– LEO-II – A Cooperative Automatic Theorem Prover for Classical Higher-Order Logic[1]

Christoph E. Benzmüller

Microsoft Research, Redmond, October 2, 2008

thanks to: L. Paulson, F. Theiss and A. Fietzke

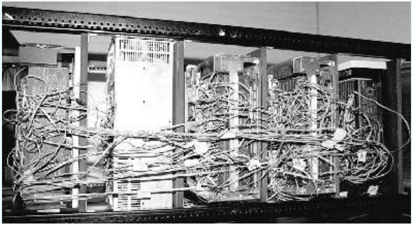
[1] Funded by EPSRC grant EP/D070511/1 at Cambridge University.

Overview

1. Higher-Order Logic (HOL)
    - The Good Thing: Expressivitity
    - The Bad Thing: Automation is a Challenge
2. The LEO-II Prover
    - Motivation and Architecture
    - Solving Lightweight Problems: Sets
    - Less Lightweight Problems: Multimodal Logics
    - More Example Problems: Access Control Logics
3. Conclusion and Outlook

Higher-Order Logic (HOL)

Some people say that HOL is like this:



I don't!

- Semantics (extensionality) [PhD-99,JSL-04]
- Proof theory [IJCAR-06,LMCS-08]
- ATPs LEO and LEO-II [CADE-98,IJCAR-08]

Higher-Order Logic (HOL)
- on one slide -

Property	FOL	HOL	Example
Quantification over			
- individuals	✓	✓	$\forall x. P(F(x))$
- functions	–	✓	$\forall F. P(F(x))$
- predicates/sets/relations	–	✓	$\forall P. P(F(x))$
Unnamed			
- functions	–	✓	$(\lambda x. x)$
- predicates/sets/relations	–	✓	$(\lambda x. x \neq 2)$
Statements about			
- functions	–	✓	$\mathit{continuous}(\lambda x. x)$
- predicates/sets/relations	–	✓	$\mathit{reflexive}(=)$

Sets and Relations in HOL

$$A \cup B := \{x \mid x \in A \vee x \in B\}$$

$$A \cup B := (\lambda x. x \in A \vee x \in B)$$

$$\cup := \lambda A. \lambda B. (\lambda x. x \in A \vee x \in B)$$

$$\mathtt{symmetric} := \lambda F. (\forall x, y. F(x, y) = F(y, x))$$

$$\textit{Theorem}: \quad \mathtt{symmetric}(\cup)$$

Sets and Relations in HOL

$$A \cup B := \{x \mid x \in A \vee x \in B\}$$

$$A \cup B := (\lambda x. x \in A \vee x \in B)$$

$$\cup := \lambda A. \lambda B. (\lambda x. x \in A \vee x \in B)$$

$$\mathtt{symmetric} := \lambda F. (\forall x, y. F(x, y) = F(y, x))$$

$$\mathit{Theorem}: \quad \mathtt{symmetric}(\cup)$$

Sets and Relations in HOL

$$A \cup B := \{x \mid x \in A \vee x \in B\}$$

$$A \cup B := (\lambda x.\, x \in A \vee x \in B)$$

$$\cup := \lambda A.\, \lambda B.\, (\lambda x.\, x \in A \vee x \in B)$$

$$\mathtt{symmetric} := \lambda F.\, (\forall x, y.\, F(x, y) = F(y, x))$$

$$\mathit{Theorem}: \quad \mathtt{symmetric}(\cup)$$

Sets and Relations in HOL

$$
\begin{aligned}
A \cup B &:= \{x \mid x \in A \vee x \in B\} \\
A \cup B &:= (\lambda x. x \in A \vee x \in B) \\
\cup &:= \lambda A. \lambda B. (\lambda x. x \in A \vee x \in B)
\end{aligned}
$$

$$\texttt{symmetric} := \lambda F. (\forall x, y. F(x, y) = F(y, x))$$

$$\textit{Theorem}: \quad \texttt{symmetric}(\cup)$$

Sets and Relations in HOL

$$
\begin{aligned}
A \cup B &:= \{x \mid x \in A \vee x \in B\} \\
A \cup B &:= (\lambda x_{\blacksquare} x \in A \vee x \in B) \\
\cup &:= \lambda A_{\blacksquare} \lambda B_{\blacksquare} (\lambda x_{\blacksquare} x \in A \vee x \in B)
\end{aligned}
$$

$$
\texttt{symmetric} := \lambda F_{\blacksquare} (\forall x, y_{\blacksquare} F(x, y) = F(y, x))
$$

$$
\textit{Theorem}: \quad \texttt{symmetric}(\cup)
$$

Sets and Relations in HOL

$$
\begin{array}{lcl}
\in & := & \lambda x. \lambda A. A(x) \\
\emptyset & := & \lambda x. \bot \\
\cap & := & \lambda A. \lambda B. (\lambda x. x \in A \land x \in B) \\
\cup & := & \lambda A. \lambda B. (\lambda x. x \in A \lor x \in B) \\
\setminus & := & \lambda A. \lambda B. (\lambda x. x \in A \land x \notin B) \\
\ldots & & \\
\subseteq & := & \lambda A. \lambda B. (\forall x. x \in A \Rightarrow x \in B) \\
\mathcal{P} & := & \lambda A. (\lambda B. B \subseteq A) \\
\ldots & & \\
\texttt{reflexive} & := & \lambda R. (\forall x. R(x, x)) \\
\texttt{transitive} & := & \lambda R. (\forall x, y, z. (R(x, y) \land R(y, z)) \Rightarrow R(x, z)) \\
\ldots & &
\end{array}
$$

Types are Needed

Typed Sets and Relations in HOL

$$
\begin{array}{lll}
\in & := & \lambda x_\alpha . \lambda A_{\alpha\to o} . A(x) \\
\emptyset & := & \lambda x_\alpha . \bot \\
\cap & := & \lambda A_{\alpha\to o} . \lambda B_{\alpha\to o} . (\lambda x_\alpha . x \in A \land x \in B) \\
\cup & := & \lambda A_{\alpha\to o} . \lambda B_{\alpha\to o} . (\lambda x_\alpha . x \in A \lor x \in B) \\
\setminus & := & \lambda A_{\alpha\to o} . \lambda B_{\alpha\to o} . (\lambda x_\alpha . x \in A \land x \notin B) \\
\ldots & &
\end{array}
$$

Polymorphism is a Challenge for Automation

- Another source of indeterminism / blind guessing

[TPHOLs-WP-07]

Automation of HOL: A Nightmare?

Undecidable and Infinitary Unification

$\exists F_{\iota \to \iota} . F(g(x)) = g(F(x))$

$\cancel{(1)}$ $F \leftarrow \lambda y_i . y$
$\cancel{(2)}$ $F \leftarrow \lambda y_i . g(y)$
$\cancel{(3)}$ $F \leftarrow \lambda y_i . g(g(y))$
(4) $\ldots$

Automation of HOL: A Nightmare?

Primitive Substitution

Example Theorem: $\exists S_{\blacksquare}\, \mathit{reflexive}(S)$

Negation and Expansion of Definitions:

$$\neg \exists S_{\blacksquare} (\forall x_{\iota \blacksquare}\, S(x, x))$$

Clause Normalisation ($a(S)$ Skolem term):

$$\neg S(a(S), a(S))$$

Guess some suitable instances for S

$S \leftarrow \lambda y_{\blacksquare} \lambda z_{\blacksquare} \top$

$$\rightsquigarrow \neg \top$$

$S \leftarrow \lambda y_{\blacksquare} \lambda z_{\blacksquare} V(y, z) = W(y, z)$

$$\rightsquigarrow V(a(S), a(S)) \neq W(a(S), a(S))$$

$S \leftarrow \ldots$

Automation of HOL: A Nightmare?

Cut rule

$$\frac{A \Rightarrow \mathbf{C} \quad \mathbf{C} \Rightarrow B}{A \Rightarrow B}$$

considered as bad in ATP

Calculi that avoid axioms

- Axiom of excluded middle ✓
- Comprehension axioms ✓
- Functional and Boolean extensionality ✓[CADE-98,PhD-99]
- Leibniz and other definitions of equality ✓[CADE-99,PhD-99]
- Axiom of induction ?
- Axiom of choice –
- Axiom of description –

Automation of HOL: A Nightmare?

Cut rule

$$\frac{A \Rightarrow \mathbf{C} \quad \mathbf{C} \Rightarrow B}{A \Rightarrow B}$$

considered as bad in ATP

[LMCS-08]: Axioms that imply Cut   Calculi that avoid axioms

- Axiom of excluded middle ✓
- Comprehension axioms ✓
- Functional and Boolean extensionality ✓[CADE-98,PhD-99]
- Leibniz and other definitions of equality ✓[CADE-99,PhD-99]
- Axiom of induction ?
- Axiom of choice –
- Axiom of description –

Automation of HOL: A Nightmare?

