Positive Logic Is **2-Exptime** Hard

Aleksy Schubert   Paweł Urzyczyn
Daria Walukiewicz-Chrząszcz
University of Warsaw

TYPES, Toulouse, April 26, 2013

Work in Progress...

Positive Logic Is **2-UExptime** Hard

Aleksy Schubert Paweł Urzyczyn
Daria Walukiewicz-Chrząszcz
University of Warsaw

TYPES, Toulouse, April 26, 2013

Positive Logic Is **2-co-NExptime** Hard

Aleksy Schubert Paweł Urzyczyn
Daria Walukiewicz-Chrząszcz
University of Warsaw

TYPES, Toulouse, April 26, 2013

Motivation

Foundational research on:

- Expressive power of “weak” intuitionistic logics.
- High-level properties of proof tactics.

Prenex normal form

In classical logic:

Every formula is classically equivalent to one of the form:

$$Q_1 x_1 Q_2 x_2 \ldots Q_k x_k . \mathit{Body}(x_, x_2, \ldots, x_k),$$

where *Body* has no quantifiers.

Prenex normal form

In classical logic:

Every formula is classically equivalent to one of the form:

$$Q_1 x_1 Q_2 x_2 \ldots Q_k x_k . \, Body(x_, x_2, \ldots, x_k),$$

where *Body* has no quantifiers.

Classification:

Formulas may be classified according to the quantifier prefix, e.g. universal ($\forall^*$) formulas are in Π_1, and Π_2 is $\forall^* \exists^*$.

Prenex normal form

In classical logic:

Every formula is classically equivalent to one of the form:

$$Q_1 x_1 Q_2 x_2 \dots Q_k x_k . \, Body(x_, x_2, \dots, x_k),$$

where *Body* has no quantifiers.

Classification:

Formulas may be classified according to the quantifier prefix, e.g. universal ($\forall^*$) formulas are in Π_1, and Π_2 is $\forall^* \exists^*$.

In intuitionistic logic:

Prenex normal form

In classical logic:

Every formula is classically equivalent to one of the form:

$$Q_1 x_1 Q_2 x_2 \ldots Q_k x_k . \, Body(x_, x_2, \ldots, x_k),$$

where *Body* has no quantifiers.

Classification:

Formulas may be classified according to the quantifier prefix, e.g. universal ($\forall^*$) formulas are in Π_1, and Π_2 is $\forall^* \exists^*$.

In intuitionistic logic:

The prenex fragment is decidable in Pspace.

The language we study

To make things simpler, we consider first-order formulas

- with universal quantifiers and implications;
- without function symbols

This fragment is known to be undecidable.

Mints Hierarchy

Can we restore the prenex classification in intuitionistic logic?

Mints Hierarchy

Can we restore the prenex classification in intuitionistic logic?

Grigori Minc (1968): Yes, consider the quantifier prefix
a formula *would get* if classically normalized.

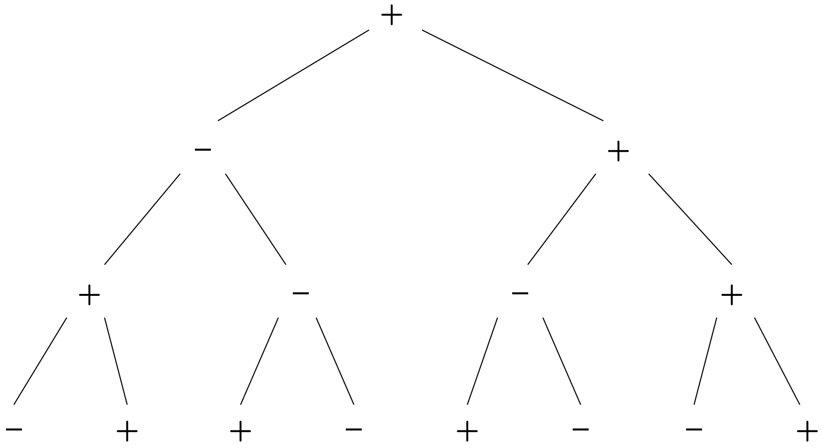
Mints Hierarchy

Can we restore the prenex classification in intuitionistic logic?

Grigori Minc (1968): Yes, consider the quantifier prefix
a formula *would get* if classically normalized.

For example, $\forall$ quantifiers occurring at *positive* positions
will remain $\forall$ in the prefix.

