathcal{HL}(\downarrow^1, @)$ and $\mathcal{HTL}(\downarrow^1)$ over the natural numbers are **EXPSPACE**-complete.

Proof: Lower bound: Reduction from the $2^n \times m$ tiling problem

Tilings





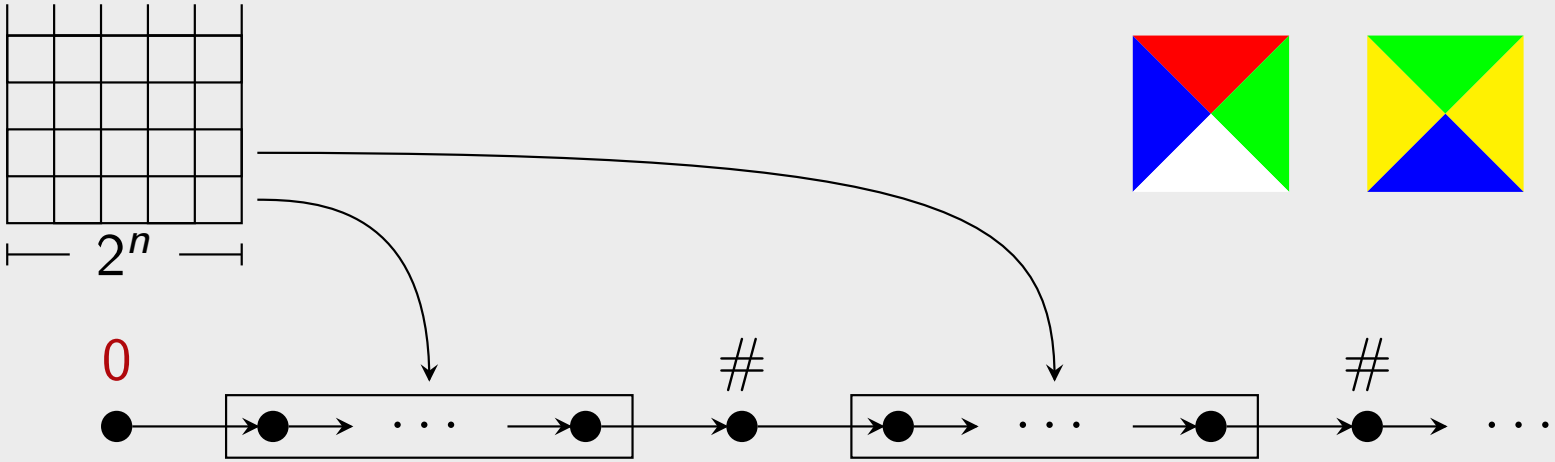
The One-Variable Fragment

Theorem

The satisfiability problems of $\mathcal{HL}(\downarrow^1, @)$ and $\mathcal{HTL}(\downarrow^1)$ over the natural numbers are **EXPSPACE**-complete.

Proof: Lower bound: Reduction from the $2^n \times m$ tiling problem

Tilings: Encoding



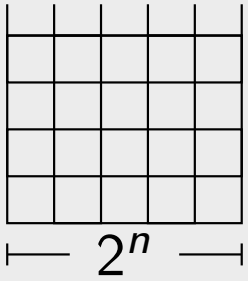
The One-Variable Fragment

Theorem

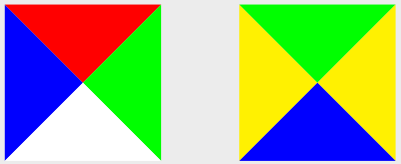
The satisfiability problems of $\mathcal{HL}(\downarrow^1, @)$ and $\mathcal{HTL}(\downarrow^1)$ over the natural numbers are **EXPSPACE**-complete.

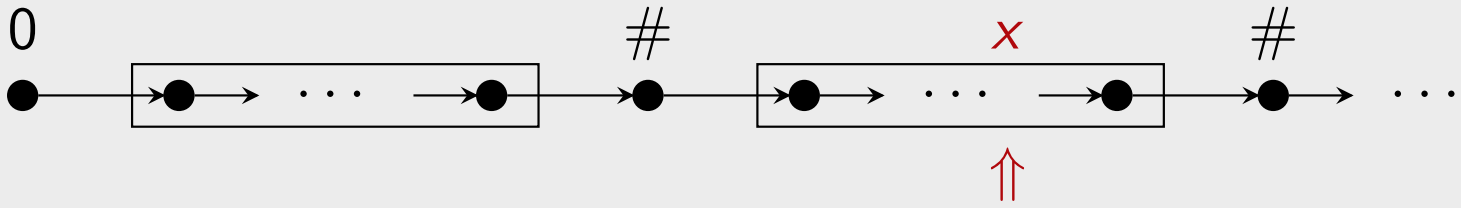
Proof: Lower bound: Reduction from the $2^n \times m$ tiling problem

