Automating Access Control Logics and Multimodal Logics in The Higher-Order Theorem Prover LEO-II[1]

Christoph E. Benzmüller

Kestrel Institute, Palo Alto, CA, October 29, 2008

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

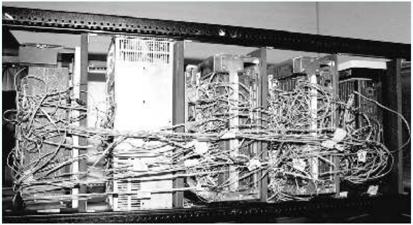
Overview

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
   - Ongoing and Future Work

Church's Simple Type Theory

Some folks 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

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

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

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

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

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

Sets and Relations in HOL

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

$$A \cup B \;:=\; (\lambda x_{\text{.}} x \in A \vee x \in B)$$

$$\cup \;:=\; \lambda A_{\text{.}} \lambda B_{\text{.}} (\lambda x_{\text{.}} x \in A \vee x \in B)$$

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

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

Sets and Relations in HOL

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

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

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

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

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

Sets and Relations in HOL

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

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

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

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

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

Sets and Relations in HOL

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

$$A \cup B := (\lambda x_{\blacksquare} x \in A \vee x \in B)$$

$$\cup := \lambda A_{\blacksquare} \lambda B_{\blacksquare} (\lambda x_{\blacksquare} x \in A \vee x \in B)$$

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

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

Sets and Relations in HOL

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

Types are Needed

Typed Sets and Relations in HOL

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

Polymorphism is a Challenge for Automation

- Another source of indeterminism / blind guessing

[TPHOLs-WP-07]

Automation is a Challenge

Automation of HOL: A Nightmare?

Undecidable and Infinitary Unification

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

~~(1)~~ $F \leftarrow \lambda y_i\text{.}\, y$
~~(2)~~ $F \leftarrow \lambda y_i\text{.}\, g(y)$
~~(3)~~ $F \leftarrow \lambda y_i\text{.}\, g(g(y))$
(4) ...

Automation of HOL: A Nightmare?

Primitive Substitution

Example Theorem: $\exists S. \mathit{reflexive}(S)$

Negation and Expansion of Definitions:

$$\neg \exists S. (\forall x_\iota. S(x, x))$$

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

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

Guess some suitable instances for S

$S \leftarrow \lambda y. \lambda z. V(y, z) \vee W(y, z)$ –

$S \leftarrow \lambda y. \lambda z. \top$ $\surd$

$S \leftarrow \lambda y. \lambda z. V(y, z) = W(y, z)$ $\surd$

$S \leftarrow \ldots$

Automation of HOL: A Nightmare?

Cut rule

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

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

[IJCAR-06]: 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 – A Cooperative Automatic Higher-Order Theorem Prover

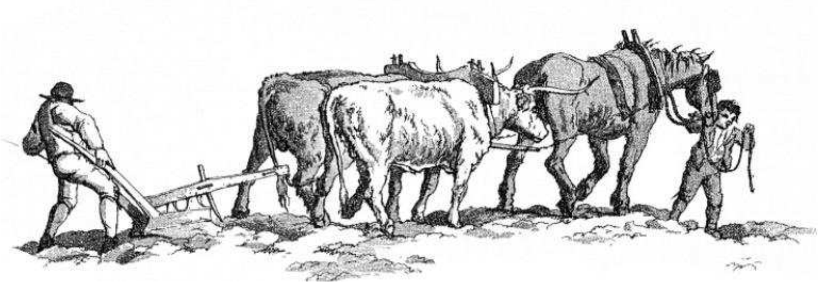
OMEGA GROUP

LEO-II

An Effective Higher-Order Theorem Prover

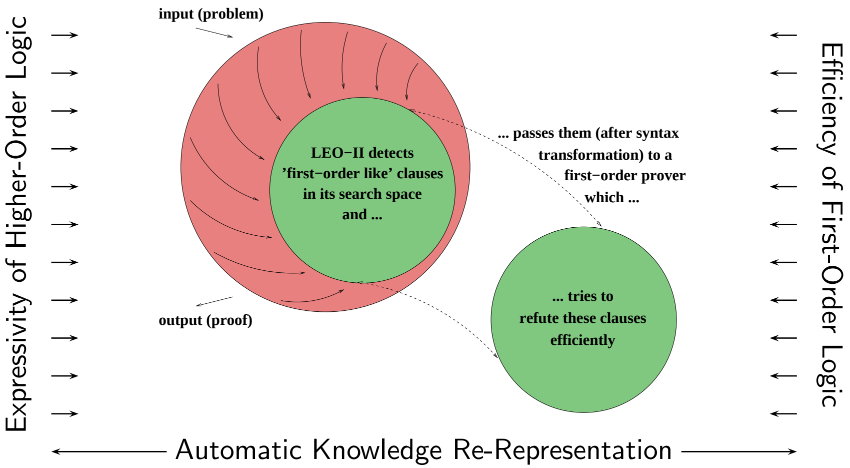
UNIVERSITY OF CAMBRIDGE

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:

