Decidable Fragments of First-Order and Fixed-Point Logic

From prefix-vocabulary classes to guarded logics

Erich Grädel

graedel@rwth-aachen.de.

Aachen University

The classical decision problem

part of Hilbert's formalist programme for the foundations of mathematics

in modern terminology:
find an algorithm that solves the satisfiability problem for first-order logic

D. Hilbert and W. Ackermann:
"Das Entscheidungsproblem ist gelöst, wenn man ein Verfahren kennt, das bei einem vorgelegten logischen Ausdruck durch endlich viele Operationen die Entscheidung über die Allgemeingültigkeit bzw. Erfüllbarkeit erlaubt. (. . .) Das Entscheidungsproblem muss als das Hauptproblem der mathematischen Logik bezeichnet werden."

Grundzüge der theoretischen Logik (1928), pp 73ff

The classical decision problem

part of Hilbert's formalist programme for the foundations of mathematics

in modern terminology:
find an algorithm that solves the satisfiability problem for first-order logic

D. Hilbert and W. Ackermann:
"The decision problem is solved when we know a procedure that allows, for any given logical expression, to decide by finitely many operations its validity or satisfiability. (. . .) The decision problem must be considered the main problem of mathematical logic."

Grundzüge der theoretischen Logik (1928), pp 73ff

The classical decision problem

part of Hilbert's formalist programme for the foundations of mathematics

in modern terminology:
find an algorithm that solves the satisfiability problem for first-order logic

D. Hilbert and W. Ackermann:
"The decision problem is solved when we know a procedure that allows, for any given logical expression, to decide by finitely many operations its validity or satisfiability. (. . .) The decision problem must be considered the main problem of mathematical logic."

Grundzüge der theoretischen Logik (1928), pp 73ff

Similar statements by many other logicians, including **Bernays, Schönfinkel, Herbrand, von Neumann, . . .**

Early results: decision procedures

Decision algorithms for fragments of first-order logic

- the monadic predicate calculus (Löwenheim 1915, Kalmár 1929)
- the prefix class $\exists^*\forall^*$ (Bernays–Schönfinkel 1928, Ramsey 1932)
- the $\exists^*\forall\exists^*$-prefix class (Ackermann 1928)
- the $\exists^*\forall^2\exists^*$-prefix class, without equality
  (Gödel, Kalmár, Schütte 1932-1934)

Early results: decision procedures

Decision algorithms for fragments of first-order logic

- the monadic predicate calculus (Löwenheim 1915, Kalmár 1929)
- the prefix class $\exists^*\forall^*$ (Bernays–Schönfinkel 1928, Ramsey 1932)
- the $\exists^*\forall\exists^*$-prefix class (Ackermann 1928)
- the $\exists^*\forall^2\exists^*$-prefix class, without equality
  (Gödel, Kalmár, Schütte 1932-1934)

These are often called the classical solvable cases of the decision problem.

Early results: reduction classes

A fragment $X \subseteq \mathrm{FO}$ is a reduction class if there is a computable function $f : \mathrm{FO} \to X$ such that

$$\psi \text{ satisfiable} \iff f(\psi) \text{ satisfiable}$$

Early reduction classes:

- relational sentences without $=$ (the pure predicate calculus)
- dyadic sentences (only predicates of arity 2)
- $\forall^*$-sentences, or relational $\forall^*\exists^*$-sentences (Skolem normal form)
- only one binary predicate (Kalmár 1936)
- relational $\forall^3\exists^*$-sentences (Gödel 1933)

The undecidability of first-order logic

Theorem (Church 1936, Turing 1937)

The satisfiability problem for first-order logic is undecidable.

The undecidability of first-order logic

Theorem (Church 1936, Turing 1937)

The satisfiability problem for first-order logic is undecidable.

Theorem (Trakhtenbrot 1950, Craig 1950)

The satisfiability problem for first-order logic on finite structures is undecidable.

The undecidability of first-order logic

Theorem (Church 1936, Turing 1937)

The satisfiability problem for first-order logic is undecidable.

Theorem (Trakhtenbrot 1950, Craig 1950)

The satisfiability problem for first-order logic on finite structures is undecidable.

$\Longrightarrow$ there is no sound and complete proof system for validity on finite structures

The undecidability of first-order logic

Theorem (Church 1936, Turing 1937)

The satisfiability problem for first-order logic is undecidable.

Theorem (Trakhtenbrot 1950, Craig 1950)

The satisfiability problem for first-order logic on finite structures is undecidable.

$\Longrightarrow$ there is no sound and complete proof system for validity on finite structures

After the negative solution, the decision problem did not disappear, but became a classification problem.

The decision problem as a classification problem

- For which classes $X \subseteq$ FO is Sat(X) decidable, which are reduction classes ?
- Similarly for satisfiablity on finite structures

The decision problem as a classification problem

- For which classes $X \subseteq$ FO is Sat(X) decidable, which are reduction classes ?
- Similarly for satisfiablity on finite structures
- Which classes have the finite model property, which contain infinity axioms ?

The decision problem as a classification problem

- For which classes $X \subseteq$ FO is Sat(X) decidable, which are reduction classes ?
- Similarly for satisfiablity on finite structures
- Which classes have the finite model property, which contain infinity axioms ?
- Complexity of decidable cases ?

The decision problem as a classification problem

- For which classes $X \subseteq$ FO is Sat(X) decidable,
  which are reduction classes ?
- Similarly for satisfiablity on finite structures
- Which classes have the finite model property,
  which contain infinity axioms ?
- Complexity of decidable cases ?

What kind of classes $X \subseteq$ FO should be considered?

There are continuum many such classes, one cannot study all of them.

A direction is needed.

