Definability problem and automata for first order logic over trees

Angelo Matteo, University of Udine

joint work with M. Benerecetti, D. Della Monica, F. Mogavero and G. Puppis

informal Formal Methods Meetings,
October 8, 2025

Regular = Monadic Second Order Logic

One of the most fundamental results in language theory connects **regular word languages** and **logic**.

Regular = Monadic Second Order Logic

One of the most fundamental results in language theory connects **regular word languages** and **logic**.

Theorem (Büchi, Elgot, Trakhtenbrot, late 1950s)
A language of finite words L over a finite alphabet Σ is regular iff it is definable in Monadic Second Order Logic.

Regular = Monadic Second Order Logic

One of the most fundamental results in language theory connects **regular word languages** and **logic**.

Theorem (Büchi, Elgot, Trakhtenbrot, late 1950s)
A language of finite words L over a finite alphabet Σ is regular iff it is definable in Monadic Second Order Logic.

The same is true when one replaces word languages with **tree languages.**

Regular = Monadic Second Order Logic

One of the most fundamental results in language theory connects **regular word languages** and **logic**.

Theorem (Büchi, Elgot, Trakhtenbrot, late 1950s)
A language of finite words L over a finite alphabet Σ is regular iff it is definable in Monadic Second Order Logic.

The same is true when one replaces word languages with **tree languages.**

Theorem (Doner, Thatcher & Wright, mid 1960s)
A language of finite trees L over a finite alphabet Σ is regular iff it is definable in Monadic Second Order Logic.

Regular = Monadic Second Order Logic

One of the most fundamental results in language theory connects **regular word languages** and **logic**.

Theorem (Büchi, Elgot, Trakhtenbrot, late 1950s)
A language of finite words L over a finite alphabet Σ is regular iff it is definable in Monadic Second Order Logic.

The same is true when one replaces word languages with **tree languages.**

Theorem (Doner, Thatcher & Wright, mid 1960s)
A language of finite trees L over a finite alphabet Σ is regular iff it is definable in Monadic Second Order Logic.

Many generalizations: infinite words, infinite trees, even **graphs of bounded treewidth**!

Monadic Second Order Logic

Monadic Second Order Logic (MSO) is Second Order Logic in which second order quantification is limited to monadic predicates, i.e., **sets.**

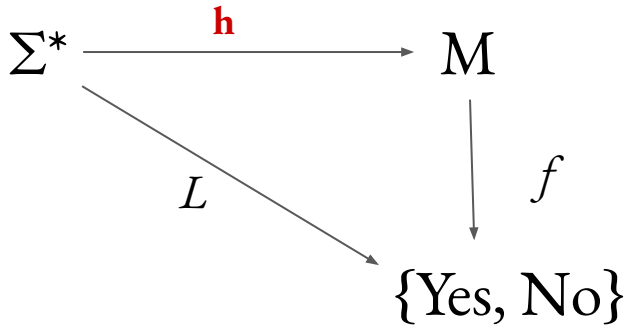
Over words, MSO can quantify over **sets of "positions"** of the word.
Over trees, it can quantify over **sets of nodes** of the tree.

Another notion of regularity (for words)

A finite **monoid M** consists of a finite set S equipped with a binary multiplication operation, which is associative, and a neutral element.

A monoid **homomorphism** is a function between underlying sets of monoids, that preserves the multiplication operation.

A language $L \subseteq \Sigma^*$ is **recognized** by a monoid homomorphism iff

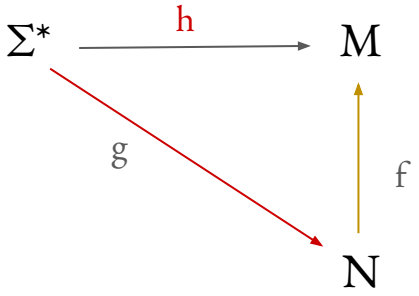


where M is a finite monoid and *f* is some function that specifies the "accepting set" of M

Another notion of regularity (for words)

Theorem
A language of finite words L over a finite alphabet Σ is regular iff it is recognized by a finite monoid homomorphism.

The **syntactic monoid** of a language L is the target of a **surjective** monoid homomorphism h, such that it recognizes L and the following commutes:



For every surjective homomorphism that recognises L, there is a unique surjective homomorphism that extends it to h.

Every regular language has an **effectively presentable** syntactic monoid.

A natural question

MSO = regular (word) languages = finite monoids

A natural question

MSO = regular (word) languages = finite monoids

What about strict **fragments** of MSO? Is it possible to obtain similar characterisations for less powerful logics?

A natural question

MSO = regular (word) languages = finite monoids

What about strict **fragments** of MSO? Is it possible to obtain similar characterisations for less powerful logics?

Given a fragment F of MSO, is the following decidable?

Input: A regular language L
Question: Is L definable in F?

Positive answers

Over finite words, many positive answers.

Positive answers

Over finite words, many positive answers.

Theorem (Schützenberger (1965))

First Order Logic [<] = star-free regular languages = Aperiodic monoids ($a^{\lambda} = a^{\lambda+1}$)

Positive answers

Over finite words, many positive answers.

Theorem (Schützenberger (1965))

First Order Logic [<] = star-free regular languages = Aperiodic monoids ($a^{\lambda} = a^{\lambda+1}$)

Theorem (Simon (1975))

Boolean combination of Σ_1 = piecewise testable languages= *J*-trivial monoids ($MaM = Ma'M \rightarrow a = a'$)

Positive answers

Over finite words, many positive answers.

Theorem (Schützenberger (1965))

First Order Logic [<] = star-free regular languages = Aperiodic monoids ($a^\lambda = a^{\lambda+1}$)

Theorem (Simon (1975))

Boolean combination of Σ_1 = piecewise testable languages= *J*-trivial monoids ($MaM = Ma'M \rightarrow a = a'$)

Theorem (Pin & Weil (1997))

FO[<] with two variables = DA monoids ($(ab)^\lambda = (ab)^\lambda a(ab)^\lambda$)

Why is this effective?