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

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

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

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

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

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

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

Proof Goal:

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

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:

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

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

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

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

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

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

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

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:

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

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

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

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

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

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

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

Proof Goal:

$\forall B, C, D_{\blacksquare}$

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

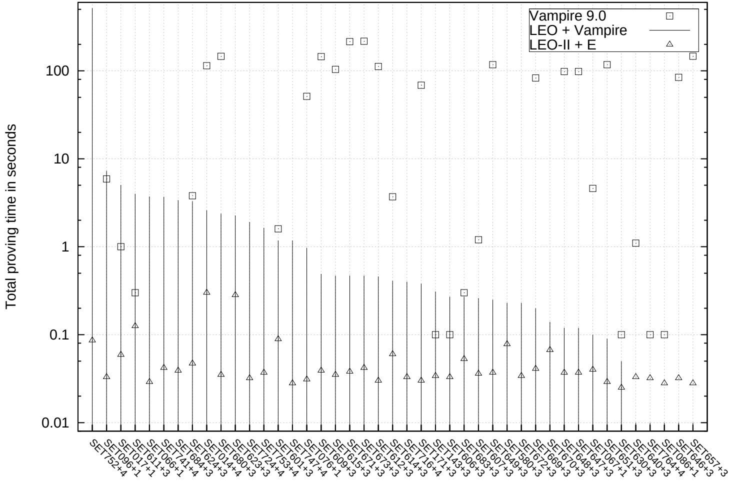
Performance: FO-ATPs

```
% 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!)
```

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

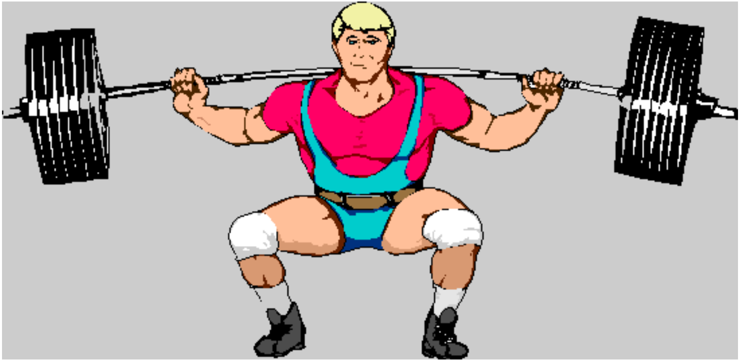
Results

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

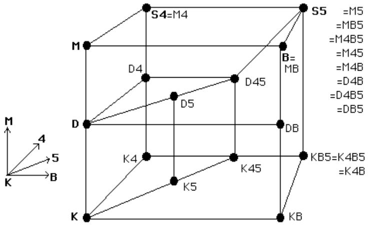
Less Lightweight Problems



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

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

Theorems:

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

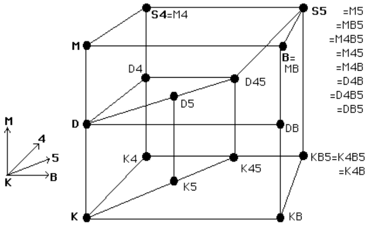
$$(M \wedge 4) \Leftrightarrow (\mathit{refl.}(R) \wedge \mathit{trans.}(R)) \qquad (2)$$

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

$$
\begin{array}{rcll}
S4 & = & K & \\
 & + & M(T): & \Box a \Rightarrow a \\
 & + & 4: & \Box a \Rightarrow \Box \Box a
\end{array}
$$

Theorems:

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

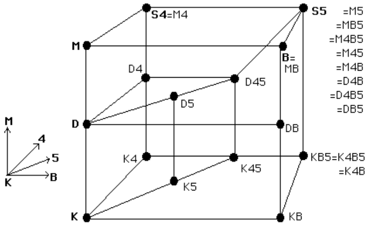
$$(M \wedge 4) \Leftrightarrow (refl.(R) \wedge trans.(R)) \qquad (2)$$

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

$$
\begin{array}{rcll}
S4 & = & K & \\
 & + & M(T): & \Box a \Rightarrow a \\
 & + & 4: & \Box a \Rightarrow \Box\Box a
\end{array}
$$

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

(Normal) Multimodal Logic in HOL

$$s, t ::= p | \neg\, s | s \vee t | \Box_r\, s$$

Simple, Straightforward Encoding

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

$$\begin{aligned}
\lfloor p \rfloor &= p_{\iota \rightarrow o} \\
\lfloor r \rfloor &= r_{\iota \rightarrow \iota \rightarrow o} \\
\lfloor \neg\, s \rfloor &= \lambda w_\iota {\scriptstyle\blacksquare} \neg(\lfloor s \rfloor\, w) \\
\lfloor s \vee t \rfloor &= \lambda w_\iota {\scriptstyle\blacksquare} (\lfloor s \rfloor\, w) \vee (\lfloor t \rfloor\, w) \\
\lfloor \Box_r\, s \rfloor &= \lambda w_\iota {\scriptstyle\blacksquare} \forall y_\iota {\scriptstyle\blacksquare} (\lfloor r \rfloor\, w\, y) \Rightarrow (\lfloor s \rfloor\, y)
\end{aligned}$$