The decision problem as a classification problem

- For which classes $X \subseteq$ FO is Sat(X) decidable,
  which are reduction classes ?
- Similarly for satisfiablity on finite structures
- Which classes have the finite model property,
  which contain infinity axioms ?
- Complexity of decidable cases ?

What kind of classes $X \subseteq$ FO should be considered?
There are continuum many such classes, one cannot study all of them.
A direction is needed.

In view of the early results, most attention (in logic) was given to classes determined by quantifier prefix and vocabulary, i.e. the number and arity of relation and function symbols.

Prefix-vocabulary classes

$$[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]_{(=)}$$

Prefix-vocabulary classes

$$[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]_{(=)}$$

- Π is a word over $\{\exists, \forall, \exists^*, \forall^*\}$, describing set of quantifier prefixes

Prefix-vocabulary classes

$$[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]_{(=)}$$

- Π is a word over $\{\exists, \forall, \exists^*, \forall^*\}$, describing set of quantifier prefixes
- $p_m, f_m \leq \omega$ indicate how many relation and function symbols of arity m may occur

Prefix-vocabulary classes

$$[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]_{(=)}$$

- Π is a word over $\{\exists, \forall, \exists^*, \forall^*\}$, describing set of quantifier prefixes
- $p_m, f_m \leq \omega$ indicate how many relation and function symbols of arity m may occur
- presence or absence of $=$ indicates whether the formulae may contain equality

Prefix-vocabulary classes

$$[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]_{(=)}$$

- Π is a word over $\{\exists, \forall, \exists^*, \forall^*\}$, describing set of quantifier prefixes
- $p_m, f_m \leq \omega$ indicate how many relation and function symbols of arity m may occur
- presence or absence of $=$ indicates whether the formulae may contain equality

Example: $[\exists^*\forall\exists^*, (\omega, 1), all]_=$

Prefix-vocabulary classes

$$[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]_{(=)}$$

- Π is a word over $\{\exists, \forall, \exists^*, \forall^*\}$, describing set of quantifier prefixes
- $p_m, f_m \leq \omega$ indicate how many relation and function symbols of arity m may occur
- presence or absence of $=$ indicates whether the formulae may contain equality

Example: $[\exists^*\forall\exists^*, (\omega, 1), all]_=$

sentences $\exists x_1 \ldots \exists x_m \forall y \exists z_1 \ldots \exists z_n \varphi$ where φ is quantifier-free and

– contains at most one binary predicate, and no predicates of arity ≥ 3,
– may contain any number of monadic predicates,
– may contain any number of function symbols of any arity,
– may contain equality.

The complete classification: undecidable cases

A: Pure predicate logic (without functions, without =)

(1)	$[\forall\exists\forall, (\omega, 1), (0)]$	(Kahr 1962)
(2)	$[\forall^3\exists, (\omega, 1), (0)]$	(Surányi 1959)
(3)	$[\forall^*\exists, (0, 1), (0)]$	(Kalmár-Surányi 1950)
(4)	$[\forall\exists\forall^*, (0, 1), (0)]$	(Denton 1963)
(5)	$[\forall\exists\forall\exists^*, (0, 1), (0)]$	(Gurevich 1966)
(6)	$[\forall^3\exists^*, (0, 1), (0)]$	(Kalmár-Surányi 1947)
(7)	$[\forall\exists^*\forall, (0, 1), (0)]$	(Kostyrko-Genenz 1964)
(8)	$[\exists^*\forall\exists\forall, (0, 1), (0)]$	(Surányi 1959)
(9)	$[\exists^*\forall^3\exists, (0, 1), (0)]$	(Surányi 1959)

The complete classification: undecidable cases

B: Classes with functions or equality

(10)	$[\forall, (0), (2)]_=$	(Gurevich 1976)
(11)	$[\forall, (0), (0, 1)]_=$	(Gurevich 1976)
(12)	$[\forall^2, (0, 1), (1)]$	(Gurevich 1969)
(13)	$[\forall^2, (1), (0, 1)]$	(Gurevich 1969)
(14)	$[\forall^2\exists, (\omega, 1), (0)]_=$	(Goldfarb 1984)
(15)	$[\exists^*\forall^2\exists, (0, 1), (0)]_=$	(Goldfarb 1984)
(16)	$[\forall^2\exists^*, (0, 1), (0)]_=$	(Goldfarb 1984)

The complete classification: decidable cases

(Exclude the trivial classes: finite prefix and finite relational vocabulary)

A: Classes with the finite model property

(1)	$[\exists^*\forall^*, all, (0)]_=$	(Bernays, Schönfinkel 1928)
(2)	$[\exists^*\forall^2\exists^*, all, (0)]$	(Gödel 1932, Kalmár 1933, Schütte 1934)
(3)	$[all, (\omega), (\omega)]$	(Löb 1967, Gurevich 1969)
(4)	$[\exists^*\forall\exists^*, all, all]$	(Gurevich 1973)
(5)	$[\exists^*, all, all]_=$	(Gurevich 1976)

B: Classes with infinity axioms

(6)	$[all, (\omega), (1)]_=$	(Rabin 1969)
(7)	$[\exists^*\forall\exists^*, all, (1)]_=$	(Shelah 1977)

Why prefix-vocabulary classes?

- historical reasons

Why prefix-vocabulary classes?

- historical reasons
- simple, syntactically defined classes

Why prefix-vocabulary classes?

- historical reasons
- simple, syntactically defined classes
- a finite classification can a priori be proved to exist.

Why prefix-vocabulary classes?

- historical reasons
- simple, syntactically defined classes
- a finite classification can a priori be proved to exist.