The previous characterisations all yielded **algorithms** to decide the definability in the given logic. The algorithms works this way:

1. Take an arbitrary representation of the language (regular expressions, MSO, automata, ...).
2. Transform the representation of the language in a finite monoid.
3. Compute the syntactic monoid of the language.
4. Check if the syntactic monoid satisfies the equation specified in the theorem.

What about tree languages?

How can one approach this problem for **trees**?

First definitions and approaches: Thomas (1984, 1987).

In the following, finite and infinite trees will be mixed without a lot of precision.

Logics for trees

Thomas identifies three “fragment” of MSO over trees:

- Monadic **Antichain** Logic: set quantification limited to **incomparable nodes** wrt <.
- Monadic **Chain** Logic: set quantification limited to **comparable nodes** wrt **<.**
- FO/Monadic **Path** logic: quantification limited to nodes/paths.

Aperiodicity for trees

A **context** is a tree with exactly **one** special constant instead of a leaf (or of an infinite subtree, if we are in the case of infinite trees).

Given a finite alphabet Σ, the set S_Σ of all the contexts obtainable from Σ is a monoid with "**tree concatenation**" as the multiplication operation.

Aperiodicity for trees

A **context** is a tree with exactly **one** special constant instead of a leaf (or of an infinite subtree, if we are in the case of infinite trees).

Given a finite alphabet Σ, the set S_Σ of all the contexts obtainable from Σ is a monoid with "**tree concatenation**" as the multiplication operation.

A tree language T is **aperiodic** if for every element a of its syntactic monoid it holds $a^\lambda = a^{\lambda+1}$.

Monadic Antichain Logic = MSO

The definability problem for Antichain Logic is **trivial**, because Potthoff & Thomas (1993) show that

Monadic Antichain Logic = MSO

over trees.

Monadic Chain Logic

Chain Logic is a very well studied logic over trees. What is known:

Monadic Chain Logic

Chain Logic is a very well studied logic over trees. What is known:

1. It is **incomparable** with aperiodicity.

Monadic Chain Logic

Chain Logic is a very well studied logic over trees. What is known:

1. It is **incomparable** with aperiodicity.
2. There are many different **modal logics** effectively equivalent to it.

Monadic Chain Logic

Chain Logic is a very well studied logic over trees. What is known:

1. It is **incomparable** with aperiodicity.
2. There are many different **modal logics** effectively equivalent to it.
3. Two recent (2024) **automaton-based characterisations**.

Monadic Chain Logic

Chain Logic is a very well studied logic over trees. What is known:

1. It is **incomparable** with aperiodicity.
2. There are many different **modal logics** effectively equivalent to it.
3. Two recent (2024) **automaton-based characterisations**.

No clue about an algebraic characterisation. A conjecture (Bojanczyk, 2004).

First Order Logic

What is known:

First Order Logic

What is known:

1. Thomas (1984) shows that every FO-definable tree language is **aperiodic.** Heuter (1991) shows that there is an aperiodic language that is not FO-definable.

First Order Logic

What is known:

1. Thomas (1984) shows that every FO-definable tree language is **aperiodic.** Heuter (1991) shows that there is an aperiodic language that is not FO-definable.
2. There are many different **modal logics** effectively equivalent to it.

First Order Logic

What is known:

1. Thomas (1984) shows that every FO-definable tree language is **aperiodic.** Heuter (1991) shows that there is an aperiodic language that is not FO-definable.
2. There are many different **modal logics** effectively equivalent to it.
3. There are two **automaton-based characterisations** (this work).

Again, no clue about an algebraic characterisations, not even a conjecture.

Side problem

Chain Logic and FO are very close. Can one at least decide if a
Chain-definable tree language is FO-definable?

Side problem

Chain Logic and FO are very close. Can one at least decide if a
Chain-definable tree language is FO-definable?

The answer is tricky and currently **unknown**. A natural conjecture, i.e.,

$$\textbf{Chain Logic} \cap \textbf{Aperiodic} = \textbf{FO}$$

was proved false by Potthoff (1995).

Side problem

Chain Logic and FO are very close. Can one at least decide if a **Chain-definable tree language is FO-definable**?

The answer is tricky and currently **unknown**. A natural conjecture, i.e.,

$$\textbf{Chain Logic} \cap \textbf{Aperiodic} = \textbf{FO}$$

was proved false by Potthoff (1995).

Solving this problem would reduce the FO-definability problem to the Chain Logic-definability.

Other directions

Many other algebraic frameworks were pursued after Thomas. The two most important ones are:

1. **preclones** (Esik & Weil, 2005)
2. **forest algebras** (Bojanczyk & Walukiewicz, 2008)

In both these formalisms (2009, 2012) a **non-effective** characterisation of FO-definability is obtained. Again, no insight about an **effective** characterisation.

Positive answers

There are some positive results. For the following logics, the definability problem over trees is decidable:

- temporal logics EX, EF, EX+EF, EF + $F^{\leftarrow}$.
- Boolean combinations of Σ_1 formulas.
- Δ_2.
- FO[S_1,...S_k].
- FO[<] with two variables.

A different approach?

What's missing for regular tree languages? A **strong algebraic framework**.

A different approach?

What's missing for regular tree languages? A **strong algebraic framework**. This is even more true for **infinite trees.**

A different approach?

What's missing for regular tree languages? A **strong algebraic framework**. This is even more true for **infinite trees.**

One could argue that solving the definability problem means **to understand deeply a language class**.

A different approach?

What's missing for regular tree languages? A **strong algebraic framework**. This is even more true for **infinite trees.**

One could argue that solving the definability problem means **to understand deeply a language class**.

Is there **another way** to gain a decent understanding of a language class?

What about automata?

Automaton-based characterisations can help in getting a good understanding of a logic over certain structures, even without solving the definability problem.

What about automata?

Automaton-based characterisations can help in getting a good understanding of a logic over certain structures, even without solving the definability problem.

Prominent example: **automaton for μ-calculus** by Janin & Walukiewicz (1996).