(Normal) Multimodal Logic in HOL

$$s, t ::= p \mid \neg\, s \mid s \vee t \mid \Box_r\, s$$

Simple, Straightforward Encoding

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

$$
\begin{array}{rcl}
\lfloor p \rfloor & = & p_{\iota\rightarrow o} \\
\lfloor r \rfloor & = & r_{\iota\rightarrow\iota\rightarrow o} \\
\lfloor \neg \rfloor & = & \lambda s_{\iota\rightarrow o}\mathbf{.}\, \lambda w_{\iota}\mathbf{.}\, \neg(a\, w) \\
\lfloor \vee \rfloor & = & \lambda s_{\iota\rightarrow o}\mathbf{.}\, \lambda t_{\iota\rightarrow o}\mathbf{.}\, \lambda w_{\iota}\mathbf{.}\, (s\, w) \vee (t\, w) \\
\lfloor \Box \rfloor & = & \lambda r_{\iota\rightarrow\iota\rightarrow o}\mathbf{.}\, \lambda s_{\iota\rightarrow o}\mathbf{.} \\
& & \qquad \lambda w_{\iota}\mathbf{.}\, \forall y_{\iota}\mathbf{.}\, (r\, w\, y) \Rightarrow (s\, y)
\end{array}
$$

(Normal) Multimodal Logic in HOL

Encoding of Validity

$$\texttt{valid}\, s_{\iota\to o} \;=\; (\forall w_\iota\centerdot(s\,w))$$

$$|\texttt{valid}| \;=\; \lambda s_{\iota\to o}\centerdot(\forall w_\iota\centerdot(s\,w))$$

Local Definition Expansion

$$\begin{aligned} |\texttt{valid}\,\Box_r\,\top| &= |\texttt{valid}|\,|\Box|\,|r|\,|\top| \\ &=^{\beta\eta} \forall w_\iota\centerdot\forall y_\iota\centerdot(r\,w\,y) \Rightarrow \top \end{aligned}$$

(Normal) Multimodal Logic in HOL

Encoding of Validity

$$\begin{aligned} \texttt{valid}\, s_{\iota\to o} &= (\forall w_\iota \centerdot (s\, w)) \\ \lfloor\texttt{valid}\rfloor &= \lambda s_{\iota\to o} \centerdot (\forall w_\iota \centerdot (s\, w)) \end{aligned}$$

Local Definition Expansion

$$\begin{aligned} \lfloor\texttt{valid}\ \Box_r\ \top\rfloor &= \lfloor\texttt{valid}\rfloor\, \lfloor\Box\rfloor\, \lfloor r\rfloor\, \lfloor\top\rfloor \\ &=^{\beta\eta} \forall w_\iota \centerdot \forall y_\iota \centerdot (r\, w\, y) \Rightarrow \top \end{aligned}$$

(Normal) Multimodal Logic in HOL

Encoding of Validity

$$\begin{aligned} \mathtt{valid}\, s_{\iota\to o} &= (\forall w_\iota . (s\, w)) \\ |\mathtt{valid}| &= \lambda s_{\iota\to o} . (\forall w_\iota . (s\, w)) \end{aligned}$$

Local Definition Expansion

$$\begin{aligned} |\mathtt{valid}\ \Box_r\ \top| &= |\mathtt{valid}|\, |\Box|\, |r|\, |\top| \\ &=^{\beta\eta} \forall w_\iota . \forall y_\iota . (r\, w\, y) \Rightarrow \top \end{aligned}$$

Even simpler: Reasoning within Multimodal Logics

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

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

Initialisation of problem

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

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 free 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: $|\texttt{valid}\ \Box_s (\Box_r a \Rightarrow \Box_r a)|$

Initialisation of problem

$$\neg|\texttt{valid}\ \Box_s (\Box_r a \Rightarrow \Box_r a)|$$

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 free variable)

$$s(x,y) \qquad \neg a(u)$$
$$r(y,u) \qquad a(V) \vee \neg r(y,V)$$

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

$$@_{(io)\_i}(@_{(i(io))\_i}(s,x),y) \qquad \neg @_{(\iota o)\_\iota}(a,u)$$
$$@_{(io)\_i}(@_{(i(io))\_i}(r,y),u) \qquad @_{(\iota o)\_\iota}(a,V) \vee \neg @_{(io)\_i}(@_{(i(io))\_i}(r,y),V)$$

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

Initialisation of problem

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

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 free 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: $|\mathtt{valid}\ \Box_s (\Box_r a \Rightarrow \Box_r a)|$

Initialisation of problem

$$\neg|\mathtt{valid}\ \Box_s (\Box_r a \Rightarrow \Box_r a)|$$

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 free 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]

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

More Examples ...