Positive and Negative



Mints Hierarchy

Π_1 – All quantifiers at positive positions.

Σ_1 – All quantifiers at negative positions.

Π_2 – One alternation: some negative quantifiers in scope of some positive ones.

Σ_2 – One alternation: some positive quantifiers in scope of some negative ones.

And so on.

Lower bounds for Mints Hierarchy

Π_1 – **2-UExptime**-hard

Σ_1 – At least **Exptime**-hard

Π_2 – Undecidable

Σ_2 – Undecidable

Work in progress: with function symbols

- Class Σ_1 becomes undecidable.
- Class Π_1 is of the same complexity as before.

A positive example

Let $\varphi = (I \to Tv) \to Ap(x), \quad \psi = I \to (D \to Tv) \to Ap(x)$ and $\vartheta = D \to Ap(x)$, and prove the formula

$$(\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv) \to Tv.$$

A positive example

Let $\varphi = (I \to Tv) \to Ap(x)$, $\psi = I \to (D \to Tv) \to Ap(x)$ and $\vartheta = D \to Ap(x)$, and prove the formula

$$(\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv) \to Tv.$$

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$	Tv

A positive example

Let $\varphi = (I \to Tv) \to Ap(x)$, $\psi = I \to (D \to Tv) \to Ap(x)$ and $\vartheta = D \to Ap(x)$, and prove the formula

$$(\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv) \to Tv.$$

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$	Tv
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$	$\forall x(\varphi \to \psi \to \vartheta \to Ap(x))$

A positive example

Let $\varphi = (I \to Tv) \to Ap(x)$, $\quad \psi = I \to (D \to Tv) \to Ap(x)$ and $\vartheta = D \to Ap(x)$, and prove the formula

$$(\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv) \to Tv.$$

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$	Tv
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$	$\forall x(\varphi \to \psi \to \vartheta \to Ap(x))$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$	$\varphi \to \psi \to \vartheta \to Ap(x)$

A positive example

Let $\varphi = (I \to Tv) \to Ap(x)$, $\quad \psi = I \to (D \to Tv) \to Ap(x)$ and $\vartheta = D \to Ap(x)$, and prove the formula

$$(\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv) \to Tv.$$

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$	Tv
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$	$\forall x(\varphi \to \psi \to \vartheta \to Ap(x))$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$	$\varphi \to \psi \to \vartheta \to Ap(x)$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ φ, ψ, ϑ	$Ap(x)$

Example continued

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x), \quad \vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	$Ap(x)$

Example continued

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x), \quad \vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	$Ap(x)$

Example continued

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x), \quad \vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	$Ap(x)$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x), \quad \vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	$I \to Tv$

Example continued

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x), \quad \vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	$Ap(x)$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x), \quad \vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	$I \to Tv$
$I, \quad \forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x), \quad \vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	Tv

Example continued

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x)$ I, $\vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	Tv

Example continued

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x)$ I, $\vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	Tv
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi(x) = (I \to Tv) \to Ap(x)$ I, $\vartheta(x) = D \to Ap(x)$ $\psi(x) = I \to (D \to Tv) \to Ap(x)$	$\forall x(\varphi \to \psi \to \vartheta \to Ap(x))$

Example continued

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x)$ I, $\vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	Tv
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi(x) = (I \to Tv) \to Ap(x)$ I, $\vartheta(x) = D \to Ap(x)$ $\psi(x) = I \to (D \to Tv) \to Ap(x)$	$\forall x(\varphi \to \psi \to \vartheta \to Ap(x))$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, $\varphi(x), \vartheta(x), \psi(x)$	$\varphi(x') \to \psi(x') \to \vartheta(x') \to Ap(x')$

Example continued

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x)$ I, $\vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	Tv
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi(x) = (I \to Tv) \to Ap(x)$ I, $\vartheta(x) = D \to Ap(x)$ $\psi(x) = I \to (D \to Tv) \to Ap(x)$	$\forall x(\varphi \to \psi \to \vartheta \to Ap(x))$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, $\varphi(x), \vartheta(x), \psi(x)$	$\varphi(x') \to \psi(x') \to \vartheta(x') \to Ap(x')$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, $\varphi(x), \vartheta(x), \psi(x)$ $\varphi(x'), \vartheta(x'), \psi(x')$	$Ap(x')$