What about automata?

Automaton-based characterisations can help in getting a good understanding of a logic over certain structures, even without solving the definability problem.

Prominent example: **automaton for μ-calculus** by Janin & Walukiewicz (1996).

Recent advances: **automata for Monadic Path Logic and Monadic Chain Logic** (Bozzelli, Benerecetti, Mogavero & Peron, 2024).

Automata for FO over trees

Two past efforts:

Automata for FO over trees

Two past efforts:

1. M. Bojanczyk (2004) proved the equivalence over **finite binary trees** of FO with **successors** and order to a **cascade product** of a specific deterministic automaton.

Automata for FO over trees

Two past efforts:

1. M. Bojanczyk (2004) proved the equivalence over **finite binary trees** of FO with **successors** and order to a **cascade product** of a specific deterministic automaton.
2. C. Ford (2019) introduced a class of automaton (let's call it FordAuto) translatable into FO with order. Did not prove the other direction.

Our work

Main result:

We obtained two automaton-based characterisations of FO with $\leq$ over infinite trees.

Our work

Main result:

We obtained two automaton-based characterisations of FO with ≤ over infinite trees.

In detail:

1. we investigated two **branching-time temporal logics** already shown equivalent to FO over infinite trees;

Our work

Main result:

We obtained two automaton-based characterisations of FO with ≤ over infinite trees.

In detail:

1. we investigated two **branching-time temporal logics** already shown equivalent to FO over infinite trees;
2. we found **two automaton classes** effectively equivalent to the temporal logics, and consequently to FO;

Our work

Main result:

We obtained two automaton-based characterisations of FO with ≤ over infinite trees.

In detail:

1. we investigated two **branching-time temporal logics** already shown equivalent to FO over infinite trees;
2. we found **two automaton classes** effectively equivalent to the temporal logics, and consequently to FO;
3. from the automaton-based characterisations, we got **normal forms** for the temporal logics and gained insight into **the behaviour of FO over infinite trees**.

Infinite trees

The trees we consider are infinite, unranked and unordered.

Infinite trees

The trees we consider are infinite, unranked and unordered.

- Infinite: every maximal path is infinite (no leaves);

Infinite trees

The trees we consider are infinite, unranked and unordered.

- Infinite: every maximal path is infinite (no leaves);
- Unranked: there is no bound (apart from finiteness) on the number of successors of a node;

Infinite trees

The trees we consider are infinite, unranked and unordered.

- Infinite: every maximal path is infinite (no leaves);
- Unranked: there is no bound (apart from finiteness) on the number of successors of a node;
- Unordered: the order of the successors of a given node is irrelevant.

First Order Logic over infinite trees

There are many **variants** for FO over trees, depending on the predicates one allows in the signature. We chose to use = and <, but not the *i-th* successor predicates.

First Order Logic over infinite trees

There are many **variants** for FO over trees, depending on the predicates one allows in the signature. We chose to use = and <, but not the *i-th* successor predicates.

What FO can say
There are at least three successors of a given node that share the property *P*.

First Order Logic over infinite trees

There are many **variants** for FO over trees, depending on the predicates one allows in the signature. We chose to use = and <, but not the *i-th* successor predicates.

What FO can say	What FO can't say
There are at least three successors of a given node that share the property *P*.	The third, fifth and eighth successor of a given node share the property *P*.

Polarized CTL with Past

The first Branching-time temporal logic we considered is Polarized CTL with past (Schlingloff, 1992) and it is:

Polarized CTL with Past

The first Branching-time temporal logic we considered is Polarized CTL with past (Schlingloff, 1992) and it is:

- **equivalent to FO** over trees;

Polarized CTL with Past

The first Branching-time temporal logic we considered is Polarized CTL with past (Schlingloff, 1992) and it is:

- **equivalent to FO** over trees;
- a fragment of **counting CTL with past.**

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid D^{n}\varphi \mid EX\varphi \mid E\varphi U\varphi \mid \sout{E\varphi R\varphi} \mid Y\varphi \mid \varphi S\varphi$$

Polarized CTL with Past

The first Branching-time temporal logic we considered is Polarized CTL with past (Schlingloff, 1992) and it is:

- **equivalent to FO** over trees;
- a fragment of **counting CTL with past.**

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid D^n\varphi \mid EX\varphi \mid E\varphi U\varphi \mid \sout{E\varphi R\varphi} \mid Y\varphi \mid \varphi S\varphi$$

What Polarized CTL can say
There is a branch on which $(s \vee pUq)Ur$ holds.

Polarized CTL with Past

The first Branching-time temporal logic we considered is Polarized CTL with past (Schlingloff, 1992) and it is:

- **equivalent to FO** over trees;
- a fragment of **counting CTL with past.**

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid D^n\varphi \mid EX\varphi \mid E\varphi U\varphi \mid \cancel{E\varphi R\varphi} \mid Y\varphi \mid \varphi S\varphi$$

What Polarized CTL can say
There is a branch on which $(s \vee pUq)Ur$ holds.

What Polarized CTL can't say
There is a branch where p holds globally.

Connections to modal μ-calculus

Syntactic restriction of Polarized CTL: similar to **Propositional Dynamic Logic**.

Connections to modal μ-calculus

Syntactic restriction of Polarized CTL: similar to **Propositional Dynamic Logic**.

Carreiro & Venema (2014) show that PDL is a fragment of the modal mu-calculus in which **existential modalities** can only be paired with **least-fixpoint operators,** and **universal modalities** can only be paired with **greatest fixpoint operators.**

Connections to modal μ-calculus

Syntactic restriction of Polarized CTL: similar to **Propositional Dynamic Logic**.

Carreiro & Venema (2014) show that PDL is a fragment of the modal mu-calculus in which **existential modalities** can only be paired with **least-fixpoint operators,** and **universal modalities** can only be paired with **greatest fixpoint operators.**

Polarized CTL is a proper fragment of PDL.

Open problem: what is the fragment of μ-calculus coinciding with Polarized CTL?