A simple equation between modal logic formulas

$$\forall r_{\blacksquare} \forall a_{\blacksquare} \forall b_{\blacksquare} \lfloor \Box_r (a \vee b) \rfloor \doteq \lfloor \Box_r (b \vee a) \rfloor$$

where $\doteq$ is defined as $\lambda u, v. \forall p_{\blacksquare} (p\, u) \Rightarrow (p\, v)$

- initialisation, definition expansion and normalisation:

$$(p\,(\lambda W_\iota . \forall y_{\iota\blacksquare} \neg((r\,W)\,y) \vee (a\,y) \vee (b\,y)))$$
$$\neg(p\,(\lambda W_\iota . \forall y_{\iota\blacksquare} \neg((r\,W)\,y) \vee (b\,y) \vee (a\,y)))$$

More Examples . . .

A simple equation between modal logic formulas

$$\forall r. \forall a. \forall b. |\Box_r (a \vee b)| \doteq |\Box_r (b \vee a)|$$

where $\doteq$ is defined as $\lambda u, v. \forall p. (p\, u) \Rightarrow (p\, v)$

- resolution:

$$(p\,(\lambda W_\iota . \forall y_\iota . \neg((r\, W)\, y) \vee (a\, y) \vee (b\, y)))$$
$$\neq$$
$$(p\,(\lambda W_\iota . \forall y_\iota . \neg((r\, W)\, y) \vee (b\, y) \vee (a\, y)))$$

More Examples ...

A simple equation between modal logic formulas

$$\forall r_{\blacksquare} \forall a_{\blacksquare} \forall b_{\blacksquare} |\Box_r (a \vee b)| \doteq |\Box_r (b \vee a)|$$

where $\doteq$ is defined as $\lambda u, v. \forall p_{\blacksquare} (p\, u) \Rightarrow (p\, v)$

- decomposition:

$$(\lambda W_\iota . \forall y_{\iota\blacksquare} \neg((r\, W)\, y) \vee (a\, y) \vee (b\, y))$$
$$\neq$$
$$(\lambda W_\iota . \forall y_{\iota\blacksquare} \neg((r\, W)\, y) \vee (b\, y) \vee (a\, y))$$

More Examples ...

A simple equation between modal logic formulas

$$\forall r_{\blacksquare} \forall a_{\blacksquare} \forall b_{\blacksquare} \lfloor \Box_r (a \vee b) \rfloor \doteq \lfloor \Box_r (b \vee a) \rfloor$$

where $\doteq$ is defined as $\lambda u, v. \forall p_{\blacksquare} (p\, u) \Rightarrow (p\, v)$

- functional extensionality:

$$(\forall y_{\iota} \blacksquare \neg((r\, w)\, y) \vee (a\, y) \vee (b\, y))$$
$$\neq$$
$$(\forall y_{\iota} \blacksquare \neg((r\, w)\, y) \vee (b\, y) \vee (a\, y))$$

More Examples ...

A simple equation between modal logic formulas

$$\forall r_\blacksquare \forall a_\blacksquare \forall b_\blacksquare |\Box_r (a \vee b)| \doteq |\Box_r (b \vee a)|$$

where $\doteq$ is defined as $\lambda u, v. \forall p_\blacksquare (p\, u) \Rightarrow (p\, v)$

- Boolean extensionality:

$$\neg((\forall y_{\iota\blacksquare} \neg((r\, w)\, y) \vee (a\, y) \vee (b\, y))$$
$$\Leftrightarrow$$
$$(\forall y_{\iota\blacksquare} \neg((r\, w)\, y) \vee (b\, y) \vee (a\, y)))$$

More Examples ...

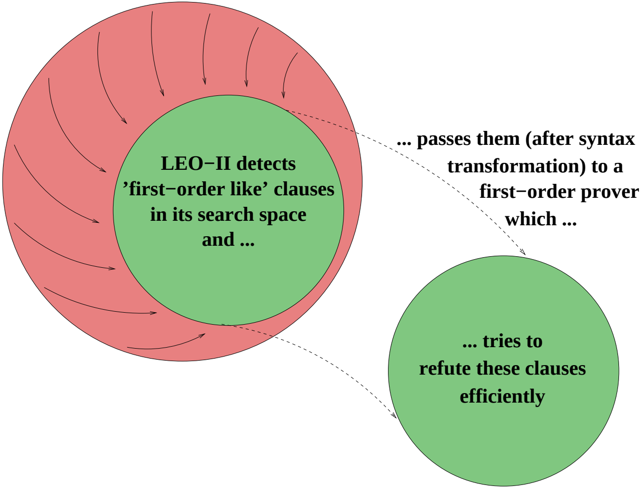
A simple equation between modal logic formulas

$$\forall r_\blacksquare \forall a_\blacksquare \forall b_\blacksquare |\Box_r (a \vee b)| \doteq |\Box_r (b \vee a)|$$

where $\doteq$ is defined as $\lambda u, v. \forall p_\blacksquare (p\, u) \Rightarrow (p\, v)$