Cut rule

$$\frac{A \Rightarrow \mathbf{C} \quad \mathbf{C} \Rightarrow B}{A \Rightarrow B}$$

considered as bad in ATP

Calculi that avoid axioms

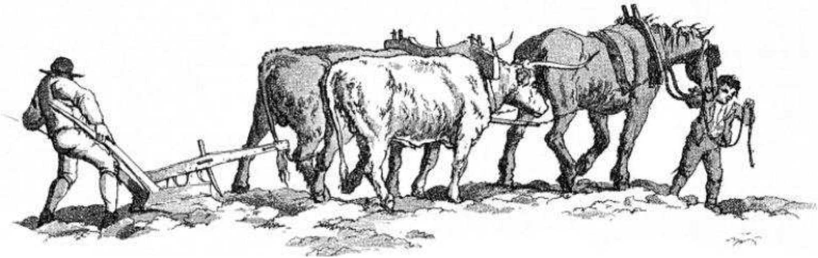
- Axiom of excluded middle ✓
- Comprehension axioms ✓
- Functional and Boolean extensionality ✓[CADE-98,PhD-99]
- Leibniz and other definitions of equality ✓[CADE-99,PhD-99]
- Axiom of induction ?
- Axiom of choice –
- Axiom of description –

LEO-II

An Effective Higher-Order Theorem Prover

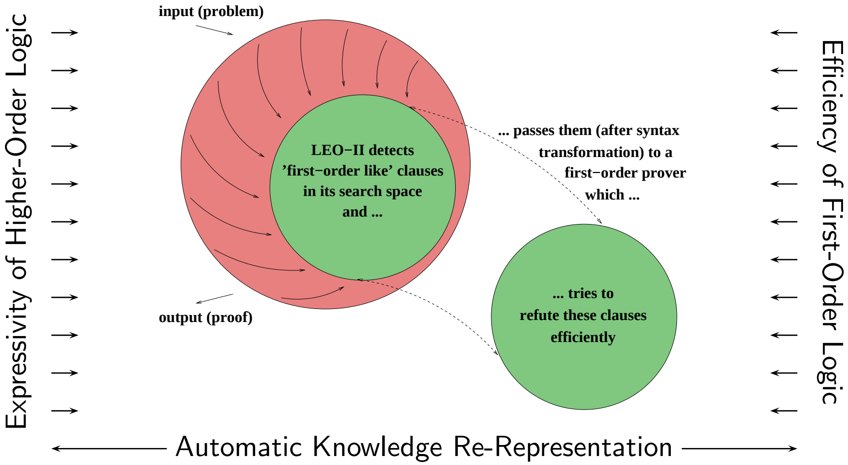
University of Cambridge

Universität des Saarlandes



LEO-II employs FO-ATPs: E, Spass, Vampire

Architecture of LEO-II



Solving Lightweight Problems

Example: TPTP Problem SET171+3

Axiomatization in FO Set Theory

Assumptions:

$\forall B, C, x. (x \in (B \cup C) \Leftrightarrow x \in B \vee x \in C)$

$\forall B, C, x. (x \in (B \cap C) \Leftrightarrow x \in B \wedge x \in C)$

$\forall B, C. (B \subseteq C \Leftrightarrow \forall x. x \in B \Rightarrow x \in C)$

$\forall B, C. (B \cup C = C \cup B)$

$\forall B, C. (B \cap C = C \cap B)$

$\forall B, C. (B = C \Leftrightarrow B \subseteq C \wedge C \subseteq B)$

$\forall B, C. (B = C \Leftrightarrow \forall x. x \in B \Leftrightarrow x \in C)$

Proof Goal:

$$\forall B, C, D. \quad B \cup (C \cap D) = (B \cup C) \cap (B \cup D)$$

Performance: FO-ATPs

```
% SPASS---3.0
% Problem : SET171+3
% SPASS beiseite: Ran out of time.

% E---0.999
% Problem : SET171+3
% Failure: Resource limit exceeded
(time)

% Vampire---9.0
% Problem : SET171+3
% Result : Theorem 68.6s
```

Performance: LEO-II + E

```
Eureka --- Thanks to Corina!
Total Reasoning Time: 0.03s
LEO-II (Proof Found!)
```

Example: TPTP Problem SET171+3

Axiomatization in FO Set Theory

Assumptions:

$\forall B, C, x_\blacksquare (x \in (B \cup C) \Leftrightarrow x \in B \vee x \in C)$

$\forall B, C, x_\blacksquare (x \in (B \cap C) \Leftrightarrow x \in B \wedge x \in C)$

$\forall B, C_\blacksquare (B \subseteq C \Leftrightarrow \forall x_\blacksquare x \in B \Rightarrow x \in C)$

$\forall B, C_\blacksquare (B \cup C = C \cup B)$

$\forall B, C_\blacksquare (B \cap C = C \cap B)$

$\forall B, C_\blacksquare (B = C \Leftrightarrow B \subseteq C \wedge C \subseteq B)$

$\forall B, C_\blacksquare (B = C \Leftrightarrow \forall x_\blacksquare x \in B \Leftrightarrow x \in C)$

Proof Goal:

$\forall B, C, D_\blacksquare$

$$B \cup (C \cap D) = (B \cup C) \cap (B \cup D)$$

Performance: FO-ATPs

```
% SPASS---3.0
% Problem : SET171+3
% SPASS beiseite: Ran out of time.

% E---0.999
% Problem : SET171+3
% Failure: Resource limit exceeded
(time)

% Vampire---9.0
% Problem : SET171+3
% Result : Theorem 68.6s
```

Performance: LEO-II + E

```
Eureka --- Thanks to Corina!
Total Reasoning Time: 0.03s
LEO-II (Proof Found!)
```

Example: TPTP Problem SET171+3

Axiomatization in FO Set Theory

Assumptions:

$\forall B, C, x. (x \in (B \cup C) \Leftrightarrow x \in B \vee x \in C)$

$\forall B, C, x. (x \in (B \cap C) \Leftrightarrow x \in B \wedge x \in C)$

$\forall B, C. (B \subseteq C \Leftrightarrow \forall x. x \in B \Rightarrow x \in C)$

$\forall B, C. (B \cup C = C \cup B)$

$\forall B, C. (B \cap C = C \cap B)$

$\forall B, C. (B = C \Leftrightarrow B \subseteq C \wedge C \subseteq B)$

$\forall B, C. (B = C \Leftrightarrow \forall x. x \in B \Leftrightarrow x \in C)$

Proof Goal:

$$\forall B, C, D. \quad B \cup (C \cap D) = (B \cup C) \cap (B \cup D)$$

Performance: FO-ATPs

```
% SPASS---3.0
% Problem : SET171+3
% SPASS beiseite: Ran out of time.

% E---0.999
% Problem : SET171+3
% Failure: Resource limit exceeded
(time)

% Vampire---9.0
% Problem : SET171+3
% Result : Theorem 68.6s
```

Performance: LEO-II + E

```
Eureka --- Thanks to Corina!
Total Reasoning Time: 0.03s
LEO-II (Proof Found!)
```

Example: TPTP Problem SET171+3

Axiomatization in FO Set Theory

Assumptions:

$\forall B, C, x.(x \in (B \cup C) \Leftrightarrow x \in B \vee x \in C)$

$\forall B, C, x.(x \in (B \cap C) \Leftrightarrow x \in B \wedge x \in C)$

$\forall B, C.(B \subseteq C \Leftrightarrow \forall x. x \in B \Rightarrow x \in C)$

$\forall B, C.(B \cup C = C \cup B)$

$\forall B, C.(B \cap C = C \cap B)$

$\forall B, C.(B = C \Leftrightarrow B \subseteq C \wedge C \subseteq B)$

$\forall B, C.(B = C \Leftrightarrow \forall x. x \in B \Leftrightarrow x \in C)$

Proof Goal:

$$\forall B, C, D. \quad B \cup (C \cap D) = (B \cup C) \cap (B \cup D)$$

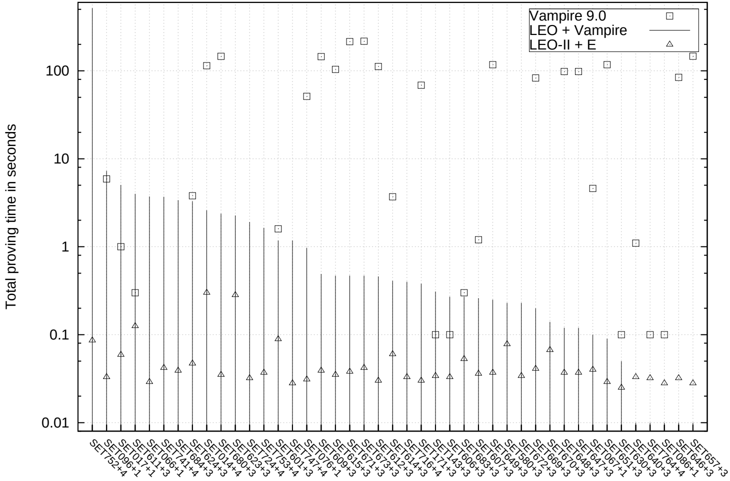
Performance: FO-ATPs

```
% SPASS---3.0
% Problem : SET171+3
% SPASS beiseite: Ran out of time.

% E---0.999
% Problem : SET171+3
% Failure: Resource limit exceeded
(time)

% Vampire---9.0
% Problem : SET171+3
% Result : Theorem 68.6s
```

Performance: LEO-II + E

```
Eureka --- Thanks to Corina!
Total Reasoning Time: 0.03s
LEO-II (Proof Found!)
```