CTL* over finite paths

The second branching-time temporal logic is **syntactically identical** to the classic counting CTL* but:

CTL* over finite paths

The second branching-time temporal logic is **syntactically identical** to the classic counting CTL* but:

- the usual path quantifiers refer only to non-empty **finite paths;**

CTL* over finite paths

The second branching-time temporal logic is **syntactically identical** to the classic counting CTL* but:

- the usual path quantifiers refer only to non-empty **finite paths;**
- it is **equivalent to FO** over trees, too.

$$\varphi := p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathrm{D}^{n}\varphi \mid \mathrm{E}\psi$$
$$\psi := \varphi \mid \neg\psi \mid \psi \vee \psi \mid \mathrm{X}\psi \mid \psi \mathrm{U} \psi$$

Finite path quantification

The semantic restriction of CTL* over finite paths **trivializes many formulas.**

Finite path quantification

The semantic restriction of CTL* over finite paths **trivializes many formulas.** The simplest examples are:

$$\mathrm{E}\dot{\mathrm{X}}\varphi \leftrightarrow \top \qquad\qquad \mathrm{A}\mathrm{X}\,\varphi \leftrightarrow \bot$$

where $\dot{\mathrm{X}}$ is the **weak version** of the X operator.

Automaton-based characterisations

We proved that two automaton classes are equivalent to Polarized CTL and CTL* over finite paths, and consequently to FO.

The two automaton classes share many similarities, since they are both:

Automaton-based characterisations

We proved that two automaton classes are equivalent to Polarized CTL and CTL* over finite paths, and consequently to FO.

The two automaton classes share many similarities, since they are both:

- Alternating: they mix nondeterminism and universal branching;

Automaton-based characterisations

We proved that two automaton classes are equivalent to Polarized CTL and CTL* over finite paths, and consequently to FO.

The two automaton classes share many similarities, since they are both:

- Alternating: they mix nondeterminism and universal branching;
- Weak: they can switch a finite number of times between accepting and rejecting states;

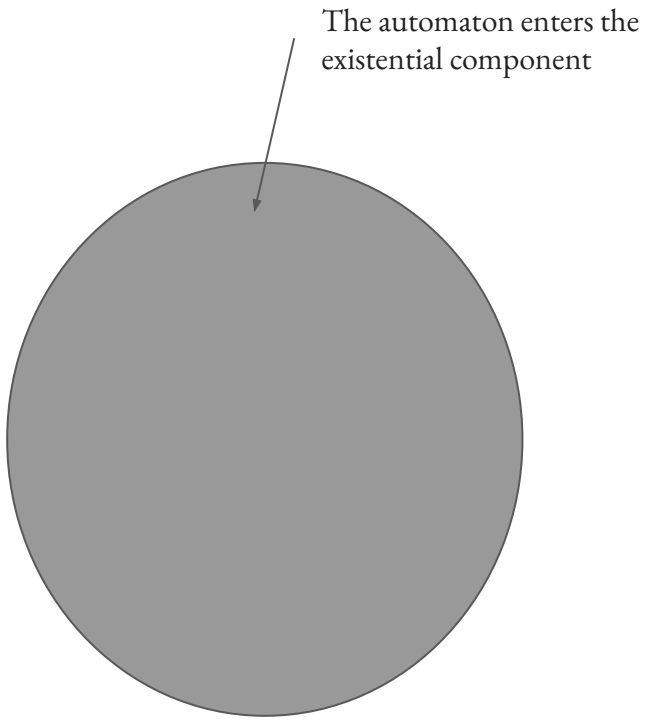
Automaton-based characterisations

We proved that two automaton classes are equivalent to Polarized CTL and CTL* over finite paths, and consequently to FO.

The two automaton classes share many similarities, since they are both:

- Alternating: they mix nondeterminism and universal branching;
- Weak: they can switch a finite number of times between accepting and rejecting states;
- Hesitant: the state-set is partitioned in a way that simulates the difference between existential and universal path quantifiers.

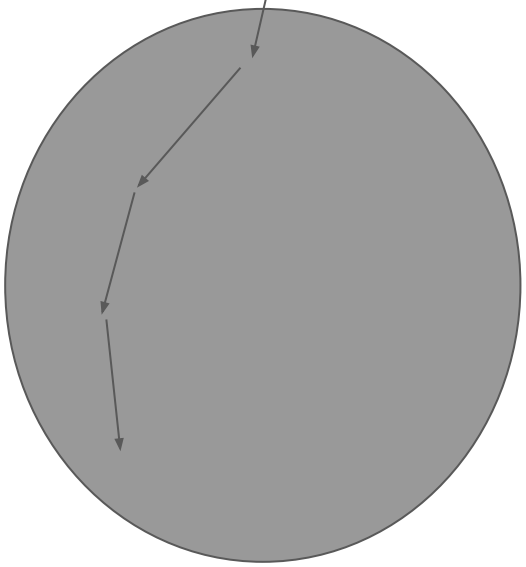
Hesitant restriction: Existential components