- 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 is $0.166s$

Architecture of LEO-II



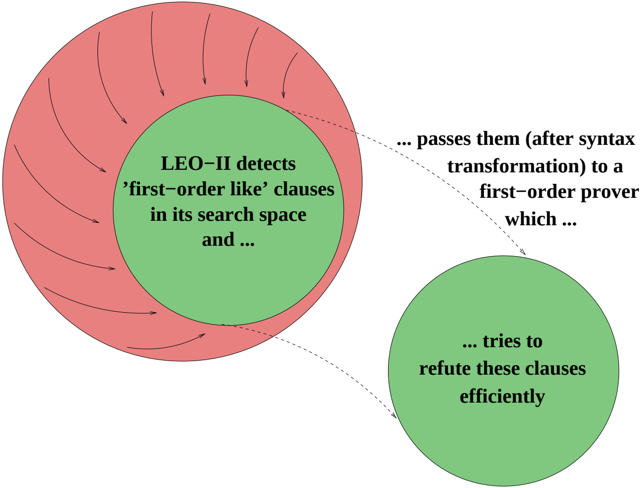
More Examples ...

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

$$\forall r. (\forall a. \lfloor \texttt{valid}\ \Box_r\ a \Rightarrow a \rfloor \land \lfloor \texttt{valid}\ \Box_r\ a \Rightarrow \Box_r\ \Box_r\ a \rfloor) \Leftrightarrow ((\texttt{reflexive}\ r) \land (\texttt{transitive}\ r))$$

- 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



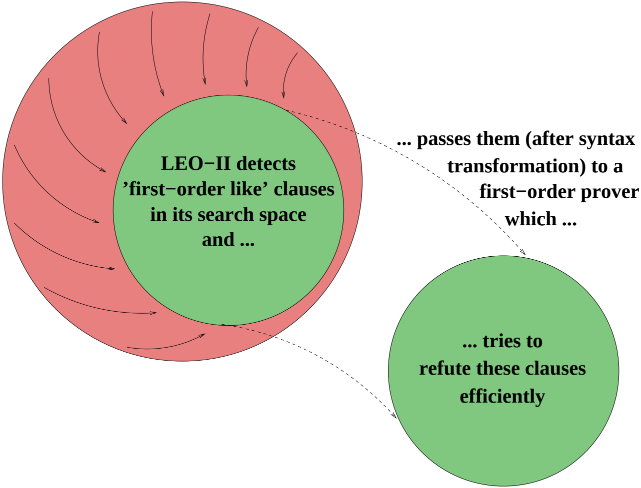
More Examples ...

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

$$\neg \forall r. \forall a. \forall b. \lfloor \mathtt{valid}\ \Box_r\ a \Rightarrow a \rfloor \wedge \lfloor \mathtt{valid}\ \Box_r\ b \Rightarrow \Box_r\ \Box_r\ b \rfloor$$

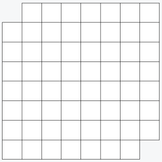
- LEO-II shows that axiom T is not valid
- R is instantiated with $\lambda x. \lambda y. (H\, x\, y) \neq (H'\, x\, y)$ via primitive substitution
- total proving time 17.3s

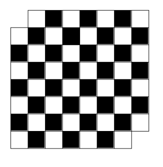
Architecture of LEO-II



Representation (and the right System Architecture) Matters!

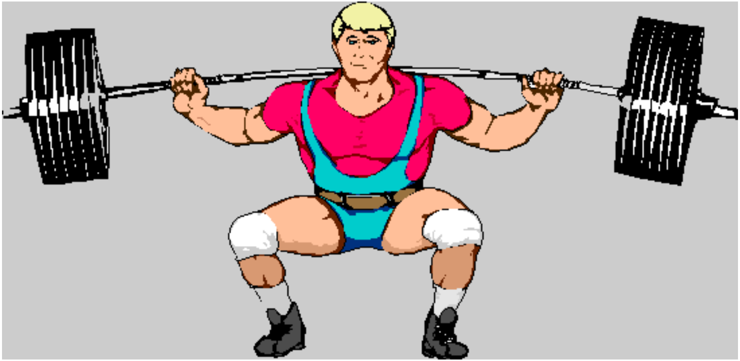
A general lesson in AI . . .





. . . and a specific lesson here

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



Access Control Logics

Access Control Logic

Example (from [GargAbadi08]): file-access scenario

- If `admin` says that `file1` should be deleted, then this must be the case.

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

- `admin` trusts Bob to decide whether `file1` should be deleted.

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

- Bob wants to delete `file1`.

  $\texttt{Bob}\,\mathsf{says}\,\texttt{deletefile1}$

- Is deletion permitted?