Results



Vampire 9.0
LEO + Vampire
LEO-II + E
Total proving time in seconds
100
10
1
0.1
0.01
SET752+4
SET096+1
SET017+1
SET611+3
SET066+1
SET741+4
SET684+3
SET624+3
SET014+4
SET680+3
SET623+3
SET724+4
SET753+4
SET601+3
SET747+4
SET076+1
SET609+3
SET615+3
SET671+3
SET673+3
SET612+3
SET614+3
SET716+4
SET171+3
SET143+3
SET606+3
SET683+3
SET607+3
SET649+3
SET580+3
SET672+3
SET669+3
SET670+3
SET648+3
SET647+3
SET067+1
SET651+3
SET630+3
SET640+3
SET764+4
SET086+1
SET646+3
SET657+3

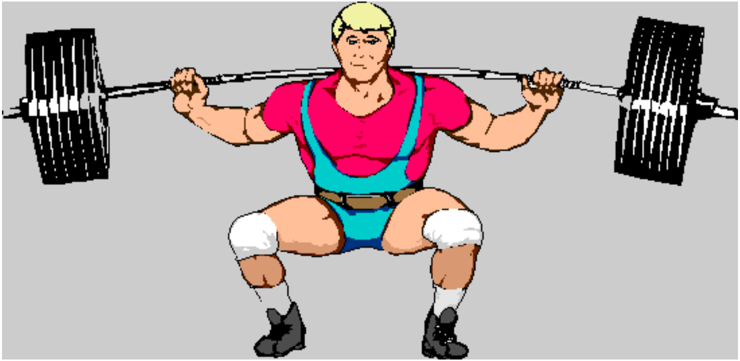
Results

Problem	Vamp. 9.0	LEO+Vamp.	LEO-II+E
014+4	114.5	2.60	0.300
017+1	1.0	5.05	0.059
066+1	–	3.73	0.029
067+1	4.6	0.10	0.040
076+1	51.3	0.97	0.031
086+1	0.1	0.01	0.028
096+1	5.9	7.29	0.033
143+3	0.1	0.31	0.034
171+3	68.6	0.38	0.030
580+3	0.0	0.23	0.078
601+3	1.6	1.18	0.089
606+3	0.1	0.27	0.033
607+3	1.2	0.26	0.036
609+3	145.2	0.49	0.039
611+3	0.3	4.00	0.125
612+3	111.9	0.46	0.030
614+3	3.7	0.41	0.060
615+3	103.9	0.47	0.035
623+3	–	2.27	0.282
624+3	3.8	3.29	0.047
630+3	0.1	0.05	0.025
640+3	1.1	0.01	0.033
646+3	84.4	0.01	0.032
647+3	98.2	0.12	0.037

Problem	Vamp. 9.0	LEO+Vamp.	LEO-II+E
648+3	98.2	0.12	0.037
649+3	117.5	0.25	0.037
651+3	117.5	0.09	0.029
657+3	146.6	0.01	0.028
669+3	83.1	0.20	0.041
670+3	–	0.14	0.067
671+3	214.9	0.47	0.038
672+3	–	0.23	0.034
673+3	217.1	0.47	0.042
680+3	146.3	2.38	0.035
683+3	0.3	0.27	0.053
684+3	–	3.39	0.039
716+4	–	0.40	0.033
724+4	–	1.91	0.032
741+4	–	3.70	0.042
747+4	–	1.18	0.028
752+4	–	516.00	0.086
753+4	–	1.64	0.037
764+4	0.1	0.01	0.032

Vamp. 9.0: 2.80GHz, 1GB memory, 600s time limit
LEO+Vamp.: 2.40GHz, 4GB memory, 120s time limit
LEO-II+E: 1.60GHz, 1GB memory, 60s time limit

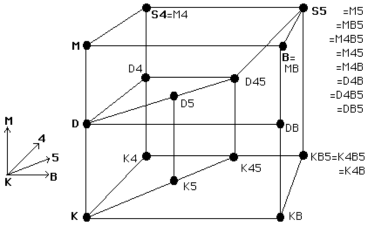
Less Lightweight Problems



Multimodal Logics

Logic Systems Interrelationships

Modal Logics Challenge



John Halleck (U Utah):
`http://www.cc.utah.edu/~nahaj/`

$100 Modal Logic Challenge:
`www.tptp.org`

Example

$$
\begin{aligned}
S4 &= K \\
&+ M(T): \quad \Box_R A \Rightarrow A \\
&+ 4: \quad \Box_R A \Rightarrow \Box_R \Box_R A
\end{aligned}
$$

Theorems:

$$S4 \not\subseteq K \quad (1)$$

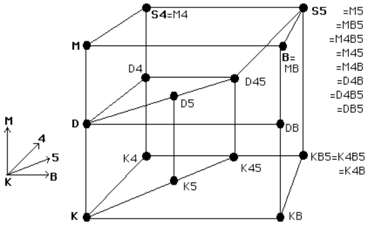
$$(M \wedge 4) \Leftrightarrow (refl.(R) \wedge trans.(R)) \quad (2)$$

Experiments

	FO-ATPs [SutcliffeEtal-07]	LEO-II + E [BePa-08]
(1)	16min + 2710s	17.3s
(2)	???	2.4s

Logic Systems Interrelationships

Modal Logics Challenge



John Halleck (U Utah):
`http://www.cc.utah.edu/~nahaj/`

$100 Modal Logic Challenge:
`www.tptp.org`

Example

$$
\begin{aligned}
S4 &= K \\
&+ \; M(T): \quad \Box_R A \Rightarrow A \\
&+ \; 4: \quad \Box_R A \Rightarrow \Box_R \Box_R A
\end{aligned}
$$

Theorems:

$$S4 \not\subseteq K \qquad (1)$$

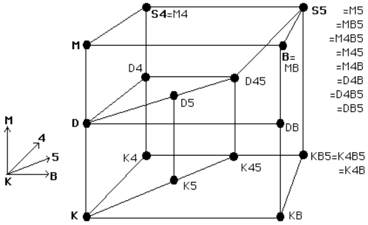
$$(M \wedge 4) \Leftrightarrow (refl.(R) \wedge trans.(R)) \qquad (2)$$

Experiments

	FO-ATPs [SutcliffeEtal-07]	LEO-II + E [BePa-08]
(1)	16min + 2710s	17.3s
(2)	???	2.4s

Logic Systems Interrelationships

Modal Logics Challenge



John Halleck (U Utah):
`http://www.cc.utah.edu/~nahaj/`

$100 Modal Logic Challenge:
`www.tptp.org`

Example

$$\begin{array}{rcl} S4 & = & K \\ & + & M(T): \quad \Box_R A \Rightarrow A \\ & + & 4: \quad \Box_R A \Rightarrow \Box_R \Box_R A \end{array}$$

Theorems:

$$\begin{array}{rcll} S4 & \not\subseteq & K & (1) \\ (M \wedge 4) & \Leftrightarrow & (\mathit{refl}.(R) \wedge \mathit{trans}.(R)) & (2) \end{array}$$

Experiments

	FO-ATPs **[SutcliffeEtal-07]**	LEO-II + E **[BePa-08]**
(1)	16min + 2710s	17.3s
(2)	???	2.4s

Even simpler: Reasoning within Multimodal Logics

Problem	LEO-II + E
$\mathtt{valid}(\Box_r\ \top)$	0.025s
$\mathtt{valid}(\Box_r\ a \Rightarrow \Box_r\ a)$	0.026s
$\mathtt{valid}(\Box_r\ a \Rightarrow \Box_s\ a)$	–
$\mathtt{valid}(\Box_s\ (\Box_r\ a \Rightarrow \Box_r\ a))$	0.026s
$\mathtt{valid}(\Box_r\ (a \wedge b) \Leftrightarrow (\Box_r\ a \wedge \Box_r\ b))$	0.044s
$\mathtt{valid}(\Diamond_r\ (a \Rightarrow b) \Rightarrow \Box_r\ a \Rightarrow \Diamond_r\ b)$	0.030s
$\mathtt{valid}(\neg \Diamond_r\ a \Rightarrow \Box_r\ (a \Rightarrow b))$	0.029s
$\mathtt{valid}(\Box_r\ b \Rightarrow \Box_r\ (a \Rightarrow b))$	0.026s
$\mathtt{valid}((\Diamond_r\ a \Rightarrow \Box_r\ b) \Rightarrow \Box_r\ (a \Rightarrow b))$	0.027s
$\mathtt{valid}((\Diamond_r\ a \Rightarrow \Box_r\ b) \Rightarrow (\Box_r\ a \Rightarrow \Box_r\ b))$	0.029s
$\mathtt{valid}((\Diamond_r\ a \Rightarrow \Box_r\ b) \Rightarrow (\Diamond_r\ a \Rightarrow \Diamond_r\ b))$	0.030s

(Normal) Multimodal Logic in HOL

Simple, Straightforward Encoding of Multimodal Logic

- base type ι: set of possible worlds
  certain terms of type $\iota \rightarrow o$: multimodal logic formulas
- multimodal logic operators:

$$\neg\, A_{\iota\rightarrow o} = (\lambda x_\iota . \neg A(x))$$

$$A_{\iota\rightarrow o} \vee B_{\iota\rightarrow o} = (\lambda x_\iota . A(x) \vee B(x))$$

$$\Box_R\, A_{\iota\rightarrow o} = (\lambda x_\iota . \forall y_\iota . R(x,y) \Rightarrow A(y))$$

Related Work

[Gallin-73], [Carpenter-98], [Merz-99],
[Brown-05], [Hardt&Smolka-07], [Kaminski&Smolka-07]