Classifiability Theorem (Gurevich) Let P be a property of prefix vocabulary classes that is closed under subsets and under union:

$$X \in P, \quad Y \subseteq X \quad \Longrightarrow \quad Y \in P$$

$$X, Y \in P \quad \Longrightarrow \quad X \cup Y \in P$$

Then there is a finite list MIN of prefix vocabulary classes such that

$$X \notin P \quad \Longleftrightarrow \quad \exists Y \in \mathrm{MIN} : Y \subseteq X$$

Why prefix-vocabulary classes?

- historical reasons
- simple, syntactically defined classes
- a finite classification can a priori be proved to exist.

Classifiability Theorem (Gurevich) Let P be a property of prefix vocabulary classes that is closed under subsets and under union:

$$X \in P, \quad Y \subseteq X \quad \Longrightarrow \quad Y \in P$$

$$X, Y \in P \quad \Longrightarrow \quad X \cup Y \in P$$

Then there is a finite list MIN of prefix vocabulary classes such that

$$X \notin P \quad \Longleftrightarrow \quad \exists Y \in \mathrm{MIN} : Y \subseteq X$$

Take $P = \{X : \mathrm{Sat}(X) \text{ decidable}\}$.

$\Longrightarrow$ there is a finite list of minimal undecidable prefix-vocabulary classes.

Why prefix-vocabulary classes?

- historical reasons
- simple, syntactically defined classes
- a finite classification can a priori be proved to exist.

Why prefix-vocabulary classes?

- historical reasons
- simple, syntactically defined classes
- a finite classification can a priori be proved to exist.

Good question !

Why prefix-vocabulary classes?

- historical reasons
- simple, syntactically defined classes
- a finite classification can a priori be proved to exist.

Good question !

Modern applications of logic in computer science often lead to quite different classes. For instance:

modal logics, two-variable logics, guarded logics.

On the other side the restriction to first-order logic is often too restrictive. Consider instead (fragments of) fixed point logics

ML: propositional modal logic

Transition systems = Kripke structures = labeled graphs

$$\mathcal{K} = (\ V\ ,\ (E_a)_{a\in A}\ ,\ (P_i)_{i\in I}\ )$$

V: states, elements — $(E_a)_{a\in A}$: actions, binary relations — $(P_i)_{i\in I}$: atomic propositions, unary relations

Syntax of ML: $\psi ::= P_i \mid \neg P_i \mid \psi \wedge \psi \mid \psi \vee \psi \mid \langle a\rangle\psi \mid [a]\psi$

Example: $P_1 \vee \langle a\rangle(P_2 \wedge [b]P_1)$

Semantics: $[\![\psi]\!]^{\mathcal{K}} = \{v : \mathcal{K}, v \models \psi\} = \{v : \psi \text{ holds at state } v \text{ in } \mathcal{K}\}.$

$$\mathcal{K}, v \models \begin{matrix}\langle a\rangle\psi \\ [a]\psi\end{matrix} \quad :\Longleftrightarrow \quad \mathcal{K}, w \models \psi \text{ for } \begin{matrix}\text{some} \\ \text{all}\end{matrix} \ w \text{ with } (v, w) \in E_a$$

ML as a fragment of first-order logic

There is standard translation, taking every $\psi \in \mathrm{ML}$ to a formula $\psi^*(x) \in \mathrm{FO}$ such that

$$\mathcal{K}, v \models \psi \iff \mathcal{K} \models \psi^*(v)$$

$$\begin{aligned} P_i &\longmapsto P_i x \\ \langle a \rangle \psi &\longmapsto \exists y(E_a xy \wedge \psi^*(y)) \\ [a]\psi &\longmapsto \forall y(E_a xy \rightarrow \psi^*(y)) \end{aligned}$$

Properties of modal logic

Good algorithmic properties:

- Sat(ML) is decidable and PSPACE-complete
- efficient model checking: time $O(|\psi| \cdot \|\mathcal{K}\|)$
- automata-based algorithms

Properties of modal logic

Good algorithmic properties:

- Sat(ML) is decidable and PSPACE-complete
- efficient model checking: time $O(|\psi| \cdot \|\mathcal{K}\|)$
- automata-based algorithms

Interesting model-theoretic properties:

- invariance under bisimulation:

$$\mathcal{K}, v \models \psi, \quad \mathcal{K}, v \sim \mathcal{K}', v' \quad \Longrightarrow \quad \mathcal{K}', v' \models \psi$$

in fact ML is the bisimulation-invariant fragment of first-order logic

- ML has the tree model property
- ML has the finite model property
- ...

Limitations of propositional modal logic

ML has very limited expressive power

- no equality
- no unbounded quantification
- no possibility to define new transition relations, not even on the quantifier-free level
- no counting mechanism
- **no recursion mechanism:** reachability problems, game problems, etc. are not expressible

Limitations of propositional modal logic

ML has very limited expressive power

- no equality
- no unbounded quantification
- no possibility to define new transition relations, not even on the quantifier-free level
- no counting mechanism
- **no recursion mechanism:** reachability problems, game problems, etc. are not expressible

Note: The absence of a recursion mechanism is also a limitation of FO

For numerous applications, in particular for reasoning about computation, reachability problems and game problems are essential.

$\Longrightarrow$ **numerous logics that extend ML by recursion mechanisms**

Computation tree logic CTL

ML + path quantification + temporal operators on paths

CTL is a very important logic for hardware verification.

Good balance between expressive power and algorithmic manageability

- $\mathbf{A}\, P\, \mathbf{U}\, Q \quad \equiv \quad (\forall \text{ paths})\ (P \text{ until } Q)$

Computation tree logic CTL

ML + path quantification + temporal operators on paths

CTL is a very important logic for hardware verification.