  $\texttt{deletefile1}$

Access Control Logic

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})

So, let’s combine this with our previous work . . . and apply LEO-II

Access Control Logic

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})

So, let's combine this with our previous work . . . and apply LEO-II

Access Control Logic translated to Modal Logic and HOL

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

Translation $\lceil . \rceil$ (of Garg and Abadi) into S4

$$\begin{array}{rcl}
\lceil p \rceil & = & \Box p \\
\lceil s \wedge t \rceil & = & \lceil s \rceil \wedge \lceil t \rceil \\
\lceil s \vee t \rceil & = & \lceil s \rceil \vee \lceil t \rceil \\
\lceil s \supset t \rceil & = & \Box(\lceil s \rceil \Rightarrow \lceil t \rceil)\texttt{(fixed on 05.11.2008)} \\
\lceil \top \rceil & = & \top \\
\lceil \bot \rceil & = & \bot \\
\lceil A \,\texttt{says}\, s \rceil & = & \Box(A \vee \lceil s \rceil)
\end{array}$$

Access Control Logic translated to Modal Logic and HOL

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

Translation $\|.\|$ to HOL

$$
\begin{array}{lcl}
 & & |r| \quad \text{(we fix one single } r\text{!!!)} \\
\|p\| & = & |\Box_r\, p| \\
\|A\| & = & |A| \\
\|\wedge\| & = & \lambda s.\, \lambda t.\, |s \wedge t| \\
\|\vee\| & = & \lambda s.\, \lambda t.\, |s \vee t| \\
\|\supset\| & = & \lambda s.\, \lambda t.\, |\Box_r\, (s \Rightarrow t)| \text{ (fixed on 05.11.2008)} \\
\|\top\| & = & |\top| \\
\|\bot\| & = & |\bot| \\
\|\texttt{says}\| & = & \lambda A.\, \lambda s.\, |\Box_r\, (A \vee s)|
\end{array}
$$

Access Control Logic translated to Modal Logic and HOL

$$s, t ::= p \mid s \wedge t \mid s \vee t \mid s \supset t \mid \bot \mid \top \mid A \ \mathtt{says}\ s$$

Translation $\|.\|$ to HOL

$$
\begin{array}{lcl}
 & & r_{\iota\to\iota\to o} \quad \text{(we fix one single } r\text{!!!)} \\
\|p\| & = & \lambda x_\iota\text{.}\forall y_\iota\text{.} r_{\iota\to\iota\to o}\, x\, y \Rightarrow p_{\iota\to o}\, Y \\
\|A\| & = & a_{\iota\to o} \quad (\texttt{(distinct from the } p_{\iota\to o}) \\
\|\wedge\| & = & \lambda s_{\iota\to o}\text{.}\lambda t_{\iota\to o}\text{.}\lambda w_\iota\text{.} s\, w \wedge t\, w \\
\|\vee\| & = & \lambda s_{\iota\to o}\text{.}\lambda t_{\iota\to o}\text{.}\lambda w_\iota\text{.} s\, w \vee t\, w \\
\|\supset\| & = & \lambda s_{\iota\to o}\text{.}\lambda t_{\iota\to o}\text{.}\lambda w_\iota\text{.}\forall y_\iota\text{.} r_{\iota\to\iota\to o}\, w\, y \Rightarrow (s\, y \Rightarrow t\, y) \ \text{(fixed on} \\
\|\top\| & = & \lambda s_{\iota\to o}\text{.} \top \\
\|\bot\| & = & \lambda s_{\iota\to o}\text{.} \bot \\
\|\mathtt{says}\| & = & \lambda A_{\iota\to o}\text{.}\lambda s_{\iota\to o}\text{.}\lambda w_\iota\text{.}\forall y_\iota\text{.} r_{\iota\to\iota\to o}\, w\, y \Rightarrow (A\, y \vee s\, y)
\end{array}
$$

Access Control Logic translated to Modal Logic and HOL

Notion of Validity

$$\texttt{iclval} = \texttt{valid}$$

Addition of Modal Logic Axioms for S4

$$\forall p_{\iota\to o}.\lfloor \texttt{valid}\ \Box_r\ p \Rightarrow p \rfloor$$

$$\forall p_{\iota\to o}.\lfloor \texttt{valid}\ \Box_r\ p \Rightarrow \Box_r\ \Box_r\ p \rfloor$$

Soundness of Embedding

see [SR-2008-01]: employs transformation from Kripke models into corresponding Henkin models

Access Control Logic translated to Modal Logic and HOL

Notion of Validity

$$\texttt{iclval} = \texttt{valid}$$

Addition of Modal Logic Axioms for S4

$$\forall p_{\iota \to o}. \lfloor \texttt{valid}\ \Box_r\, p \Rightarrow p \rfloor$$

$$\forall p_{\iota \to o}. \lfloor \texttt{valid}\ \Box_r\, p \Rightarrow \Box_r\, \Box_r\, p \rfloor$$

Soundness of Embedding

see [SR-2008-01]: employs transformation from Kripke models into corresponding Henkin models

Access Control Logic translated to Modal Logic and HOL

Notion of Validity

$$\texttt{iclval} = \texttt{valid}$$

Addition of Modal Logic Axioms for S4

$$\forall p_{\iota \to o}. \lfloor \texttt{valid}\ \Box_r\ p \Rightarrow p \rfloor$$

$$\forall p_{\iota \to o}. \lfloor \texttt{valid}\ \Box_r\ p \Rightarrow \Box_r\ \Box_r\ p \rfloor$$

Soundness of Embedding

see [SR-2008-01]: employs transformation from Kripke models into corresponding Henkin models

Access Control Logic translated to Modal Logic and HOL

Example (from [GargAbadi08]): file-access scenario

- If `admin` says that `file1` should be deleted, then this must be the case.