(Normal) Multimodal Logic in HOL

Simple, Straightforward Encoding of Multimodal Logic

- base type ι: set of possible worlds
  certain terms of type $\iota \to o$: multimodal logic formulas
- multimodal logic operators:

$$\neg_{(\iota\to o)\to(\iota\to o)} = \lambda A_{\iota\to o} \centerdot (\lambda x_\iota \centerdot \neg A(x))$$

$$\vee_{(\iota\to o)\to(\iota\to o)\to(\iota\to o)} = \lambda A_{\iota\to o}, B_{\iota\to o} \centerdot (\lambda x_\iota \centerdot A(x) \vee B(x))$$

$$\Box_{(\iota\to\iota\to o)\to(\iota\to o)\to(\iota\to o)} = \lambda R_{\iota\to\iota\to o}, A_{\iota\to o} \centerdot (\lambda x_\iota \centerdot \forall y_\iota \centerdot R(x,y) \Rightarrow A(y))$$

Related Work

[Gallin-73], [Carpenter-98], [Merz-99],
[Brown-05], [Hardt&Smolka-07], [Kaminski&Smolka-07]

(Normal) Multimodal Logic in HOL

Encoding of Validity

$$\texttt{valid}\, A_{\iota\rightarrow o} \quad = \quad (\forall w_\iota . A(w))$$

Encoding of Validity

$$\mathtt{valid} \quad = \quad \lambda A_{\iota \to o} \centerdot (\forall w_\iota \centerdot A(w))$$

Example Proof: $\mathbf{valid}(\Box_s\,(\Box_r\, a \Rightarrow \Box_r\, a))$

Initialisation of problem

$$\neg\mathbf{valid}(\Box_s\,(\Box_r\, a \Rightarrow \Box_r\, a))$$

Definition expansion

$$\neg(\forall x_\iota \centerdot \forall y_\iota \centerdot \neg s(x,y) \vee ((\neg(\forall u_\iota \centerdot \neg r(y,u) \vee a(u))) \vee (\forall v_\iota \centerdot \neg r(y,v) \vee a(v)))$$

Normalisation (x, y, u are now Skolem constants, V is a variable)

$$\begin{array}{ll} s(x,y) & \neg a(u) \\ r(y,u) & a(V) \vee \neg r(y,V) \end{array}$$

Translation to first-order logic [Kerber-94], [Hurd-02], [MengPaulson-04]

$$\begin{array}{ll} @_{(io)\_i}(@_{(i(io))\_i}(s,x),y) & \neg @_{(\iota o)\_\iota}(a,u) \\ @_{(io)\_i}(@_{(i(io))\_i}(r,y),u) & @_{(\iota o)\_\iota}(a,V) \vee \neg @_{(io)\_i}(@_{(i(io))\_i}(r,y),V) \end{array}$$

Example Proof: $\textbf{valid}(\Box_s\,(\Box_r\, a \Rightarrow \Box_r\, a))$

Initialisation of problem

$$\neg\textbf{valid}(\Box_s\,(\Box_r\, a \Rightarrow \Box_r\, a))$$

Definition expansion

$$\neg(\forall x_\iota \blacksquare \forall y_\iota \blacksquare \neg s(x,y) \vee ((\neg(\forall u_\iota \blacksquare \neg r(y,u) \vee a(u))) \vee (\forall v_\iota \blacksquare \neg r(y,v) \vee a(v)))$$

Normalisation (x, y, u are now Skolem constants, V is a variable)

$$\begin{array}{ll} s(x,y) & \neg a(u) \\ r(y,u) & a(V) \vee \neg r(y,V) \end{array}$$

Translation to first-order logic [Kerber-94], [Hurd-02], [MengPaulson-04]

$$\begin{array}{ll} @_{(io)\_i}(@_{(i(io))\_i}(s,x),y) & \neg @_{(\iota o)\_\iota}(a,u) \\ @_{(io)\_i}(@_{(i(io))\_i}(r,y),u) & @_{(\iota o)\_\iota}(a,V) \vee \neg @_{(io)\_i}(@_{(i(io))\_i}(r,y),V) \end{array}$$

Example Proof: $\mathbf{valid}(\Box_s\,(\Box_r\, a \Rightarrow \Box_r\, a))$

Initialisation of problem

$$\neg\mathbf{valid}(\Box_s\,(\Box_r\, a \Rightarrow \Box_r\, a))$$

Definition expansion

$$\neg(\forall x_\iota\text{.}\forall y_\iota\text{.}\neg s(x,y) \vee ((\neg(\forall u_\iota\text{.}\neg r(y,u) \vee a(u))) \vee (\forall v_\iota\text{.}\neg r(y,v) \vee a(v)))$$

Normalisation (x, y, u are now Skolem constants, V is a variable)

$$\begin{array}{ll} s(x,y) & \neg a(u) \\ r(y,u) & a(V) \vee \neg r(y,V) \end{array}$$

Translation to first-order logic [Kerber-94], [Hurd-02], [MengPaulson-04]

$$\begin{array}{ll} @_{(io)\_i}(@_{(i(io))\_i}(s,x),y) & \neg @_{(\iota o)\_\iota}(a,u) \\ @_{(io)\_i}(@_{(i(io))\_i}(r,y),u) & @_{(\iota o)\_\iota}(a,V) \vee \neg @_{(io)\_i}(@_{(i(io))\_i}(r,y),V) \end{array}$$

Example Proof: $\textbf{valid}(\Box_s(\Box_r\, a \Rightarrow \Box_r\, a))$

Initialisation of problem

$$\neg\textbf{valid}(\Box_s(\Box_r\, a \Rightarrow \Box_r\, a))$$

Definition expansion

$$\neg(\forall x_\iota.\forall y_\iota.\neg s(x,y) \vee ((\neg(\forall u_\iota.\neg r(y,u) \vee a(u))) \vee (\forall v_\iota.\neg r(y,v) \vee a(v)))$$

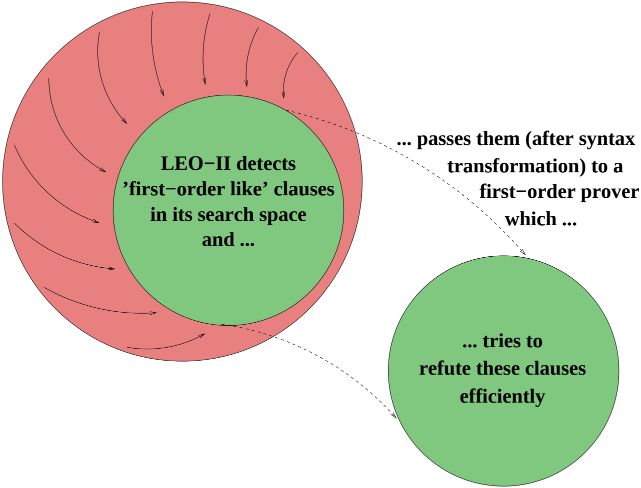
Normalisation (x, y, u are now Skolem constants, V is a variable)

$$\begin{array}{ll} s(x,y) & \neg a(u) \\ r(y,u) & a(V) \vee \neg r(y,V) \end{array}$$

Translation to first-order logic [Kerber-94], [Hurd-02], [MengPaulson-04]

$$\begin{array}{ll} @_{(io)\_i}(@_{(i(io))\_i}(s,x),y) & \neg @_{(\iota o)\_\iota}(a,u) \\ @_{(io)\_i}(@_{(i(io))\_i}(r,y),u) & @_{(\iota o)\_\iota}(a,V) \vee \neg @_{(io)\_i}(@_{(i(io))\_i}(r,y),V) \end{array}$$

Architecture of LEO-II



More Examples . . .

A simple equation between modal logic formulas

$$\forall R_{\blacksquare}\forall A_{\blacksquare}\forall B_{\blacksquare}(\Box_R(A \vee B)) = (\Box_R(B \vee A))$$

- initialisation, definition expansion and normalisation:

$$\begin{array}{l}(\lambda X_\iota.\forall Y_{\iota\blacksquare}\neg((r\,X)\,Y) \vee (a\,Y) \vee (b\,Y)) \\ \neq \\ (\lambda X_\iota.\forall Y_{\iota\blacksquare}\neg((r\,X)\,Y) \vee (b\,Y) \vee (a\,Y))\end{array}$$

More Examples ...

A simple equation between modal logic formulas

$$\forall R.\forall A.\forall B.(\Box_R (A \vee B)) = (\Box_R (B \vee A))$$

- functional extensionality :

$$(\forall Y_\iota.\neg((r\,w)\,Y) \vee (a\,Y) \vee (b\,Y))$$
$$\neq$$
$$(\forall Y_\iota.\neg((r\,w)\,Y) \vee (b\,Y) \vee (a\,Y))$$

More Examples . . .