Example continued

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi = (I \to Tv) \to Ap(x)$ $I,\quad \vartheta = D \to Ap(x)$ $\psi = I \to (D \to Tv) \to Ap(x)$	Tv
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $\varphi(x) = (I \to Tv) \to Ap(x)$ $I,\quad \vartheta(x) = D \to Ap(x)$ $\psi(x) = I \to (D \to Tv) \to Ap(x)$	$\forall x(\varphi \to \psi \to \vartheta \to Ap(x))$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $I,\quad \varphi(x), \vartheta(x), \psi(x)$	$\varphi(x')\to\psi(x')\to\vartheta(x')\to Ap(x')$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $I,\quad \varphi(x), \vartheta(x), \psi(x)$ $\varphi(x'), \vartheta(x'), \psi(x')$	$Ap(x')$

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ $I, \quad \varphi(x), \ \vartheta(x), \ \psi(x)$ $\varphi(x') = (I \to Tv) \to Ap(x')$ $\vartheta(x') = D \to Ap(x')$ $\psi(x') = I \to (D \to Tv) \to Ap(x')$	$Ap(x')$

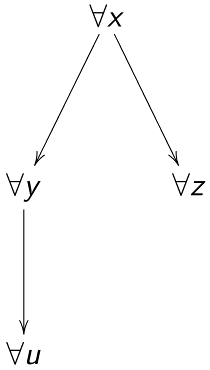
The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x') = (I \to Tv) \to Ap(x')$ $\vartheta(x') = D \to Ap(x')$ $\psi(x') = I \to (D \to Tv) \to Ap(x')$	$Ap(x')$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, D, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x') = (I \to Tv) \to Ap(x')$ $\vartheta(x') = D \to Ap(x')$ $\psi(x') = I \to (D \to Tv) \to Ap(x')$	Tv

The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x') = (I \to Tv) \to Ap(x')$ $\vartheta(x') = D \to Ap(x')$ $\psi(x') = I \to (D \to Tv) \to Ap(x')$	$Ap(x')$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, D, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x') = (I \to Tv) \to Ap(x')$ $\vartheta(x') = D \to Ap(x')$ $\psi(x') = I \to (D \to Tv) \to Ap(x')$	Tv
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, D, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x')$, $\vartheta(x')$, $\psi(x')$	$\forall x(\varphi \to \psi \to \vartheta \to Ap(x))$

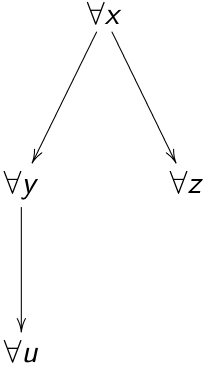
The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x') = (I \to Tv) \to Ap(x')$ $\vartheta(x') = D \to Ap(x')$ $\psi(x') = I \to (D \to Tv) \to Ap(x')$	$Ap(x')$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, D, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x') = (I \to Tv) \to Ap(x')$ $\vartheta(x') = D \to Ap(x')$ $\psi(x') = I \to (D \to Tv) \to Ap(x')$	Tv
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, D, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x')$, $\vartheta(x')$, $\psi(x')$	$\forall x(\varphi \to \psi \to \vartheta \to Ap(x))$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, D, $\varphi(x), \vartheta(x), \psi(x), \varphi(x'),$ $\vartheta(x'), \psi(x'), \varphi(x''), \vartheta(x''), \psi(x'')$	$Ap(x'')$

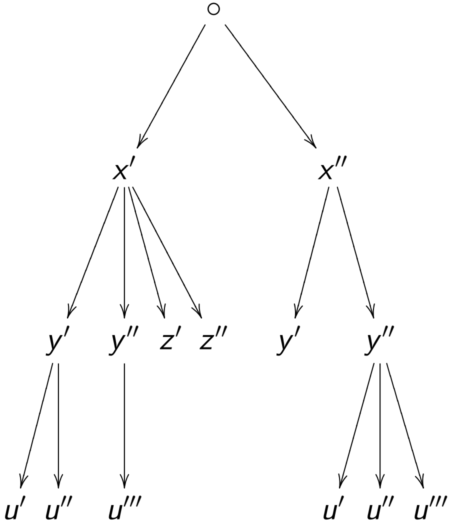
The Assumptions	The Goal
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x') = (I \to Tv) \to Ap(x')$ $\vartheta(x') = D \to Ap(x')$ $\psi(x') = I \to (D \to Tv) \to Ap(x')$	$Ap(x')$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, D, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x') = (I \to Tv) \to Ap(x')$ $\vartheta(x') = D \to Ap(x')$ $\psi(x') = I \to (D \to Tv) \to Ap(x')$	Tv
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, D, $\varphi(x)$, $\vartheta(x)$, $\psi(x)$ $\varphi(x')$, $\vartheta(x')$, $\psi(x')$	$\forall x(\varphi \to \psi \to \vartheta \to Ap(x))$
$\forall x(\varphi \to \psi \to \vartheta \to Ap(x)) \to Tv$ I, D, $\varphi(x), \vartheta(x), \psi(x), \varphi(x'),$ $\vartheta(x'), \psi(x'), \varphi(x''), \vartheta(x''), \psi(x'')$	$Ap(x'')$