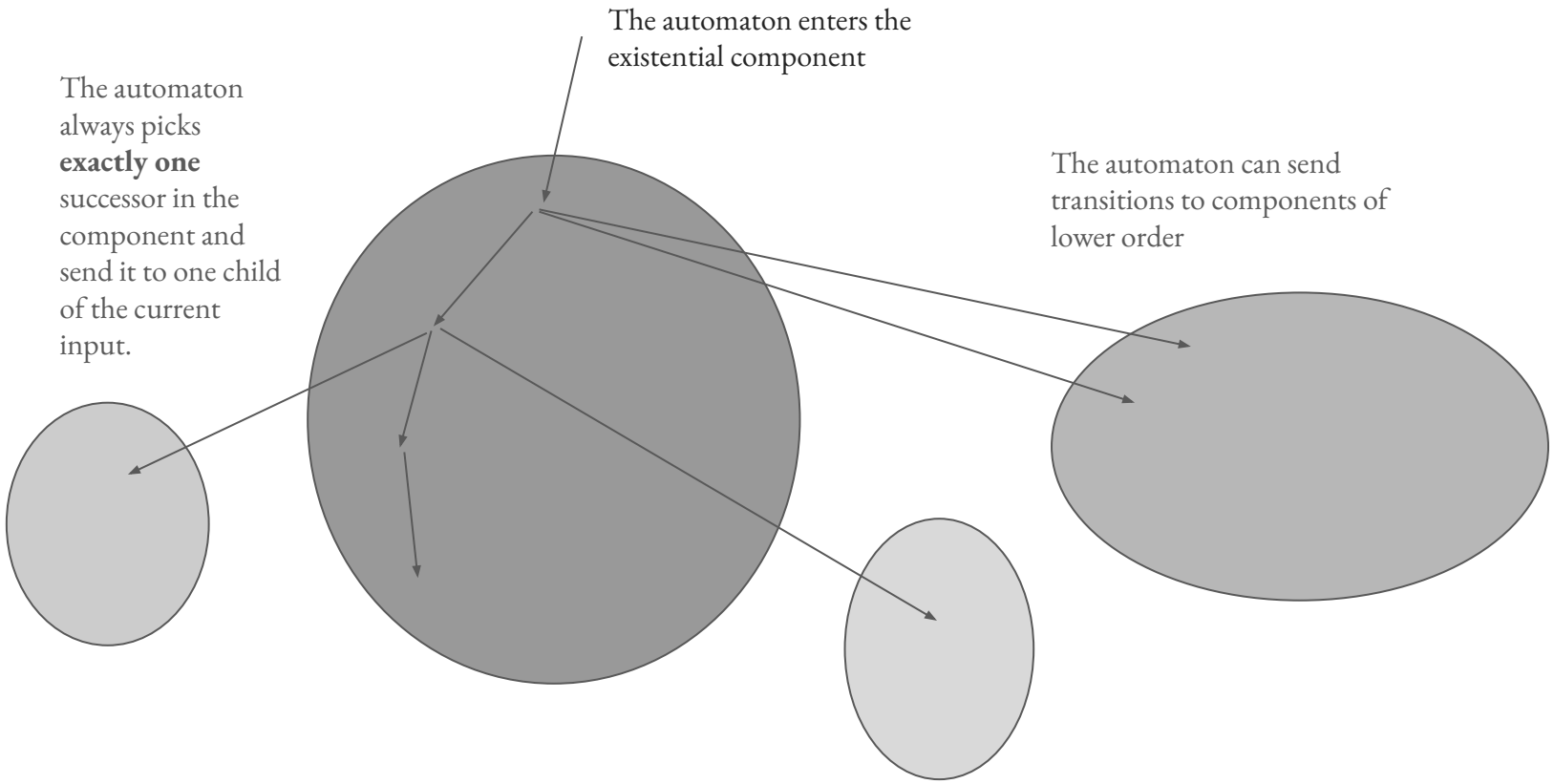
Hesitant restriction: Existential components

The automaton enters the existential component

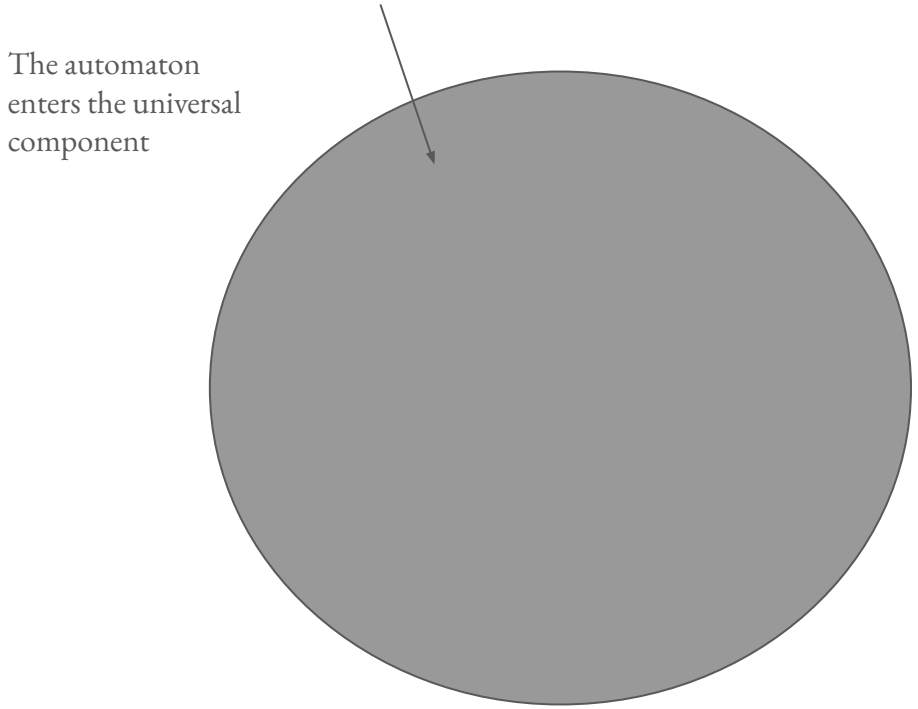
The automaton always picks **exactly one** successor in the component and send it to one child of the current input.