A simple equation between modal logic formulas

$$\forall R_{\blacksquare}\forall A_{\blacksquare}\forall B_{\blacksquare}(\square_R (A \vee B)) = (\square_R (B \vee A))$$

- functional extensionality and Boolean extensionality:

$$(\forall Y_{\iota}\blacksquare \neg((r\, w)\, Y) \vee (a\, Y) \vee (b\, Y))$$
$$\not\Leftrightarrow$$
$$(\forall Y_{\iota}\blacksquare \neg((r\, w)\, Y) \vee (b\, Y) \vee (a\, Y))$$

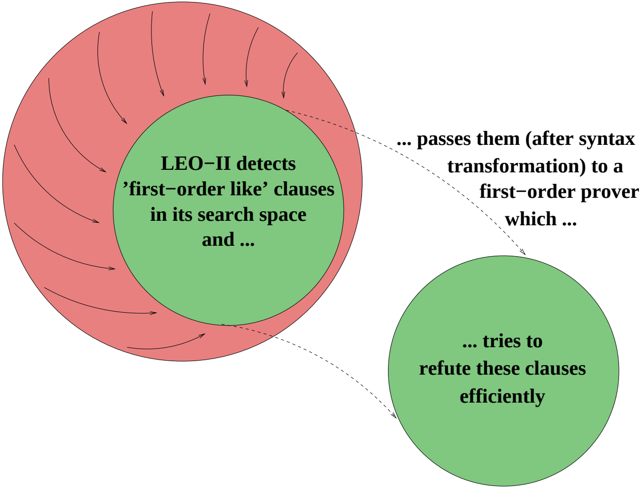
More Examples . . .

A simple equation between modal logic formulas

$$\forall R. \forall A. \forall B. (\Box_R (A \vee B)) = (\Box_R (B \vee A))$$

- normalisation:
  $40 : (b\,V) \vee (a\,V) \vee \neg((r\,w)\,V) \vee \neg((r\,w)\,W) \vee (b\,W) \vee (a\,W)$
  $41 : ((r\,w)\,z) \vee ((r\,w)\,v)$
  $42 : \neg(a\,z) \vee ((r\,w)\,v)$
  $43 : \neg(b\,z) \vee ((r\,w)\,v)$
  $44 : ((r\,w)\,z) \vee \neg(a\,v)$
  $45 : \neg(a\,z) \vee \neg(a\,v)$
  $46 : \neg(b\,z) \vee \neg(a\,v)$
  $47 : ((r\,w)\,z) \vee \neg(b\,v)$
  $48 : \neg(a\,z) \vee \neg(b\,v)$
  $49 : \neg(b\,z) \vee \neg(b\,v)$
- total proving time (notebook with 1.60GHz, 1GB): 0.071s

Architecture of LEO-II



More Examples . . .

A simple equation between modal logic formulas

$$\forall R.\forall A.\forall B.(\Box_R (A \vee B)) \doteq (\Box_R (B \vee A))$$

where $\doteq$ is defined as $\lambda X, Y.\forall P.(P\,X) \Rightarrow (P\,Y)$

- initialisation, definition expansion and normalisation:

$$(p\,(\lambda X_\iota.\forall Y_\iota.\neg((r\,X)\,Y) \vee (a\,Y) \vee (b\,Y)))$$
$$\neg(p\,(\lambda X_\iota.\forall Y_\iota.\neg((r\,X)\,Y) \vee (b\,Y) \vee (a\,Y)))$$

More Examples ...

A simple equation between modal logic formulas

$$\forall R_\blacksquare \forall A_\blacksquare \forall B_\blacksquare (\Box_R (A \vee B)) \doteq (\Box_R (B \vee A))$$

where $\doteq$ is defined as $\lambda X, Y.\forall P_\blacksquare (P\,X) \Rightarrow (P\,Y)$

- resolution:

$$(p\,(\lambda X_\iota.\forall Y_{\iota\blacksquare} \neg((r\,X)\,Y) \vee (a\,Y) \vee (b\,Y)))$$
$$\neq$$
$$(p\,(\lambda X_\iota.\forall Y_{\iota\blacksquare} \neg((r\,X)\,Y) \vee (b\,Y) \vee (a\,Y)))$$

More Examples ...

A simple equation between modal logic formulas

$$\forall R_{\blacksquare} \forall A_{\blacksquare} \forall B_{\blacksquare} (\Box_R (A \vee B)) \doteq (\Box_R (B \vee A))$$

where $\doteq$ is defined as $\lambda X, Y. \forall P_{\blacksquare} (P\,X) \Rightarrow (P\,Y)$

- decomposition:

$$(\lambda X_\iota . \forall Y_{\iota\blacksquare} \neg((r\,X)\,Y) \vee (a\,Y) \vee (b\,Y))$$
$$\neq$$
$$(\lambda X_\iota . \forall Y_{\iota\blacksquare} \neg((r\,X)\,Y) \vee (b\,Y) \vee (a\,Y))$$

More Examples ...

A simple equation between modal logic formulas

$$\forall R.\forall A.\forall B.(\Box_R\,(A \vee B)) \doteq (\Box_R\,(B \vee A))$$

where $\doteq$ is defined as $\lambda X, Y.\forall P.(P\,X) \Rightarrow (P\,Y)$

- functional and Boolean extensionality:

$$(\forall Y_\iota.\neg((r\,w)\,Y) \vee (a\,Y) \vee (b\,Y))$$
$$\not\Leftrightarrow$$
$$(\forall Y_\iota.\neg((r\,w)\,Y) \vee (b\,Y) \vee (a\,Y))$$

More Examples . . .

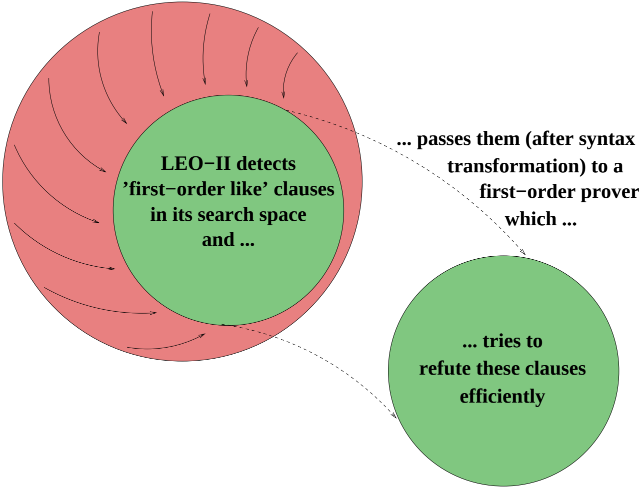
A simple equation between modal logic formulas

$$\forall R. \forall A. \forall B. (\Box_R (A \vee B)) \doteq (\Box_R (B \vee A))$$

where $\doteq$ is defined as $\lambda X, Y. \forall P. (P\,X) \Rightarrow (P\,Y)$

- normalisation: . . . see previous example . . .
- total proving time is $0.166s$

Architecture of LEO-II



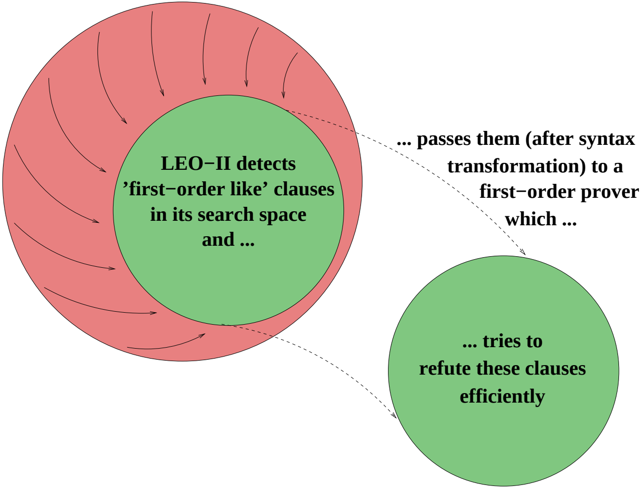
More Examples . . .

In modal logic **K**, the axioms T and 4 are equivalent to reflexivity and transitivity of the accessibility relation R

$$\forall R. (\forall A. \mathtt{valid}(\Box_R A \Rightarrow A) \wedge \mathtt{valid}(\Box_R A \Rightarrow \Box_R \Box_R A)) \\ \Leftrightarrow (\mathtt{reflexive}(R) \wedge \mathtt{transitive}(R))$$

- processing in LEO-II analogous to previous example
- now 70 clauses are passed to E
- E generates **21769** clauses before finding the empty clause
- total proving time 2.4s
- this proof cannot be found in LEO-II alone

Architecture of LEO-II



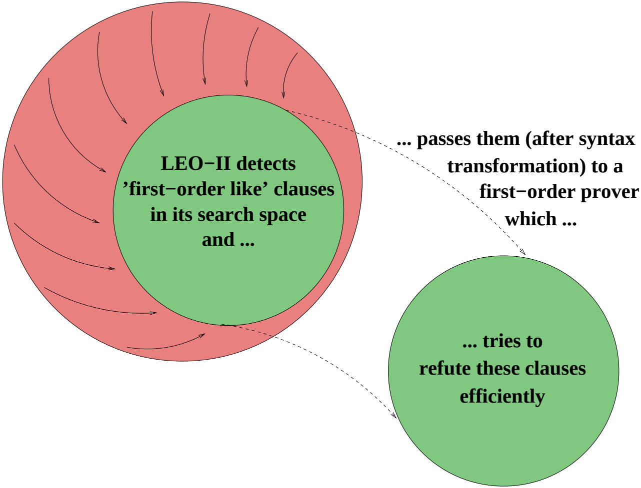
More Examples ...