Good balance between expressive power and algorithmic manageability

- $\mathbf{A}\, P\, \mathbf{U}\, Q \equiv$ ($\forall$ paths) (P until Q)
- $\mathbf{E}\, \mathbf{G}\, P \equiv$ ($\exists$ path) always P
- $\mathbf{A}\, \mathbf{F}\, P \equiv$ ($\forall$ paths) eventually P

Computation tree logic CTL

ML + path quantification + temporal operators on paths

CTL is a very important logic for hardware verification.

Good balance between expressive power and algorithmic manageability

- $\mathbf{A}\,P\,\mathbf{U}\,Q \quad \equiv \quad (\forall \text{ paths})\ (P \text{ until } Q)$
- $\mathbf{E}\,\mathbf{G}\,P \quad \equiv \quad (\exists \text{ path})$ always P
- $\mathbf{A}\,\mathbf{F}\,P \quad \equiv \quad (\forall \text{ paths})$ eventually P
- $\mathbf{A}\,\mathbf{G}\,\mathbf{A}\,\mathbf{F}\,P \quad \equiv \quad (\forall \text{ paths})$ always $(\forall \text{ paths})$ eventually P
  $\equiv \quad (\forall \text{ paths})$ infinitely often P

L_μ: modal μ-calculus

ML + least and greatest fixed points

For any definable monotone relational operator

$$F_\varphi : X \mapsto \{w : \varphi(X) \text{ true at } w\}$$

make also the least and the greatest fixed point of F_φ definable:

$$\mathcal{K}, w \models \mu X.\varphi \iff w \in \bigcap \{X : F_\varphi(X) = X\} \quad \text{(least fixed point)}$$

$$\mathcal{K}, w \models \nu X.\varphi \iff w \in \bigcup \{X : F_\varphi(X) = X\} \quad \text{(greatest fixed point)}$$

L_μ: Examples

- $\mathcal{K}, w \models \nu X \,.\, \langle a \rangle X \iff$ there is an infinite a-path from w in $\mathcal{K}$

  $\mathcal{K}, w \models \mu X \,.\, P \vee [a] X \iff$ every infinite a-path from w eventually hits P

L_μ: Examples

- $\mathcal{K}, w \models \nu X \,.\, \langle a \rangle X \iff$ there is an infinite a-path from w in $\mathcal{K}$

  $\mathcal{K}, w \models \mu X \,.\, P \vee [a]X \iff$ every infinite a-path from w eventually hits P

- Two player game on graph $\mathcal{G} = (V, E)$.

  Players alternate, moves along edges, who cannot move, loses.

Player 0 has winning strategy for game $\mathcal{G}$ from position w

$$\Longleftrightarrow$$

$$\mathcal{G}, w \models \mu X \,.\, \Diamond \Box X$$

L_μ: Examples

- $\mathcal{K}, w \models \nu X \,.\, \langle a \rangle X \iff$ there is an infinite a-path from w in $\mathcal{K}$

  $\mathcal{K}, w \models \mu X \,.\, P \vee [a]X \iff$ every infinite a-path from w eventually hits P

- Two player game on graph $\mathcal{G} = (V, E)$.

  Players alternate, moves along edges, who cannot move, loses.

  Player 0 has winning strategy for game $\mathcal{G}$ from position w

  $\iff$

  $\mathcal{G}, w \models \mu X \,.\, \Diamond \Box X$

- $\mathcal{K}, w \models \nu X \, \mu Y \,.\, \Diamond((P \wedge X) \vee Y) \iff$

  on some path from w, P occurs infinitely often

Importance of CTL and modal μ-calculus

- ML $\subseteq$ CTL $\subseteq$ L_μ
- μ-calculus encompasses most of the popular logics used in hardware verification: LTL, CTL, CTL*, PDL,. . . , and also many logics from other fields: game logic, description logics, etc.
- reasonably good algorithmic properties:
  - satisfiability problem decidable (EXPTIME-complete)
  - efficient model checking for CTL and other important fragments of μ-calculus
  - automata-based algorithms
- nice model-theoretic properties:
  - finite model property
  - tree model property
- L_μ is the bisimulation-invariant fragment of MSO

Summery: properties of modal logics

The basic modal logic ML has good algorithmic and model-theoretic properties, but for most applications in computer science, it is not expressive enough

Summery: properties of modal logics

The basic modal logic ML has good algorithmic and model-theoretic properties, but for most applications in computer science, it is not expressive enough

There are numerous modal logics, that extend ML by recursion mechanisms and other features.

These extensions are practically important because the provide the necessary expressive power for applications, while preserving the good algorithmic properties.

Of particular importance for theoretical studies: modal μ-calculus (modal fixed point logic)

Summery: properties of modal logics

The basic modal logic ML has good algorithmic and model-theoretic properties, but for most applications in computer science, it is not expressive enough

There are numerous modal logics, that extend ML by recursion mechanisms and other features.

These extensions are practically important because the provide the necessary expressive power for applications, while preserving the good algorithmic properties.

Of particular importance for theoretical studies: modal μ-calculus (modal fixed point logic)

Question. (Vardi 96)
Why have modal logics so robust decidability properties ?

Bounded variable logics

FO^k: relational first-order logic with only k variables

Bounded variable logics

FO^k: relational first-order logic with only k variables

we may re-use variables: "there is a path of length 17" $\in \mathrm{FO}^2$

$$\exists x \exists y (Exy \wedge \exists x (Eyx \wedge \exists y (Exy \wedge \ldots \ldots \exists y Exy) \cdots)))$$

Bounded variable logics

FO^k: relational first-order logic with only k variables

we may re-use variables: "there is a path of length 17" $\in \mathrm{FO}^2$

$$\exists x \exists y (Exy \wedge \exists x (Eyx \wedge \exists y (Exy \wedge \ldots \exists y Exy) \cdots)))$$

Note: instead of the number of variables, we can restrict the width

width(ψ): maximal number of free variables in subformulae of ψ

Proposition. $\mathrm{FO}^k \equiv \{\psi \in \mathrm{FO} : \mathrm{width}(\psi) \leq k\}$

Proposition. $\mathrm{Sat}(\mathrm{FO}^k)$ is undecidable for all $k \geq 3$

indeed, even the $\forall\exists\forall$-prefix class is undecidable

Question: What about FO^2 ?