Tilings: Encoding and Verification



$\downarrow x.$





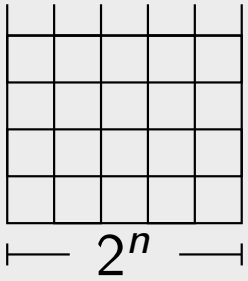
The One-Variable Fragment

Theorem

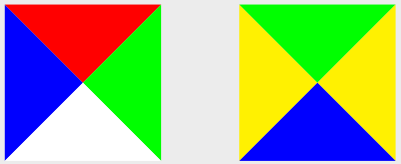
The satisfiability problems of $\mathcal{HL}(\downarrow^1, @)$ and $\mathcal{HTL}(\downarrow^1)$ over the natural numbers are **EXPSPACE**-complete.

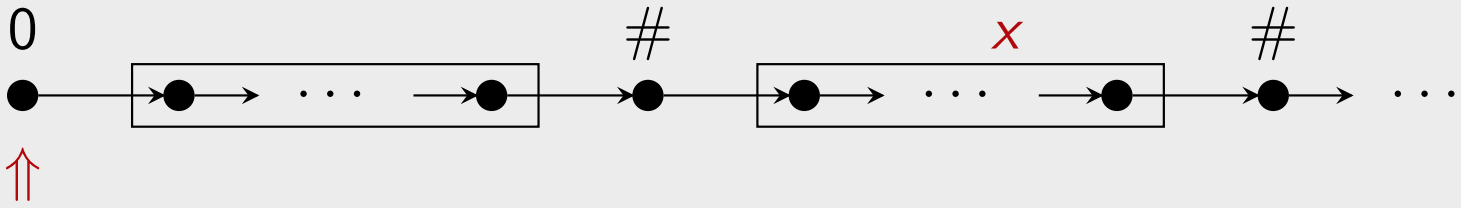
Proof: Lower bound: Reduction from the $2^n \times m$ tiling problem

Tilings: Encoding and Verification



$\downarrow x.@_0$





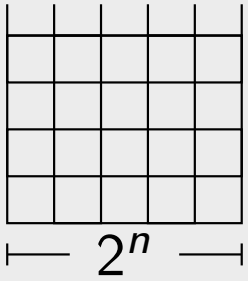
The One-Variable Fragment

Theorem

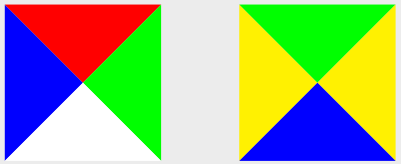
The satisfiability problems of $\mathcal{HL}(\downarrow^1, @)$ and $\mathcal{HTL}(\downarrow^1)$ over the natural numbers are **EXPSPACE**-complete.

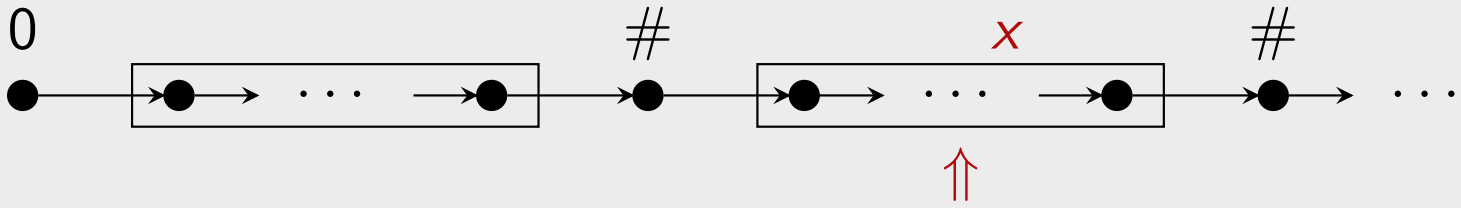
Proof: Lower bound: Reduction from the $2^n \times m$ tiling problem

Tilings: Encoding and Verification



$$\downarrow x.@_0\mathbf{F}(\mathbf{F}x \wedge \neg\mathbf{FF}x \wedge \ldots)$$