Hesitant restriction: Existential components

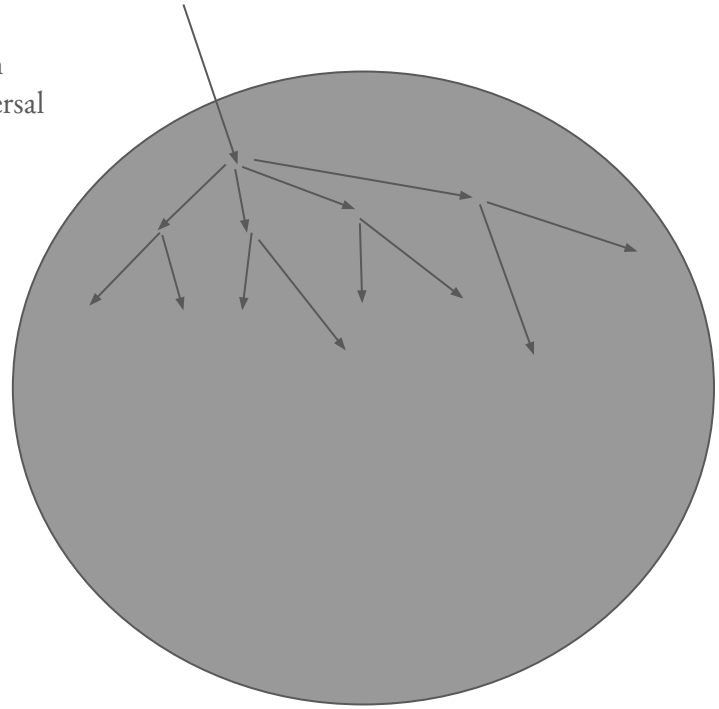


Hesitant restriction: Universal components



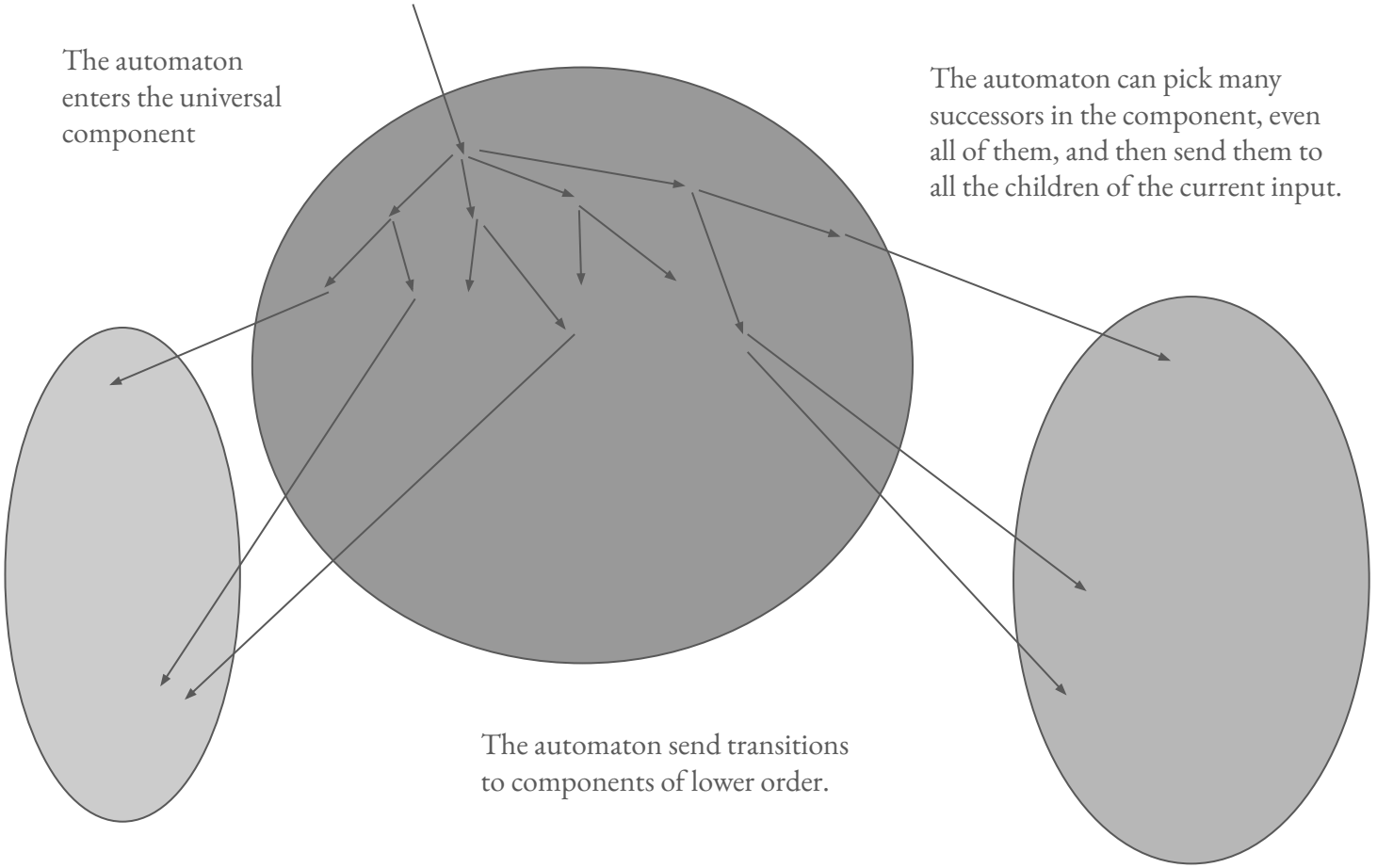
Hesitant restriction: Universal components

The automaton enters the universal component



The automaton can pick many successors in the component, even all of them, and then send them to all the children of the current input.

Hesitant restriction: Universal components



Two-way vs One-way head movements

Most interesting difference: **head movement**.

Two-way vs One-way head movements

Most interesting difference: **head movement**.

The automaton class modeled after Polarized CTL allows **two-way head movements** along the input tree; the other does not.
Why is it **interesting**?

Two-way vs One-way head movements

Most interesting difference: **head movement**.

The automaton class modeled after Polarized CTL allows **two-way head movements** along the input tree; the other does not.
Why is it **interesting**?

Unlike the case of word automata, **two-way head movements increase expressiveness!**

Two-way vs One-way head movements

Most interesting difference: **head movement**.

The automaton class modeled after Polarized CTL allows **two-way head movements** along the input tree; the other does not.
Why is it **interesting**?

Unlike the case of word automata, **two-way head movements increase expressiveness!**

One has to impose different restrictions on the state set to obtain equivalence with FO.