Decidability of FO^2

Theorem (Mortimer 75) FO^2 has the finite model property.

Mortimer's proof implies: every satisfiable $\psi \in FO^2$ has a model with at most $2^{2^{O(|\psi|)}}$ elements.

$\Longrightarrow$ $Sat(FO^2) \in$ 2-NEXPTIME

Decidability of FO^2

Theorem (Mortimer 75) FO^2 has the finite model property.

Mortimer's proof implies: every satisfiable $\psi \in FO^2$ has a model with at most $2^{2^{O(|\psi|)}}$ elements.

$\Longrightarrow$ $Sat(FO^2) \in$ 2-NEXPTIME

Theorem (Lewis 80, Fürer 84) $Sat(FO^2)$ is NEXPTIME-hard.

holds even for sentences of form $\forall x \forall y\, \alpha \wedge \forall x \exists y\, \beta$ (α and β quantifier-free), with only monadic predicates

Decidability of FO^2

Theorem (Mortimer 75) FO^2 has the finite model property.

Mortimer's proof implies: every satisfiable $\psi \in FO^2$ has a model with at most $2^{2^{O(|\psi|)}}$ elements.

$\Longrightarrow$ $Sat(FO^2) \in$ 2-NEXPTIME

Theorem (Lewis 80, Fürer 84) $Sat(FO^2)$ is NEXPTIME-hard.

holds even for sentences of form $\forall x \forall y\, \alpha \wedge \forall x \exists y\, \beta$ (α and β quantifier-free), with only monadic predicates

Theorem (Grädel, Kolaitis, Vardi 97)
Every satisfiable $\psi \in FO^2$ has a model with at most $2^{O(|\psi|)}$ elements.

Corollary $Sat(FO^2)$ is NEXPTIME-complete

Modal logic and two-variable logic

ML is a fragment of FO^2.

The standard translation, taking $\psi \in ML$ to $\psi^*(x) \in FO$ does not need more than two variables:

$$
\begin{array}{rcl}
P_i & \longmapsto & P_i x \\
\langle a \rangle \psi & \longmapsto & \exists y(E_a xy \wedge \psi^*(y)) \\
[a]\psi & \longmapsto & \forall y(E_a xy \rightarrow \psi^*(y))
\end{array}
$$

Modal logic and two-variable logic

ML is a fragment of FO^2.

The standard translation, taking $\psi \in ML$ to $\psi^*(x) \in FO$ does not need more than two variables:

$$
\begin{array}{rcl}
P_i & \longmapsto & P_i x \\
\langle a \rangle \psi & \longmapsto & \exists y(E_a xy \wedge \psi^*(y)) \\
[a]\psi & \longmapsto & \forall y(E_a xy \rightarrow \psi^*(y))
\end{array}
$$

Traditionally the embedding into two-variable logic has been used as an explanation for the good algorithmic properties of modal logics.

Is this explanation convincing ?

Modal logic and two-variable logic

ML is a fragment of FO^2.

The standard translation, taking $\psi \in ML$ to $\psi^*(x) \in FO$ does not need more than two variables:

$$
\begin{array}{rcl}
P_i & \longmapsto & P_i x \\
\langle a \rangle \psi & \longmapsto & \exists y(E_a xy \wedge \psi^*(y)) \\
[a]\psi & \longmapsto & \forall y(E_a xy \rightarrow \psi^*(y))
\end{array}
$$

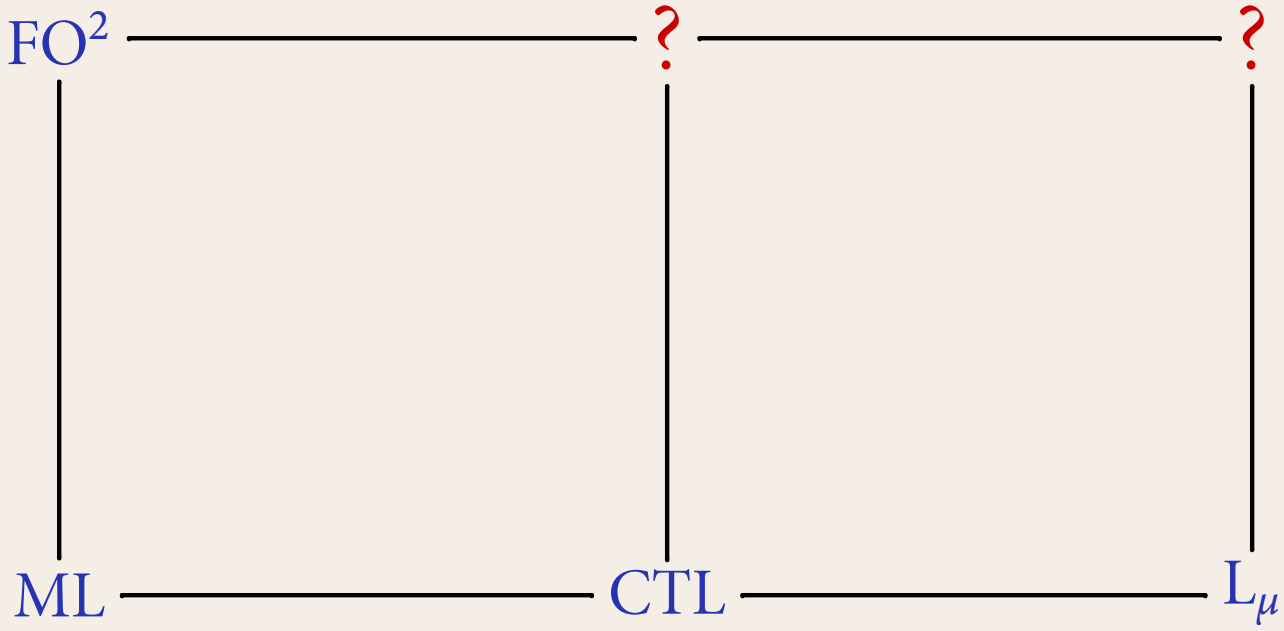
Traditionally the embedding into two-variable logic has been used as an explanation for the good algorithmic properties of modal logics.