Nested quantifiers

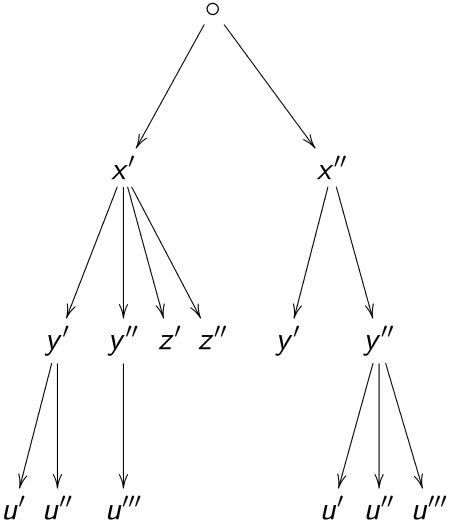


Nested quantifiers

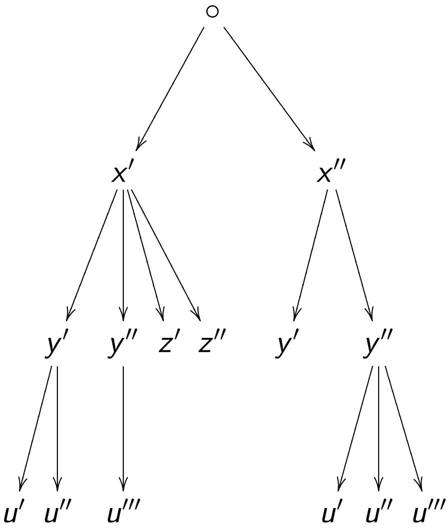




The tree of knowledge



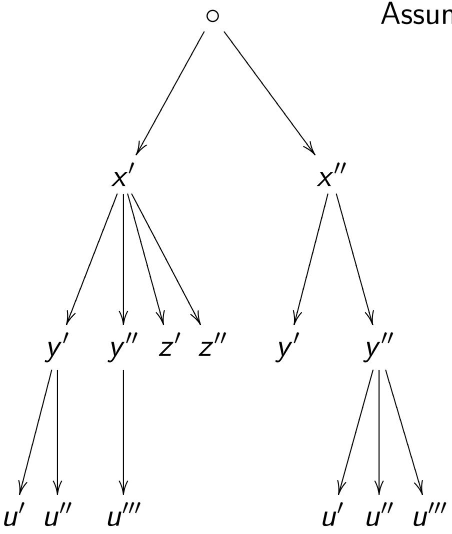
The tree of knowledge



There may be different assumptions about each of the variables.

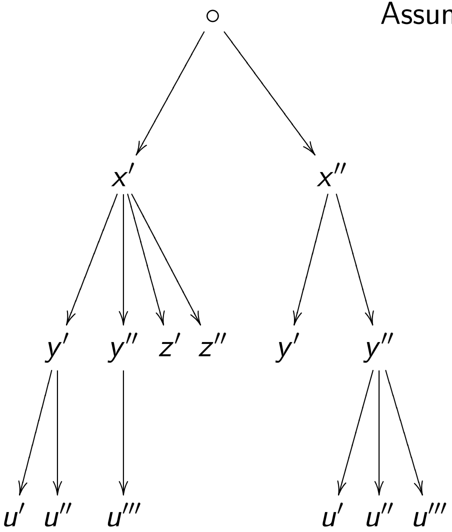
In other words, every node in the tree has a different "knowledge".

How many of them?