Two-way restriction

If two-way head movements are available, a **very simple restriction** is needed to obtain **FO expressiveness.**

Two-way restriction

If two-way head movements are available, a **very simple restriction** is needed to obtain **FO expressiveness.**

State-set can be partitioned in **linearly ordered singletons.**
Every singleton must be of an “hesitant” type.

Two-way restriction

If two-way head movements are available, a **very simple restriction** is needed to obtain **FO expressiveness.**

State-set can be partitioned in **linearly ordered singletons.**
Every singleton must be of an “hesitant” type.

Similar to alternating automata for **FO over words.**

Conjecture

Call the automaton-class modeled after Polarized CTL: PolarizedAuto. We proved:

Theorem
2-way PolarizedAuto is **strictly more expressive** than 1-way PolarizedAuto.

Conjecture

Call the automaton-class modeled after Polarized CTL: PolarizedAuto. We proved:

Theorem
2-way PolarizedAuto is **strictly more expressive** than 1-way PolarizedAuto.

We conjecture that the FordAuto is **strictly less expressive** than FO, because of the following.

Conjecture
FordAuto is equivalent to 1-way PolarizedAuto.

One-way restriction

To regain FO expressiveness with one-way head movements: more work needed.

One-way restriction

To regain FO expressiveness with one-way head movements: more work needed.

State-set can be partitioned but **no restriction on the size of the components**.

One-way restriction

To regain FO expressiveness with one-way head movements: more work needed.

State-set can be partitioned but **no restriction on the size of the components**.

The first restriction is:

> Every component, seen as a **word automaton**, is **counter-free**.

One-way restriction

To regain FO expressiveness with one-way head movements: more work needed.

State-set can be partitioned but **no restriction on the size of the components**.

The first restriction is:

> Every component, seen as a **word automaton**, is **counter-free**.

Counter-free automata are the automata equivalent to **FO over words**.

One-way restriction

The counter-free restriction is not enough.

One-way restriction

The counter-free restriction is not enough.

Second restriction:

> The components are equivalent to a **SafetyLTL formula** when **universal** and to a **CoSafetyLTL formula** when **existential**.

One-way restriction

The counter-free restriction is not enough.

Second restriction:

> The components are equivalent to a **SafetyLTL formula** when **universal** and to a **CoSafetyLTL formula** when **existential**.

SafetyLTL = Safety fragment of FO over words.

CoSafetyLTL = CoSafety fragment of FO over words.

Results

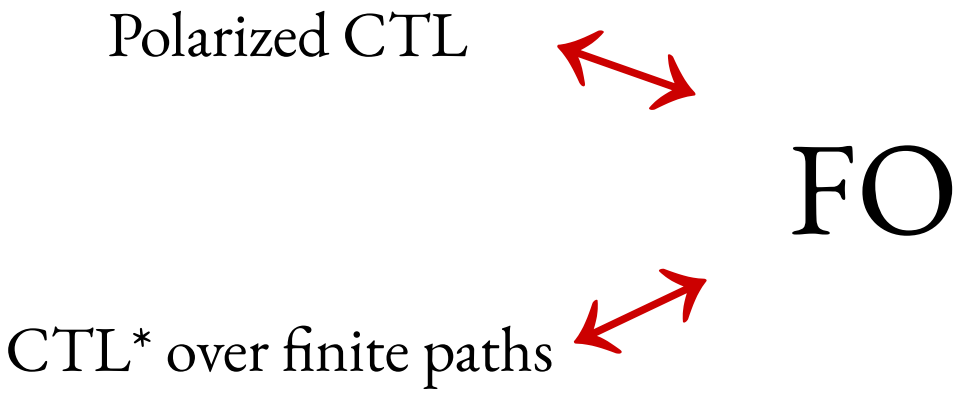
Automaton-class modeled after Polarized CTL: PolarizedAuto.

Automaton-class modeled after CTL* over finite paths: CTL*Auto.

Results

Automaton-class modeled after Polarized CTL: PolarizedAuto.

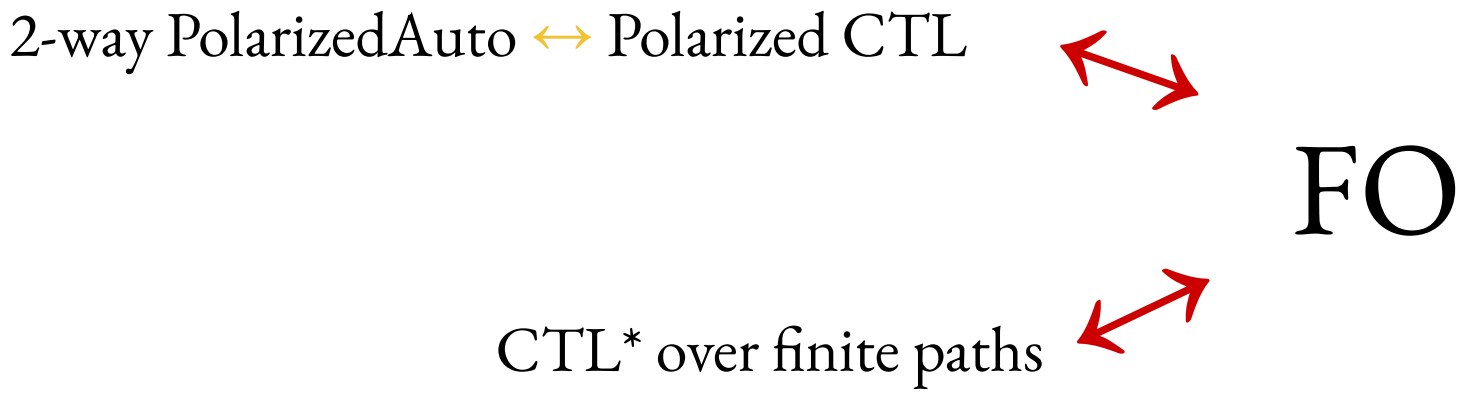
Automaton-class modeled after CTL* over finite paths: CTL*Auto.



Results

Automaton-class modeled after Polarized CTL: PolarizedAuto.