Is this explanation convincing ?

Answer: No !!!

Stronger two-variable logics

Challenge: Extend FO^2 in a similar way, as CTL and μ-calculus extend propositional modal logic. Are the resulting logics still decidable ?



Stronger two-variable logics

TC^2 = FO^2 + transitive closures

$(\mathrm{TC}\,\varphi)(x, y)$

CL^2 = least common natural extension of CTL and FO^2

$\mathrm{FO}^2 + \langle\beta(x, y)\rangle^*\varphi + [\beta(x, y)]^*\varphi$

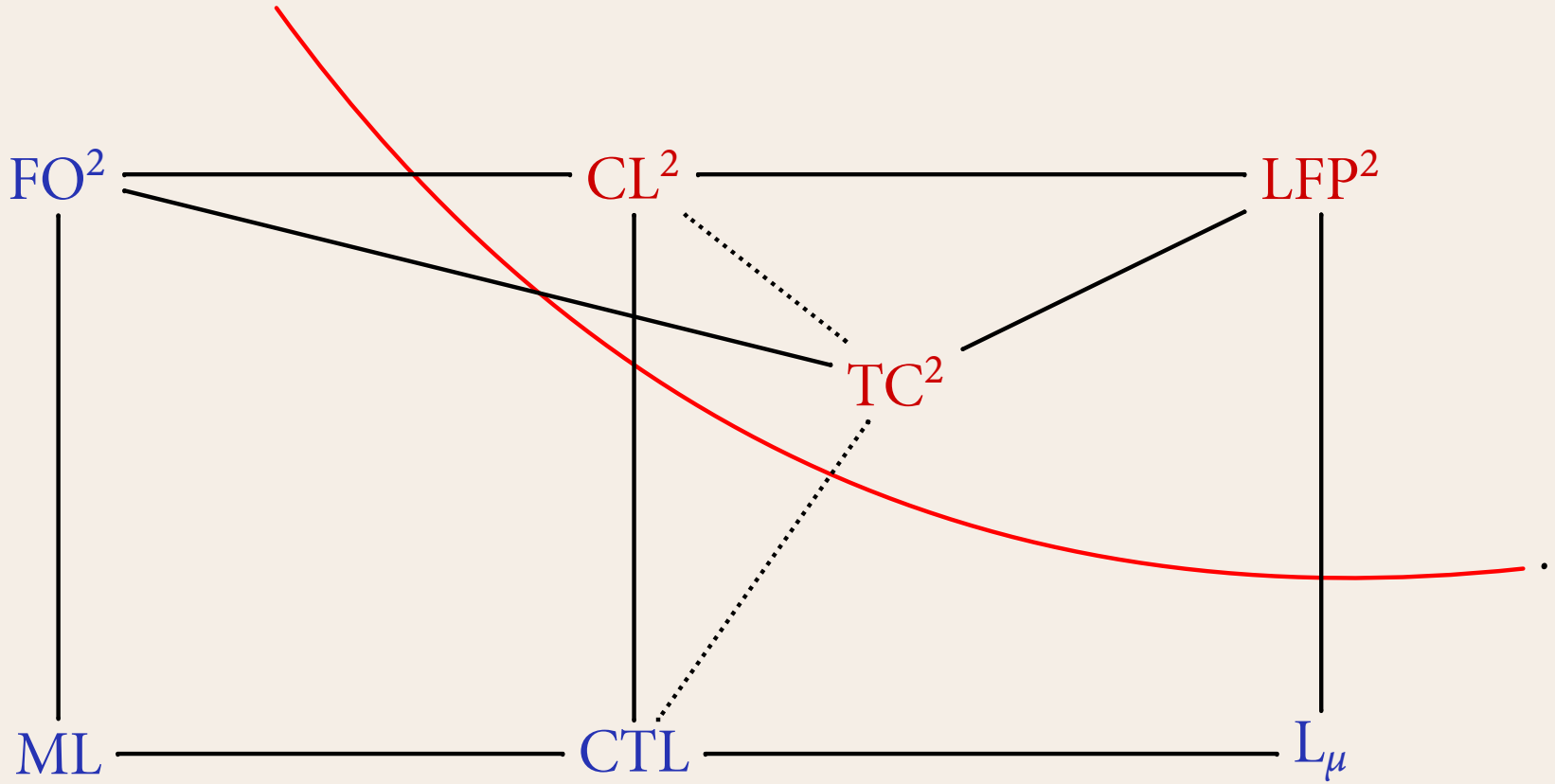
LFP^2 = FO^2 monadic least and greatest fixed points

$[\mathbf{lfp}\, Tx\,.\,\varphi(T, x)](x)$

Theorem. (Grädel, Otto, Rosen)

$\mathrm{Sat}(\mathrm{TC}^2)$, $\mathrm{Sat}(\mathrm{CL}^2)$, and $\mathrm{Sat}(\mathrm{LFP}^2)$ are highly undecidable (Σ^1_1-hard).