Assume n unary predicates.

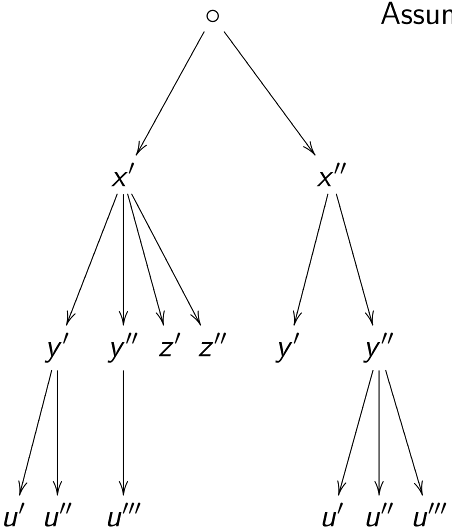
How many of them?



Assume n unary predicates.

$\Longleftarrow$ 2^n possibilities.

How many of them?

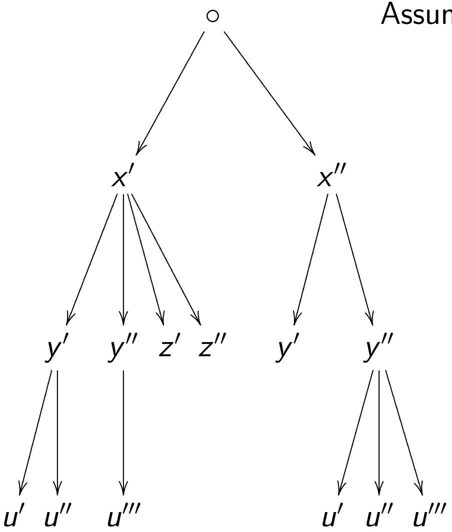


Assume n unary predicates.

$\Longleftarrow$ 2^{2^n} possibilities.

$\Longleftarrow$ 2^n possibilities.

How many of them?



Assume n unary predicates.

$\Longleftarrow$ $2^{2^{2^n}}$ etc. etc.

$\Longleftarrow$ 2^{2^n} possibilities.

$\Longleftarrow$ 2^n possibilities.

Eden automaton

The proof-search procedure can be interpreted as a computation of an automaton.

Eden automaton

The proof-search procedure can be interpreted as
a computation of an automaton.

The automaton operates on the tree of knowledge;
nodes of the tree correspond to the various eigenvariables.

Eden automaton

The proof-search procedure can be interpreted as a computation of an automaton.

The automaton operates on the tree of knowledge; nodes of the tree correspond to the various eigenvariables.

The depth of the tree is bounded, the width is not.

Eden automaton

The proof-search procedure can be interpreted as
a computation of an automaton.

The automaton operates on the tree of knowledge;
nodes of the tree correspond to the various eigenvariables.

The depth of the tree is bounded, the width is not.

The state of the automaton corresponds to the proof goal.

Eden automaton

The proof-search procedure can be interpreted as
a computation of an automaton.

The automaton operates on the tree of knowledge;
nodes of the tree correspond to the various eigenvariables.

The depth of the tree is bounded, the width is not.

The state of the automaton corresponds to the proof goal.

The available assumptions on a variable y'' constitute the
„knowledge" of node y'' of the tree. This can be modeled
by memory registers associated to every node.

Proof search as computation

Assumptions about y'' = data written in registers at node y''

Present goal $P(y'')$ = machine at node y'' in state P.

Proof search as computation

Assumptions about y'' = data written in registers at node y''

Present goal $P(y'')$ = machine at node y'' in state P.

Using assumptions:

$Q(x') \to P(y'')$ = change state from P to Q and move from node y'' to node x'.

Proof search as computation

Assumptions about y'' = data written in registers at node y''

Present goal $P(y'')$ = machine at node y'' in state P.

Using assumptions:

$Q(x') \to P(y'')$ = change state from P to Q and move from node y'' to node x'.

$(R(x') \to Q(x')) \to P(y'')$ = as above; in addition write "1" to register R at node x'.

Proof search as computation

Assumptions about y'' = data written in registers at node y''

Present goal $P(y'')$ = machine at node y'' in state P.

Using assumptions:

$Q(x') \to P(y'')$ = change state from P to Q and move from node y'' to node x'.

$(R(x') \to Q(x')) \to P(y'')$ = as above; in addition write "1" to register R at node x'.