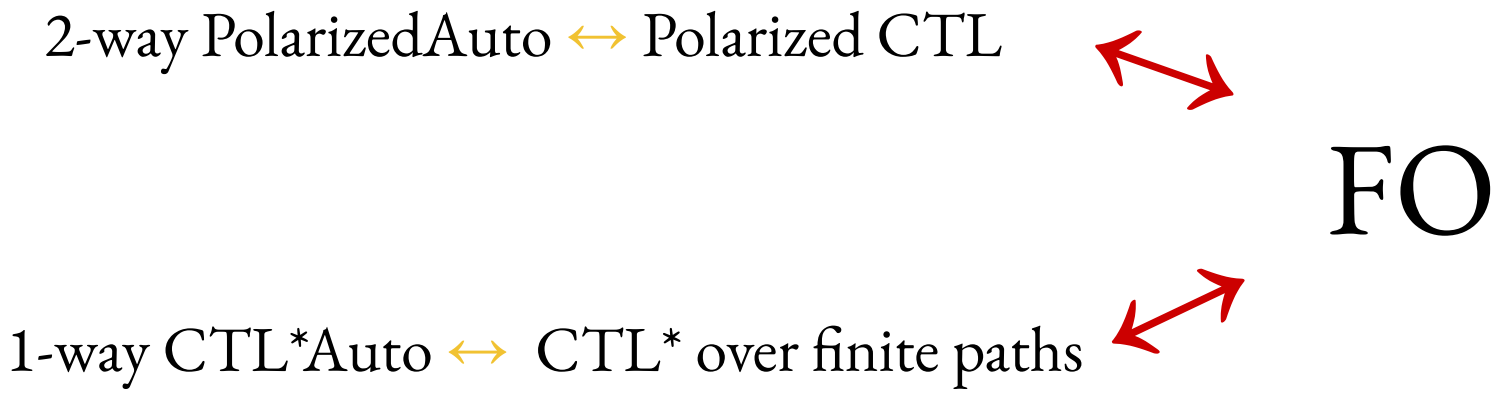
Automaton-class modeled after CTL* over finite paths: CTL*Auto.



Results

Automaton-class modeled after Polarized CTL: PolarizedAuto.

Automaton-class modeled after CTL* over finite paths: CTL*Auto.



Normal forms and insight into FO over infinite trees

The two automaton-based characterisations of FO we obtained offered as a by-product two **normal forms** for Polarized CTL:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \bigvee \varphi \mid \mathrm{D}^n\varphi \mid \mathrm{EF}\psi$$
$$\psi ::= \varphi \mid \psi \bigwedge \psi \mid \psi \bigvee \psi \mid \mathrm{Y}\psi \mid \psi\mathrm{S}\psi$$

and CTL* over finite paths:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \bigvee \varphi \mid \mathrm{D}^n\varphi \mid \mathrm{E}\psi$$
$$\psi ::= \varphi \mid \psi \bigwedge \psi \mid \psi \bigvee \psi \mid \mathrm{X}\psi \mid \psi\mathrm{U}\psi$$

Normal forms and insight into FO over infinite trees

The two automaton-based characterisations of FO we obtained offered as a by-product two **normal forms** for Polarized CTL:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \bigvee \varphi \mid \mathrm{D}^n\varphi \mid \mathrm{EF}\psi$$
$$\psi ::= \varphi \mid \psi \bigwedge \psi \mid \psi \bigvee \psi \mid \mathrm{Y}\psi \mid \psi\mathrm{S}\psi$$

and CTL* over finite paths:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \bigvee \varphi \mid \mathrm{D}^n\varphi \mid \mathrm{E}\psi$$
$$\psi ::= \varphi \mid \psi \bigwedge \psi \mid \psi \bigvee \psi \mid \mathrm{X}\psi \mid \psi\mathrm{U}\psi$$

Take-away messagge
Over infinite trees, when FO predicates existentially over a path, it can only express **co-safety properties**, while when it predicates over all paths, it can only express **safety properties**.

Conclusions

We obtained:

- two **automaton-based characterisations** for FO over infinite trees
- **normal forms** for two temporal logics equivalent to FO
- **insight** into the behaviour of FO over infinite trees

Open problems: Definability problem

What about the **definability problem**?

Open problems: Definability problem

What about the **definability problem**?

Our work does not provide an **immediate advancement**.

However, ...

Open problems: Definability problem

What about the **definability problem**?

Our work does not provide an **immediate advancement**.

However, ...

Over **finite trees**, Bojanczyk (2009) proves that the definability problem for the logic

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathrm{EF}\varphi \mid \mathsf{TS}\varphi$$

is **decidable.**

Open problems: Definability problem

What about the **definability problem**?

Our work does not provide an **immediate advancement**.

However, ...

Over **finite trees**, Bojanczyk (2009) proves that the definability problem for the logic

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathrm{EF}\varphi \mid \mathsf{T}\mathrm{S}\varphi$$

is **decidable.**

Very similar to normal form of Polarized CTL!

Idea: Investigate definability problem of Polarized CTL over **finite** trees.

Open problems: Automata simulation

2-way PolarizedAuto and 1-way CTL*Auto are equivalent by the **equivalence of temporal logics**.

Interesting problem: Find **direct simulation** of each automaton by the other.

Open problems: Full CTL with past expressiveness

Schlingloff (1992): Polarized CTL with past = FO

Open problems: Full CTL with past expressiveness

Schlingloff (1992): Polarized CTL with past = FO

Moller & Rabinovich (2003): CTL* = Monadic Path Logic

Open problems: Full CTL with past expressiveness

Schlingloff (1992): Polarized CTL with past = FO

Full CTL with past = ?

Moller & Rabinovich (2003): CTL* = Monadic Path Logic

Open problems: Full CTL with past expressiveness

Schlingloff (1992): Polarized CTL with past = FO

Full CTL with past = ?

Moller & Rabinovich (2003): CTL* = Monadic Path Logic

Polarized CTL with past < Full CTL with past < CTL*

Thank you
for your attention!