$\mathrm{S4} \not\subseteq \mathrm{K}$: Axioms T and 4 are not valid in modal logic K

$$\neg\forall R.\forall A.\forall B.(\mathrm{valid}(\Box_R A \Rightarrow A)) \wedge (\mathrm{valid}(\Box_R B \Rightarrow \Box_R \Box_R B))$$

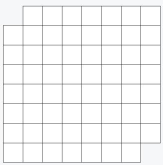
- LEO-II shows that axiom T is not valid
- R is instantiated with $\neq$ via primitive substitution
- total proving time 17.3s

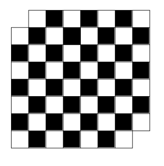
Architecture of LEO-II



Representation (and the right System Architecture) Matters!

A general lesson in AI ...

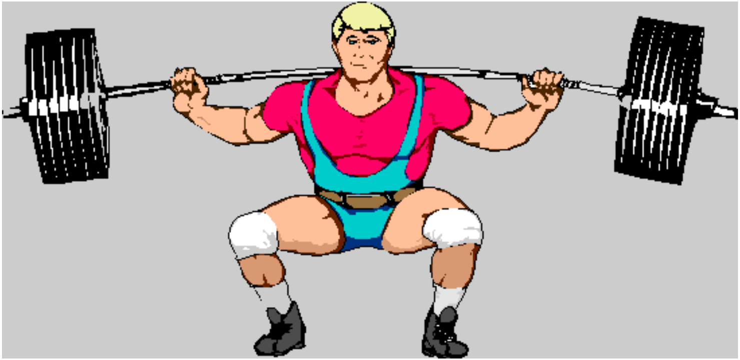




... and a specific lesson here

FOL	HOL
$+$	$+$
FO-ATP	LEO-II $+$ FO-ATP

More Example Problems



Access Control Logics
(thanks to: Catalin Hritcu)

Access Control Logics

Motivation

(Specialized) logics for describing, analyzing and enforcing access control policies

Example (from [GargAbadi-08]):

- If the admin says that file1 should be deleted, then this must be the case
  `(admin says deletefile1) ⊃ deletefile1`
- admin trusts Bob to decide whether file1 should be deleted
  `admin says ((Bob says deletefile1) ⊃ deletefile1)`
- Bob wants to delete file1
  `Bob says deletefile1`

More Example Problems

Deepak Garg, Martín Abadi:
A Modal Deconstruction of Access Control Logics
FoSSaCS 2008: 216-230, LNCS 4962 ©Springer

- translation of a logic of access control with "says" operator into classical modal logic S4
- sound and complete
- extends to logics with
  - "speaks for" relation ($\mathrm{ICL}^{\Rightarrow}$)
  - Boolean combinations of principals (ICL^{B})
- Direct second-order modeling of $\mathrm{ICL}^{\Rightarrow}$

So, let's realize this quickly in LEO-II ...

More Example Problems

Deepak Garg, Martín Abadi:
A Modal Deconstruction of Access Control Logics
FoSSaCS 2008: 216-230, LNCS 4962 ©Springer

- translation of a logic of access control with "says" operator into classical modal logic S4
- sound and complete
- extends to logics with
  - "speaks for" relation ($\mathrm{ICL}^{\Rightarrow}$)
  - Boolean combinations of principals (ICL^{B})
- Direct second-order modeling of $\mathrm{ICL}^{\Rightarrow}$

So, let's realize this quickly in LEO-II ...

ICL Logic translated to K

$$s ::= p \mid s_1 \wedge s_2 \mid s_1 \vee s_2 \mid s_1 \supset s_2 \mid \bot \mid \top \mid A \,\texttt{says}\, s$$

Translation a la [GargAbadi-08]

$$\begin{aligned}
\lceil p \rceil &= \Box p \\
\lceil s \wedge t \rceil &= \lceil s \rceil \wedge \lceil t \rceil \\
\lceil s \vee t \rceil &= \lceil s \rceil \vee \lceil t \rceil \\
\lceil s \supset t \rceil &= \lceil s \rceil \supset \lceil t \rceil \\
\lceil \top \rceil &= \top \\
\lceil \bot \rceil &= \bot \\
\lceil A \texttt{ says } s \rceil &= \Box (A \vee \lceil s \rceil)
\end{aligned}$$

ICL Logic translated to K

$$s_{\iota\to o} ::= p \mid s_1 \wedge s_2 \mid s_1 \vee s_2 \mid s_1 \supset s_2 \mid \bot \mid \top \mid A \, \texttt{says} \, s$$

Representation/Translation of Principals and Atoms

- principals: $A_{\iota\to o}, B_{\iota\to o}, C_{\iota\to o}, \ldots$
  translation of principals: $\texttt{princ}\, A_{\iota\to o} = A_{\iota\to o}$
- atomic propositions: $p_{\iota\to o}, q_{\iota\to o}, s_{\iota\to o}, t_{\iota\to o}, \ldots$
  translation of atomic propositions: $\texttt{atom}\, p_{\iota\to o} = \Box_r \, p_{\iota\to o}$
- fixed accessibility relation: $r_{\iota\to\iota\to o}$

ICL Logic translated to K

$$s_{\iota\to o} ::= p \mid s_1 \wedge s_2 \mid s_1 \vee s_2 \mid s_1 \supset s_2 \mid \bot \mid \top \mid A \,\texttt{says}\, s$$

Representation/Translation of ICL Connectives

- translation of $\wedge$: $p_{\iota\to o} \wedge q_{\iota\to o} = p \wedge q$
- translation of $\vee$: $p_{\iota\to o} \vee q_{\iota\to o} = p \vee q$
- translation of $\supset$: $p_{\iota\to o} \supset q_{\iota\to o} = \Box_r\,(p \Rightarrow q)$
- translation of $\bot$ and $\top$: $\bot = \bot$ and $\top = \top$
- translation of $\texttt{says}$: $A_{\iota\to o}\,\texttt{says}\,s_{\iota\to o} = \Box_r\,(A \vee s)$

Notion of Validity (wrt K)

$$\texttt{icl\_valid}\,p = \texttt{valid}\,p$$

ICL Logic translated to S4

Addition of Modal Logic Axioms for S4

$$\forall P_{\iota\to o}.\mathtt{valid}(\Box_r P \Rightarrow P)$$

$$\forall P_{\iota\to o}.\mathtt{valid}(\Box_r P \Rightarrow \Box_r \Box_r P)$$

Notion of Validity (wrt S4)

$$\mathtt{icl\_valid}\, p = \mathtt{valid}\, p$$

Example 1 from [GargAbadi-08]

Original Formulation

$$(\texttt{admin}\,\texttt{says}\,\texttt{deletefile1}) \supset \texttt{deletefile1}$$
$$\texttt{admin}\,\texttt{says}\,((\texttt{Bob}\,\texttt{says}\,\texttt{deletefile1}) \supset \texttt{deletefile1})$$
$$\texttt{Bob}\,\texttt{says}\,\texttt{deletefile1}$$

ICL Problem Encoding for LEO-II

$$\texttt{icl\_valid}\ [((\texttt{princ admin})\ \texttt{says}\ (\texttt{atom deletefile1})) \supset (\texttt{atom deletefile1})] \quad \text{(ax1)}$$
$$\texttt{icl\_valid}\ [(\texttt{princ admin})\ \texttt{says}\ (((\texttt{princ Bob})\ \texttt{says}\ (\texttt{atom deletefile1})) \supset (\texttt{atom deletefile1}))] \quad \text{(ax2)}$$
$$\texttt{icl\_valid}\ [(\texttt{princ Bob})\ \texttt{says}\ (\texttt{atom deletefile1})] \quad \text{(ax3)}$$
$$\texttt{icl\_valid}\ (\texttt{atom deletefile1})] \quad \text{(conj)}$$

Example 1 from [GargAbadi-08]

Definition Expansion: Translation to HOL and FOL via S4

- Example (ax3):

  `icl_valid [(princ Bob) says (atom deletefile1)]`

- Expansion to S4:

  $$\mathtt{valid}\ [\Box_r\ (\mathtt{Bob} \vee (\Box_r\ \mathtt{deletefile1}))]$$

- Expansion to HOL:

  $$\forall w_\iota\text{.}((\lambda x_\iota\text{.}\forall y_\iota\text{.}(r\,x\,y) \Rightarrow ((\mathtt{Bob}\,y) \vee ((\lambda z_\iota\text{.}\forall u_\iota\text{.}(r\,z\,u) \Rightarrow (\mathtt{deletefile1}\,u))\,x))\,)\,w)$$

- β-reduction to FO-like

  $$\forall w_\iota\text{.}(\forall y_\iota\text{.}(r\,w\,y) \Rightarrow ((\mathtt{Bob}\,y) \vee \forall u_\iota\text{.}(r\,w\,u) \Rightarrow (\mathtt{deletefile1}\,u)))$$

- Further normalisation

  ...

Definition Expansion: Translation to HOL and FOL via S4

- Example (ax3):

  `icl_valid [(princ Bob) says (atom deletefile1)]`

- Expansion to S4:

  $$\texttt{valid}\ [\Box_r\ (\text{Bob} \vee (\Box_r\ \texttt{deletefile1}))]$$

- Expansion to HOL:

  $$\forall w_{\iota \blacksquare}((\lambda x_{\iota \blacksquare}\forall y_{\iota \blacksquare}(r\,x\,y) \Rightarrow ((\text{Bob}\,y) \vee ((\lambda z_{\iota \blacksquare}\forall u_{\iota \blacksquare}(r\,z\,u) \Rightarrow (\texttt{deletefile1}\,u))\,x)))\,w)$$

- β-reduction to FO-like

  $$\forall w_{\iota \blacksquare}(\forall y_{\iota \blacksquare}(r\,w\,y) \Rightarrow ((\text{Bob}\,y) \vee \forall u_{\iota \blacksquare}(r\,w\,u) \Rightarrow (\texttt{deletefile1}\,u)))$$

- Further normalisation

  ...