$R(x') \to Q(x') \to P(y'')$ = action possible only if register R at node x' is "1".

Proof search as computation

Assumptions about y'' = data written in registers at node y''

Present goal $P(y'')$ = machine at node y'' in state P.

Using assumptions:

$Q(x') \to P(y'')$ = change state from P to Q and move from node y'' to node x'.

$(R(x') \to Q(x')) \to P(y'')$ = as above; in addition write "1" to register R at node x'.

$R(x') \to Q(x') \to P(y'')$ = action possible only if register R at node x' is "1".

$\forall z(T \to Q_0(z)) \to P(y'')$ = create a new child z' of y''; enter node z' in initial state Q_0.

The tree of knowledge of good and...

Restricted access to data:

In intuitionistic logic one cannot reason from non-existence of assumptions, or delete assumptions.

The tree of knowledge of good and. . .

Restricted access to data:

In intuitionistic logic one cannot reason from non-existence of assumptions, or delete assumptions.

Therefore in an Eden automaton one cannot verify that
a register is "0",

The tree of knowledge of good and...

Restricted access to data:

In intuitionistic logic one cannot reason from non-existence of assumptions, or delete assumptions.

Therefore in an Eden automaton one cannot verify that a register is "0",

One cannot also set a register to "0".

The tree of knowledge of good and. . . no evil

Restricted access to data:

In intuitionistic logic one cannot reason from non-existence of assumptions, or delete assumptions.

Therefore in an Eden automaton one cannot verify that a register is "0",

One cannot also set a register to "0".

In this tree there is only good!

Alternation

Alternation

Existential choice because there may be more than one usable assumption.

Alternation

Existential choice because there may be more than one usable assumption.

Universal choice because an assumption may have more than one premise.

Alternation

Existential choice because there may be more than one usable assumption.

Universal choice because an assumption may have more than one premise.

(To derive $Ap(x)$ from $\varphi \to \psi \to Ap(x)$ one has to prove both φ and ψ.)

Eden automaton (simplified)

An ID is a triple $\langle q, T, w \rangle$, where q is a state, T is a tree of knowledge, and w is the *current apple* (a node of T).

There is a fixed number of binary registers at every node.

Eden automaton (simplified)

An ID is a triple $\langle q, T, w \rangle$, where q is a state, T is a tree of knowledge, and w is the *current apple* (a node of T).

There is a fixed number of binary registers at every node.

Possible actions:

- Move the apple can up to the father of w or down to a nondeterministically chosen child of w;

Eden automaton (simplified)

An ID is a triple $\langle q, T, w \rangle$, where q is a state, T is a tree of knowledge, and w is the *current apple* (a node of T).

There is a fixed number of binary registers at every node.

Possible actions:

- Move the apple can up to the father of w or down to a nondeterministically chosen child of w;
- Raise a selected flag at a given ancestor node of w;

Eden automaton (simplified)

An ID is a triple $\langle q, T, w \rangle$, where q is a state, T is a tree of knowledge, and w is the *current apple* (a node of T).

There is a fixed number of binary registers at every node.

Possible actions:

- Move the apple can up to the father of w or down to a nondeterministically chosen child of w;
- Raise a selected flag at a given ancestor node of w;
- Check if a selected flag is up at a given ancestor of w;

Eden automaton (simplified)

An ID is a triple $\langle q, T, w\rangle$, where q is a state, T is a tree of knowledge, and w is the *current apple* (a node of T).

There is a fixed number of binary registers at every node.

Possible actions:

- Move the apple can up to the father of w or down to a nondeterministically chosen child of w;
- Raise a selected flag at a given ancestor node of w;
- Check if a selected flag is up at a given ancestor of w;
- Create a new child of w and move the apple there.

The main result (1)

The halting problem for Eden Automata is 2-UExptime hard.

The main result (1)

The halting problem for Eden Automata is 2-UExptime hard.

From a universal Turing Machine M working in time $2^{2^{\mathcal{O}(n)}}$ and an input word x of length n

we construct (in Logspace) an Eden automaton A such that

M accepts x iff A terminates.

The main result (2)

Provability of positive formulas is 2-UExptime hard.

The main result (2)

Provability of positive formulas is 2-UExptime hard.

From an Eden automaton A we define (in Logspace) a positive first-order formula Φ such that

A terminates iff Φ is provable.