Two-variable logics



Two-variable logics do not have the robust decidability properties of modal logics.

Is there another explanation?

The guarded fragment of first-order logic (GF)

Fragment of first-order logic with only guarded quantification

$$\exists \overline{y}(\alpha(\overline{x}, \overline{y}) \wedge \varphi(\overline{x}, \overline{y})) \qquad \forall \overline{y}(\alpha(\overline{x}, \overline{y}) \rightarrow \varphi(\overline{x}, \overline{y}))$$

with guards α : atomic formulae containing all free variables of φ

The guarded fragment of first-order logic (GF)

Fragment of first-order logic with only guarded quantification

$$\exists \bar{y}(\alpha(\bar{x}, \bar{y}) \wedge \varphi(\bar{x}, \bar{y})) \qquad \forall \bar{y}(\alpha(\bar{x}, \bar{y}) \rightarrow \varphi(\bar{x}, \bar{y}))$$

with guards α : atomic formulae containing all free variables of φ

Generalizes modal quantification: ML $\subseteq$ GF $\subseteq$ FO

$$\langle a \rangle \varphi \equiv \exists y(E_a xy \wedge \varphi(y)) \qquad [a]\varphi \equiv \forall y(E_a xy \rightarrow \varphi(y))$$

Model-theoretic and algorithmic properties of GF

- Satisfiability for GF is decidable (Andréka, van Benthem, Németi)
- GF has finite model property (Andréka, Hodkinson, Németi; Grädel)
- GF has (generalized) tree model property:
  every satisfiable formula has model of small tree width (Grädel)
- Sat(GF) is 2Exptime-complete
  and Exptime-complete for formulae of bounded width (Grädel)
- Guarded logics have small model checking games:
  $\|\mathcal{G}(\mathfrak{A}, \psi)\| = O(|\psi| \cdot \|\mathfrak{A}\|)$
  $\Longrightarrow$ efficient game-based model checking algorithms
- GF is invariant under guarded bisimulation

A thesis – and a challenge

Thesis: (Andréka, van Benthem, Németi) The guarded nature of quantification in modal logics is the real reason for their good algorithmic and model-theoretic properties.

A thesis – and a challenge

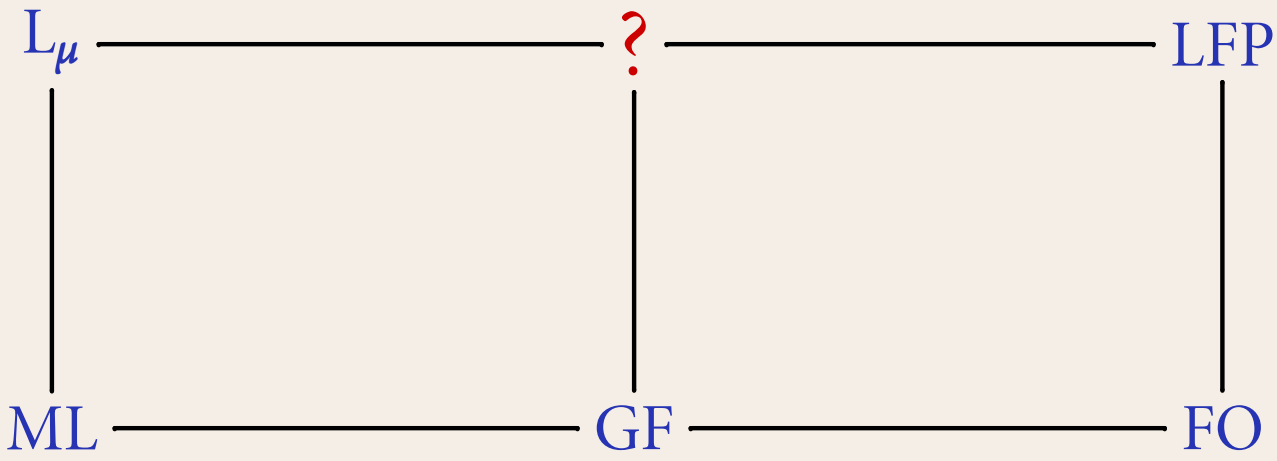
Thesis: (Andréka, van Benthem, Németi) The guarded nature of quantification in modal logics is the real reason for their good algorithmic and model-theoretic properties.

Results on GF provide some evidence for this thesis.

Real test: algorithmic properties of stronger guarded logics than GF.

Challenge: Extend GF in a similar way, as μ-calculus extends ML.

Is the resulting guarded fixed-point logic still decidable ?



A thesis – and a challenge

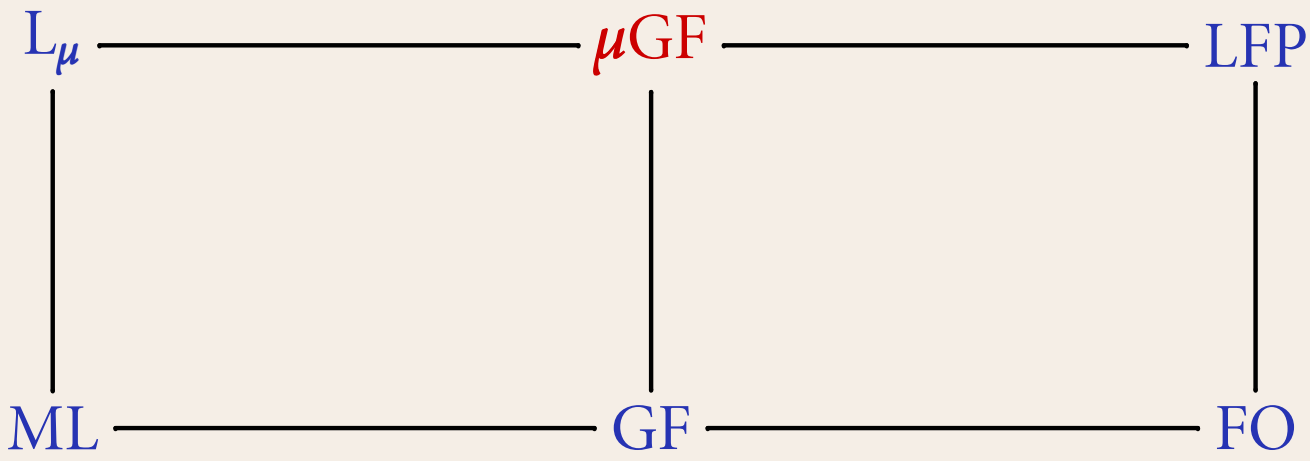
Thesis: (Andréka, van Benthem, Németi) The guarded nature of quantification in modal logics is the real reason for their good algorithmic and model-theoretic properties.