Example 1 from [GargAbadi-08]

Definition Expansion: Translation to HOL and FOL via S4

- Example (ax3):

  $$\texttt{icl\_valid [(princ Bob) says (atom deletefile1)]}$$

- Expansion to S4:

  $$\texttt{valid}\ [\Box_r\ (\mathrm{Bob} \vee (\Box_r\ \texttt{deletefile1}))]$$

- Expansion to HOL:

  $$\forall w_\iota.((\lambda x_\iota.\forall y_\iota.(r\,x\,y) \Rightarrow ((\mathrm{Bob}\,y)\vee((\lambda z_\iota.\forall u_\iota.(r\,z\,u) \Rightarrow (\texttt{deletefile1}\,u))\,x)))\,w)$$

- β-reduction to FO-like

  $$\forall w_\iota.(\forall y_\iota.(r\,w\,y) \Rightarrow ((\mathrm{Bob}\,y) \vee \forall u_\iota.(r\,w\,u) \Rightarrow (\texttt{deletefile1}\,u)))$$

- Further normalisation

  ...

Example 1 from [GargAbadi-08]

Definition Expansion: Translation to HOL and FOL via S4

- Example (ax3):

  icl_valid [(princ Bob) says (atom deletefile1)]

- Expansion to S4:

  $$\mathtt{valid}\ [\Box_r\,(\mathtt{Bob} \vee (\Box_r\,\mathtt{deletefile1}))]$$

- Expansion to HOL:

  $$\forall w_{\iota}\mathbf{.}((\lambda x_{\iota}\mathbf{.}\forall y_{\iota}\mathbf{.}(r\,x\,y) \Rightarrow ((\mathtt{Bob}\,y) \vee ((\lambda z_{\iota}\mathbf{.}\forall u_{\iota}\mathbf{.}(r\,z\,u) \Rightarrow (\mathtt{deletefile1}\,u))\,x)))\,w)$$

- β-reduction to FO-like

  $$\forall w_{\iota}\mathbf{.}(\forall y_{\iota}\mathbf{.}(r\,w\,y) \Rightarrow ((\mathtt{Bob}\,y) \vee \forall u_{\iota}\mathbf{.}(r\,w\,u) \Rightarrow (\mathtt{deletefile1}\,u)))$$

- Further normalisation

  ...

Definition Expansion: Translation to HOL and FOL via S4

- Example (ax3):

  `icl_valid [(princ Bob) says (atom deletefile1)]`

- Expansion to S4:

  $$\texttt{valid}\ [\Box_r\,(\mathrm{Bob} \vee (\Box_r\,\texttt{deletefile1}))]$$

- Expansion to HOL:

  $$\forall w_\iota \centerdot ((\lambda x_\iota \centerdot \forall y_\iota \centerdot (r\,x\,y) \Rightarrow ((\mathrm{Bob}\,y) \vee ((\lambda z_\iota \centerdot \forall u_\iota \centerdot (r\,z\,u) \Rightarrow (\texttt{deletefile1}\,u))\,x)))\,w)$$

- β-reduction to FO-like

  $$\forall w_\iota \centerdot (\forall y_\iota \centerdot (r\,w\,y) \Rightarrow ((\mathrm{Bob}\,y) \vee \forall u_\iota \centerdot (r\,w\,u) \Rightarrow (\texttt{deletefile1}\,u)))$$

- Further normalisation

  ...

Example 1 from [GargAbadi-08]

After Complete Definition Expansion and Normalisation of Ex. 1

- 11 clauses, all FO-like
- example clause:
  $(admin\ V^5) \vee (deletefile1\ V^8) \vee \neg(rel\ V^2\ V^5) \vee \neg(rel\ V^5\ V^8)$
  $\vee(rel\ V^5\ (sk4\ V^8\ @V^2\ @V5))$
- Proof immediately found be E
- Total proving time 0.095 sec.

ICL Logic translated to K and S4

Filename	Status	LEO (s) + ATP-calls (s)	Total (s)
ICL_unit_k.thf	–	...	–
ICL_unit_s4.thf	$\surd$	0.008 0.023	0.031
		$s \supset (A \text{ says } s)$	
ICL_cuc_k.thf	$\surd$	0.004 0.022	0.026
ICL_cuc_s4.thf	$\surd$	0.012 0.024	0.036
		$(A \text{ says } (s \supset t)) \supset (A \text{ says } s) \supset (A \text{ says } t)$	
ICL_idem_k.thf	–	...	–
ICL_idem_s4.thf	$\surd$	0.040 0.050	0.090
		$(A \text{ says } A \text{ says } s) \supset (A \text{ says } s)$	
ICL_ex1_k.thf	$\surd$	0.040 0.055	0.095
ICL_ex1_s4.thf	$\surd$	0.028 0.084	0.112
		$(\texttt{admin} \text{ says } \texttt{deletefile1}) \supset \texttt{deletefile1}$	
		$\texttt{admin} \text{ says } ((\texttt{Bob} \text{ says } \texttt{deletefile1}) \supset \texttt{deletefile1})$	
		$\texttt{Bob} \text{ says } \texttt{deletefile1}$	

ICL$^{\Rightarrow}$ Logic translated to K and S4

Filename	Status	LEO (s) + ATP-calls (s)	Total (s)
ICLimp_refl_k.thf	$\surd$	0.008 0.045	0.053
ICLimp_refl_s4.thf	$\surd$	0.040 0.084	0.124
	$A \Rightarrow A$		
ICLimp_trans_k.thf	$\surd$	0.012 0.021	0.033
ICLimp_trans_s4.thf	$\surd$	0.012 0.020	0.032
	$(A \Rightarrow B) \supset (B \Rightarrow C) \supset (A \Rightarrow C)$		
ICLimp_speaking_k.thf	$\surd$	0.008 0.020	0.028
ICLimp_speaking_s4.thf	$\surd$	0.024 0.021	0.045
	$(A \Rightarrow B) \supset (A \text{ says } s) \supset (B \text{ says } s)$		
ICLimp_handoff_k.thf	–	–	–
ICLimp_handoff_s4.thf	$\surd$	0.028 0.047	0.075
	$(B \text{ says } (A \Rightarrow B)) \supset (A \Rightarrow B)$		
ICLimp_ex2_s4.thf	$\surd$	0.112 0.263	0.375
	$(\texttt{admin says deletefile1}) \supset \texttt{deletefile1}$		
	$\texttt{admin says} ((\texttt{Bob says deletefile1}) \supset \texttt{deletefile1})$		
	$\texttt{Bob says} (\texttt{Alice} \Rightarrow \texttt{Bob})$		
	$\texttt{Alice says deletefile1}$		

ICL^B Logic translated to K and S4

Filename	Status	LEO (s) + ATP-calls (s)	Total (s)
ICLb_trust_k.thf	–	...	–
ICLb_trust_s4.thf	$\surd$	0.008 0.022	0.030
	$(\bot \,\mathsf{says}\, s) \supset s$		
ICLb_untrust_k.thf	$\surd$	0.004 0.020	0.024
ICLb_untrust_s4.thf	$\surd$	0.028 0.046	0.074
	If $A \equiv \top$ **then** $\vdash A \,\mathsf{says}\, s$		
ICLb_cuc_k.thf	$\surd$	0.012 0.021	0.033
ICLb_cuc_s4.thf	$\surd$	0.024 0.021	0.045
	$((A \supset B) \,\mathsf{says}\, s) \supset (A \,\mathsf{says}\, s) \supset (B \,\mathsf{says}\, s)$		
ICLb_ex3_k.thf	–	...	–
ICLb_ex3_s4.thf	$\surd$	0.016 0.024	0.040
	$(\mathtt{admin} \supset \bot) \,\mathsf{says}\, \mathtt{deletefile1}$ $\mathtt{admin} \,\mathsf{says}\, ((\mathtt{Bob} \supset \mathtt{admin}) \,\mathsf{says}\, \mathtt{deletefile1})$ $\mathtt{Bob} \,\mathsf{says}\, \mathtt{deletefile1}$		

$\mathrm{ICL}^{\forall}$ Logic translated to K and S4

Filename	Status	LEO (s) + ATP-calls (s)	Total (s)
ICLall_unit.thf	$\surd$	0.104 0.043 0.041 0.042	0.230
		$s \supset (A \text{ says } s)$	
ICLall_cuc.thf	$\surd$	0.140 0.043 0.039 0.040 0.043	0.306
		$(A \text{ says } (s \supset t)) \supset (A \text{ says } s) \supset (A \text{ says } t)$	
ICLall_idem.thf	$\surd$	0.056 0.019 ...	0.131
		$(A \text{ says } A \text{ says } s) \supset (A \text{ says } s)$	
ICLall_ex1.thf	$\surd$	0.468 0.042 ...	0.716
		$(\texttt{admin} \text{ says } \texttt{deletefile1}) \supset \texttt{deletefile1}$ $\texttt{admin} \text{ says } ((\texttt{Bob} \text{ says } \texttt{deletefile1}) \supset \texttt{deletefile1})$ $\texttt{Bob} \text{ says } \texttt{deletefile1}$	
ICLall_ex2.thf	$\surd$	0.000 0.062 ...	2.081
		$(\texttt{admin} \text{ says } \texttt{deletefile1}) \supset \texttt{deletefile1}$ $\texttt{admin} \text{ says } ((\texttt{Bob} \text{ says } \texttt{deletefile1}) \supset \texttt{deletefile1})$ $\texttt{Bob} \text{ says } (\texttt{Alice} \Rightarrow \texttt{Bob})$ $\texttt{Alice} \text{ says } \texttt{deletefile1}$	

Conclusion

What makes LEO-II strong? The combination of

- expressive higher-order representations
- reduction to first-order representations
- cooperation with first-order ATPs
- higher-order termsharing and termindexing techniques

Try LEO-II (running under Ocaml 3.10)

- Website: `http://www.ags.uni-sb.de/~leo`
  - download version, very easy to install
  - online demo
- Systems on TPTP:
  `http://www.cs.miami.edu/~tptp/cgi-bin/SystemOnTPTP`

... there is much left to be done!