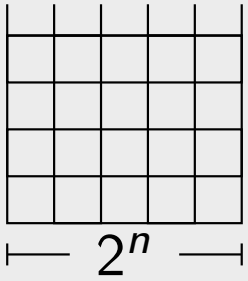
The One-Variable Fragment

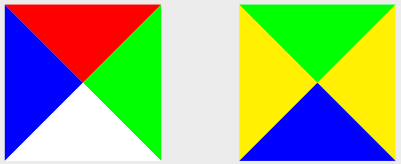
Theorem

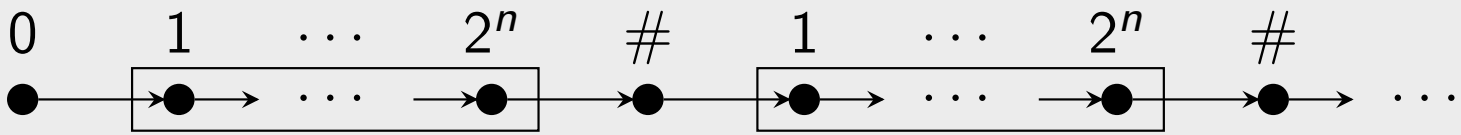
The satisfiability problems of $\mathcal{HL}(\downarrow^1, @)$ and $\mathcal{HTL}(\downarrow^1)$ over the natural numbers are **EXPSPACE**-complete.

Proof: Lower bound: Reduction from the $2^n \times m$ tiling problem

Tilings: Encoding and Verification







The One-Variable Fragment

Theorem

The satisfiability problems of $\mathcal{HL}(\downarrow^1, @)$ and $\mathcal{HTL}(\downarrow^1)$ over the natural numbers are **EXPSPACE**-complete.

Proof: Lower bound: Reduction from the $2^n \times m$ tiling problem
Upper Bound: **Automata-theoretic approach**

- alternating Büchi automata and $\mathcal{LTL}$ [Vardi et al. since 1994]
- temporal logic $\rightarrow$ two-wayness
- $\downarrow^1 \rightarrow$ one pebble

The One-Variable Fragment

Theorem

The satisfiability problems of $\mathcal{HL}(\downarrow^1, @)$ and $\mathcal{HTL}(\downarrow^1)$ over the natural numbers are **EXPSPACE**-complete.

Proof: Lower bound: Reduction from the $2^n \times m$ tiling problem
Upper Bound: **Automata-theoretic approach**

- alternating Büchi automata and $\mathcal{LTL}$ [Vardi et al. since 1994]
- temporal logic $\rightarrow$ two-wayness
- $\downarrow^1 \rightarrow$ one pebble

Theorem

The non-emptiness problem for alternating one-pebble Büchi automata is **EXPSPACE**-complete.

Alternating One-Pebble Büchi Automata

Theorem

The non-emptiness problem for alternating one-pebble Büchi automata is **EXPSPACE**-complete.

Proof: Lower bound: Corresponding result for finite strings

Alternating One-Pebble Büchi Automata

Theorem

The non-emptiness problem for alternating one-pebble Büchi automata is **EXPSPACE**-complete.

Proof: Lower bound: Corresponding result for finite strings
Upper Bound:

- Existence of **memoryless runs**
  - accepting run $\equiv$ winning strategy in a two-player game on the configuration graph
  - existence of memoryless winning strategies [Emerson, Jutla 91]
  - Thanks to an anonymous referee for this nice proof
- Simulation by **non-deterministic one-way Büchi automata** of double exponential size
  - checks existence of a memoryless accepting run
  - uses behavioral functions
  - in contrast to the finite case: local consistency does not suffice

Conclusion

- Hybrid $\downarrow$-languages are undecidable / have non-elementary complexity in general.

$\rightarrow$ **Bounded-variable fragments over the natural numbers**

- The one-variable fragment
  - Has full expressive power
  - Model checking can be done in polynomial time
  - Satisfiability is **EXPSPACE**-complete
- The two-variable fragment has non-elementary complexity

Conclusion

- Hybrid $\downarrow$-languages are undecidable / have non-elementary complexity in general.

$\rightarrow$ **Bounded-variable fragments over the natural numbers**

- The one-variable fragment
  - Has full expressive power
  - Model checking can be done in polynomial time
  - Satisfiability is **EXPSPACE**-complete
- The two-variable fragment has non-elementary complexity

On the way

- Non-emptiness for alternating one-pebble Büchi automata is **EXPSPACE**-complete

Conclusion

- Hybrid $\downarrow$-languages are undecidable / have non-elementary complexity in general.

$\rightarrow$ **Bounded-variable fragments over the natural numbers**

- The one-variable fragment
  - Has full expressive power
  - Model checking can be done in polynomial time
  - Satisfiability is **EXPSPACE**-complete
- The two-variable fragment has non-elementary complexity

On the way

- Non-emptiness for alternating one-pebble Büchi automata is **EXPSPACE**-complete

What could be next?

- Logic: extend results to trees

$\rightarrow$ same for automata (Conjecture: **2EXPTIME**)

- Automata: extend results to k-pebble Büchi automata