Results on GF provide some evidence for this thesis.

Real test: algorithmic properties of stronger guarded logics than GF.

Challenge: Extend GF in a similar way, as μ-calculus extends ML.

Is the resulting guarded fixed-point logic still decidable ?



Guarded fixed-point logic

μGF = GF + least and greatest fixed points

Guarded fixed-point logic

μGF = GF + least and greatest fixed points

For every formula $\psi(T, x_1 \dots x_k)$, where T is a k-ary relation variable, occuring only positive in ψ,
build formulae $[\mathbf{lfp}\ T\overline{x}\,.\,\psi](\overline{x})$ and $[\mathbf{gfp}\ T\overline{x}\,.\,\psi](\overline{x})$

Guarded fixed-point logic

μGF = GF + least and greatest fixed points

For every formula $\psi(T, x_1 \dots x_k)$, where T is a k-ary relation variable, occuring only positive in ψ,
build formulae $[\mathbf{lfp}\, T\overline{x}\,.\,\psi](\overline{x})$ and $[\mathbf{gfp}\, T\overline{x}\,.\,\psi](\overline{x})$

On every structure $\mathfrak{A}$, $\psi(T, \overline{x})$ defines monotone operator

$$\begin{aligned} \psi^{\mathfrak{A}} : \mathcal{P}(A^k) &\longrightarrow \mathcal{P}(A^k) \\ T &\longmapsto \{\overline{a} : (\mathfrak{A}, T) \models \psi(T, \overline{a})\} \end{aligned}$$

$\psi^{\mathfrak{A}}$ has a least fixed point $\mathbf{lfp}(\psi^{\mathfrak{A}})$ and a greatest fixed point $\mathbf{gfp}(\psi^{\mathfrak{A}})$

$\mathfrak{A} \models [\mathbf{lfp}\, T\overline{x}\,.\,\psi(T, \overline{x})](\overline{a}) \;:\Longleftrightarrow\; \overline{a} \in \mathbf{lfp}(\psi^{\mathfrak{A}})$

$\mathfrak{A} \models [\mathbf{gfp}\, T\overline{x}\,.\,\psi(T, \overline{x})](\overline{a}) \;:\Longleftrightarrow\; \overline{a} \in \mathbf{gfp}(\psi^{\mathfrak{A}})$

Guarded fixed point logic

Example.

$$\psi \quad := \quad \exists xyFxy \wedge (\forall xy\,.\,Fxy)\exists zFyz \wedge$$
$$(\forall xy\,.\,Fxy)\,\underbrace{[\mathbf{lfp}\,Wy\,.\,(\forall x\,.\,Fxy)Wx]}(y)$$
$$\{y : \text{ there is no infinite chain } y \leftarrow y_1 \leftarrow y_2 \leftarrow \cdots\}$$

- ψ is satisfiable: $(\omega, <) \models \psi$
- ψ has no finite models

Proposition. μGF does not have the finite model property.

Decidability of guarded fixed point logic

Theorem. (Grädel, Walukiewicz) Sat(μGF) 2-EXPTIME-complete, and EXPTIME-complete for formulae of bounded width.

same complexity bounds as for GF: no penalty for fixed points !

Proof. Relies on generalized tree model property:
Every satisfiable $\psi \in \mu$GF has a model of small tree width

Reduction of Sat(μGF) to emptiness problem for certain automata (alternating two-way tree automata with parity acceptance condition).

Final remarks

The world of guarded logics is much richer than shown in this talk:

- more liberal guardedness conditions: loosely guarded, clique guarded, action guarded, packed logics
- guarded fragments of other logics, e.g. GSO, Datalog LITE, . . .

Final remarks

The world of guarded logics is much richer than shown in this talk:

- more liberal guardedness conditions: loosely guarded, clique guarded, action guarded, packed logics
- guarded fragments of other logics, e.g. GSO, Datalog LITE, . . .

While modal logics talk only about graphs, guarded logics are applicable to arbitrary relational structures.

Many of the mathematical tools used for analysing modal logics have their counterpart in the world of guarded logics: bisimulation, automata, . . .

Final remarks

The world of guarded logics is much richer than shown in this talk:

- more liberal guardedness conditions: loosely guarded, clique guarded, action guarded, packed logics
- guarded fragments of other logics, e.g. GSO, Datalog LITE, . . .

While modal logics talk only about graphs, guarded logics are applicable to arbitrary relational structures.

Many of the mathematical tools used for analysing modal logics have their counterpart in the world of guarded logics: bisimulation, automata, . . .

Conclusion. Guarded logics provide a general and versatile family of logical systems that generalise, and to a large extent explain the good algorithmic and model-theoretic properties of modal logics.