LEO-II

- Equational Reasoning
- Termindexing
- Handling of Definitions

Cooperat. with Specialist Reasoners

- Prop. Logic, Arithmetic, ...Z3?
- Logic Translations
- Feedback for LEO-II
- Proof Transf./Verification
- Agent-based Architecture

Integration into Proof Assistants

- Relevance of Axioms
- Proof Transf./Verification

International Infrastructure

- TPTP Language(s) for HOL
- Repository of Proof Problems
- HOL Prover Contest

Applications

Formal Methods, Ontology Reasoning, Comp. Linguistics, Logic System Interrelationships, ...

… there is much left to be done!

LEO-II

- Equational Reasoning
- Termindexing
- Handling of Definitions

Cooperat. with Specialist Reasoners

- Prop. Logic, Arithmetic, … Z3?
- Logic Translations
- Feedback for LEO-II
- Proof Transf./Verification
- Agent-based Architecture

Integration into Proof Assistants

- Relevance of Axioms
- Proof Transf./Verification

International Infrastructure

- TPTP Language(s) for HOL
- Repository of Proof Problems
- HOL Prover Contest

Applications

Formal Methods, Ontology Reasoning, Comp. Linguistics, Logic System Interrelationships, …

... there is much left to be done!

LEO-II

- Equational Reasoning
- Termindexing
- Handling of Definitions

Cooperat. with Specialist Reasoners

- Prop. Logic, Arithmetic, ... Z3?
- Logic Translations
- Feedback for LEO-II
- Proof Transf./Verification
- Agent-based Architecture

Integration into Proof Assistants

- Relevance of Axioms
- Proof Transf./Verification

International Infrastructure

- TPTP Language(s) for HOL
- Repository of Proof Problems
- HOL Prover Contest

Applications

Formal Methods, Ontology Reasoning, Comp. Linguistics, Logic System Interrelationships, ...

… there is much left to be done!

LEO-II

- Equational Reasoning
- Termindexing
- Handling of Definitions

Cooperat. with Specialist Reasoners

- Prop. Logic, Arithmetic, … Z3?
- Logic Translations
- Feedback for LEO-II
- Proof Transf./Verification
- Agent-based Architecture

Integration into Proof Assistants

- Relevance of Axioms
- Proof Transf./Verification

International Infrastructure

- TPTP Language(s) for HOL
- Repository of Proof Problems
- HOL Prover Contest

Applications

Formal Methods, Ontology Reasoning, Comp. Linguistics, Logic System Interrelationships, …

... there is much left to be done!

LEO-II

- Equational Reasoning
- Termindexing
- Handling of Definitions

Cooperat. with Specialist Reasoners

- Prop. Logic, Arithmetic, ... Z3?
- Logic Translations
- Feedback for LEO-II
- Proof Transf./Verification
- Agent-based Architecture

Integration into Proof Assistants

- Relevance of Axioms
- Proof Transf./Verification

International Infrastructure

- TPTP Language(s) for HOL
- Repository of Proof Problems
- HOL Prover Contest

Applications

Formal Methods, Ontology Reasoning, Comp. Linguistics, Logic System Interrelationships, ...

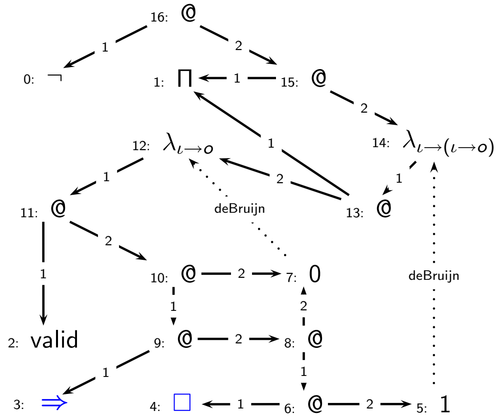
More Information on LEO-II

- Website with online version of LEO-II:

  `http://www.ags.uni-sb.de/~leo`

- System description [IJCAR-08]
- TPTP THF input syntax [IJCAR-THF-08]
  Higher-Order TPTP Infrastructure EU project THFTPTP
- Reasoning in and about multimodal logic [Festschrift-Andrews-08]

Term Graph for: $\neg\forall R.\forall A.(\mathbf{valid}(\Box_R A \Rightarrow A))$

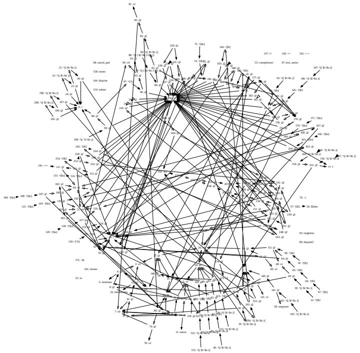


Term graph videos: `http://www.ags.uni-sb.de/~leo/art`

Latest Application of LEO-II: Dancefloor Animation

Grooving to an animation of LEO-II's dynamically growing termgraph (while LEO-II is proving Cantor's theorem)

Termsharing



In LEO-II:

- Terms as unique instances
- Perfect Term Sharing
- Shallow data structures

Features:

- β-η-normalization
- DeBruijn indices
- local contexts for polymorphic type variables

More Examples ...

$\mathtt{T} \not\subseteq \mathtt{K}$: Axiom T is not valid in modal logic K

$$\neg\forall R.\forall A.(\mathrm{valid}(\Box_R\, A \Rightarrow A))$$

- initialisation, definition expansion and normalization generates:

$$((R\,W)\,s^{A,W,R}) \vee (A\,W)$$
$$\neg(A\,s^{A,W,R}) \vee (A\,W)$$

where $s^{A,W,R} = (((s\,A)\,W)\,R)$ is a new Skolem term

More Examples ...

$\mathtt{T} \not\subseteq \mathtt{K}$: Axiom T is not valid in modal logic K

$$\neg\forall R\text{.}\forall A\text{.}(\text{valid}(\Box_R A \Rightarrow A))$$

- the refutation employs only the former clause

$$((R\,W)\,s^{A,W,R}) \vee (A\,W)$$

More Examples . . .

$\mathtt{T} \not\subseteq \mathtt{K}$: Axiom T is not valid in modal logic K

$$\neg \forall R. \forall A. (\text{valid}(\Box_R A \Rightarrow A))$$

- $((R\,W)\,s^{A,W,R}) \vee (A\,W)$
- LEO-II 'guesses' the instantiations

$$R \leftarrow \lambda X, Y.((M\,X)\,Y) \neq ((N\,X)\,Y)$$
$$A \leftarrow \lambda X.(O\,X) \neq (P\,X)$$

with primitive substitution rule (M, N, O, P are new free variables) . . .

More Examples ...

$\mathtt{T} \not\subseteq \mathtt{K}$: Axiom T is not valid in modal logic K

$$\neg \forall R_{\blacksquare} \forall A_{\blacksquare} (\text{valid}(\Box_R A \Rightarrow A))$$

- ... and applies them

$$((M\,(R\,W))\,s^{A,W,R}) \neq ((N\,(R\,W))\,s^{A,W,R}) \\ \vee \\ (O\,W) \neq (P\,W)$$

- such flex-flex unification constraints are always solvable!
- total proving time 9.0s

Limitations

LEO-II cannot prove the following example:

Modal logic $K4$ (which adds only axiom 4 to K) is not entailed in K:

$$\neg\forall R_{\blacksquare}\forall B_{\blacksquare}(\text{valid}(\Box_R B \Rightarrow \Box_R \Box_R B))$$

Limitations

LEO-II also cannot prove this related example:

$$\neg \forall R. \mathrm{trans}(R)$$

- reason: not a theorem; domain of possible worlds may well just consist of a single world w.
- LEO-II can in fact prove the latter example under the additional assumption

$$\neg \forall X. \forall Y. X = Y$$

Limitations

LEO-II also cannot prove this related example:

$$\neg \forall R. \mathrm{trans}(R)$$

- reason: not a theorem; domain of possible worlds may well just consist of a single world w.
- LEO-II can in fact prove the latter example under the additional assumption

$$\neg \forall X. \forall Y. X = Y$$

Limitations

LEO-II also cannot prove this related example:

$$\neg\forall R\text{.}\mathrm{trans}(R)$$

- reason: not a theorem; domain of possible worlds may well just consist of a single world w.
- LEO-II can in fact prove the latter example under the additional assumption

$$\neg\forall X\text{.}\forall Y\text{.}X = Y$$