  $\|\mathtt{iclval}\ (\mathtt{admin\,says\,deletefile1}) \supset \mathtt{deletefile1}\|$

- `admin` trusts Bob to decide whether `file1` should be deleted.

  $\|\mathtt{iclval\ admin\,says}\,((\mathtt{Bob\,says\,deletefile1}) \supset \mathtt{deletefile1})\|$

- Bob wants to delete `file1`.

  $\|\mathtt{iclval\ Bob\,says\,deletefile1}\|$

- Is deletion permittted?

  $\|\mathtt{iclval\ deletefile1}\|$

Access Control Logic translated to Modal Logic and HOL

Example (from [GargAbadi08]): file-access scenario

- If `admin` says that `file1` should be deleted, then this must be the case.

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

- `admin` trusts Bob to decide whether `file1` should be deleted.

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

- Bob wants to delete `file1`.

  $$|\texttt{valid}\ \Box_r\,(\texttt{Bob} \vee \Box_r\,\texttt{deletefile1})|$$

- Is deletion permittted?

  $$\|\texttt{iclval}\ \texttt{deletefile1}\|$$

Access Control Logic translated to Modal Logic and HOL

Example (from [GargAbadi08]): file-access scenario

- If `admin` says that `file1` should be deleted, then this must be the case.

  $$\|\texttt{iclval (admin says deletefile1)} \supset \texttt{deletefile1}\|$$

- `admin` trusts Bob to decide whether `file1` should be deleted.

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

- Bob wants to delete `file1`.

  $$\forall w_\iota . (\forall y_\iota . (r\ w\ y) \Rightarrow ((Bob\ y) \vee \forall u_\iota . (r\ w\ u) \Rightarrow (deletefile1\ \ u)))$$

- Is deletion permittted?

  $$\|\texttt{iclval deletefile1}\|$$

LEO-II: 0.149 seconds

Logic ICL:

Name	Problem	LEO (s)
unit	$\{\mathtt{R},\mathtt{T}\} \vdash \Vert\mathtt{ICLval}\ s \supset (A\,\mathsf{says}\,s)\Vert$	0.048
cuc	$\{\mathtt{R},\mathtt{T}\} \vdash \Vert\mathtt{ICLval}\ (A\,\mathsf{says}\,(s \supset t)) \supset (A\,\mathsf{says}\,s) \supset (A\,\mathsf{says}\,t)\Vert$	0.055
idem	$\{\mathtt{R},\mathtt{T}\} \vdash \Vert\mathtt{ICLval}\ (A\,\mathsf{says}\,A\,\mathsf{says}\,s) \supset (A\,\mathsf{says}\,s)\Vert$	0.048
Ex1	$\{\mathtt{R},\mathtt{T},\Vert\mathtt{ICLval}\ (1.1)\Vert,\ldots,\Vert\mathtt{ICLval}\ (1.3)\Vert\} \vdash \Vert\mathtt{ICLval}\ (1.4)\Vert$	0.149
unit^K	$\vdash \Vert\mathtt{ICLval}\ s \supset (A\,\mathsf{says}\,s)\Vert$	–
cuc^K	$\vdash \Vert\mathtt{ICLval}\ (A\,\mathsf{says}\,(s \supset t)) \supset (A\,\mathsf{says}\,s) \supset (A\,\mathsf{says}\,t)\Vert$	0.041
idem^K	$\vdash \Vert\mathtt{ICLval}\ (A\,\mathsf{says}\,A\,\mathsf{says}\,s) \supset (A\,\mathsf{says}\,s)\Vert$	–
$\mathsf{Ex1}^K$	$\{\Vert\mathtt{ICLval}\ (1.1)\Vert,\ldots,\Vert\mathtt{ICLval}\ (1.3)\Vert\} \vdash \Vert\mathtt{ICLval}\ (1.4)\Vert$	0.053

R, T: reflexivity and transitivity axioms as seen before

Exp.: Access Control Logic in HOL

Logic ICL$^{\Rightarrow}$:

Name	Problem	LEO (s)
refl	$\{\mathtt{R},\mathtt{T}\} \vdash \lVert\mathtt{ICLval}\ A \Rightarrow A\rVert$	0.052
trans	$\{\mathtt{R},\mathtt{T}\} \vdash \lVert\mathtt{ICLval}\ (A \Rightarrow B) \supset (B \Rightarrow C) \supset (A \Rightarrow C)\rVert$	0.044
sp.-for	$\{\mathtt{R},\mathtt{T}\} \vdash \lVert\mathtt{ICLval}\ (A \Rightarrow B) \supset (A\,\mathsf{says}\,s) \supset (B\,\mathsf{says}\,s)\rVert$	0.052
handoff	$\{\mathtt{R},\mathtt{T}\} \vdash \lVert\mathtt{ICLval}\ (B\,\mathsf{says}\,(A \Rightarrow B)) \supset (A \Rightarrow B)\rVert$	0.044
Ex2	$\{\mathtt{R},\mathtt{T},\lVert\mathtt{ICLval}\ (2.1)\rVert,\ldots,\lVert\mathtt{ICLval}\ (2.4)\rVert\} \vdash \lVert\mathtt{ICLval}\ (2.5)\rVert$	0.251
refl^K	$\vdash \lVert\mathtt{ICLval}\ A \Rightarrow A\rVert$	0.034
trans^K	$\vdash \lVert\mathtt{ICLval}\ (A \Rightarrow B) \supset (B \Rightarrow C) \supset (A \Rightarrow C)\rVert$	0.043
sp.-for^K	$\vdash \lVert\mathtt{ICLval}\ (A \Rightarrow B) \supset (A\,\mathsf{says}\,s) \supset (B\,\mathsf{says}\,s)\rVert$	0.039
handoff^K	$\vdash \lVert\mathtt{ICLval}\ (B\,\mathsf{says}\,(A \Rightarrow B)) \supset (A \Rightarrow B)\rVert$	–
Ex2^K	$\{\lVert\mathtt{ICLval}\ (2.1)\rVert,\ldots,\lVert\mathtt{ICLval}\ (2.4)\rVert\} \vdash \lVert\mathtt{ICLval}\ (2.5)\rVert$	–

R, T: reflexivity and transitivity axioms as seen before

Exp.: Access Control Logic in HOL

Logic ICL^B:

Name	Problem	LEO (s)
trust	$\{\mathtt{R},\mathtt{T}\} \vdash \lVert \mathtt{ICLval}\ (\bot\ \mathsf{says}\ s) \supset s \rVert$	0.044
untrust	$\{\mathtt{R},\mathtt{T}, \lVert \mathtt{ICLval}\ A \equiv \top \rVert\} \vdash \lVert \mathtt{ICLval}\ A\ \mathsf{says}\ \bot \rVert$	0.046
cuc'	$\{\mathtt{R},\mathtt{T}\} \vdash \lVert \mathtt{ICLval}\ ((A \supset B)\ \mathsf{says}\ s) \supset (A\ \mathsf{says}\ s) \supset (B\ \mathsf{says}\ s) \rVert$	0.048
Ex3	$\{\mathtt{R},\mathtt{T}, \lVert \mathtt{ICLval}\ (3.1) \rVert, \ldots, \lVert \mathtt{ICLval}\ (3.3) \rVert\} \vdash \lVert \mathtt{ICLval}\ (3.4) \rVert$	0.060
trust^K	$\vdash \lVert \mathtt{ICLval}\ (\bot\ \mathsf{says}\ s) \supset s \rVert$	–
untrust^K	$\{\lVert \mathtt{ICLval}\ A \equiv \top \rVert\} \vdash \lVert \mathtt{ICLval}\ A\ \mathsf{says}\ \bot \rVert$	0.035
cuc'^K	$\vdash \lVert \mathtt{ICLval}\ ((A \supset B)\ \mathsf{says}\ s) \supset (A\ \mathsf{says}\ s) \supset (B\ \mathsf{says}\ s) \rVert$	0.044
Ex3^K	$\{\lVert \mathtt{ICLval}\ (3.1) \rVert, \ldots, \lVert \mathtt{ICLval}\ (3.3) \rVert\} \vdash \lVert \mathtt{ICLval}\ (3.4) \rVert$	–

R, T: reflexivity and transitivity axioms as seen before

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

- Monadic Second-Order Logic, Prop. Logic, Arithmetic, ...
- 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

Logic System Interrelationships, Ontology Reasoning (SUMO, CYC), Formal Methods, CL, ...

... there is much left to be done!

LEO-II

- Equational Reasoning
- Termindexing
- Handling of Definitions

Cooperat. with Specialist Reasoners

- Monadic Second-Order Logic, Prop. Logic, Arithmetic, . . .
- 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

Logic System Interrelationships, Ontology Reasoning (SUMO, CYC), Formal Methods, CL, . . .

... there is much left to be done!

LEO-II

- Equational Reasoning
- Termindexing
- Handling of Definitions

Cooperat. with Specialist Reasoners

- Monadic Second-Order Logic, Prop. Logic, Arithmetic, ...
- 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

Logic System Interrelationships, Ontology Reasoning (SUMO, CYC), Formal Methods, CL, ...

… there is much left to be done!

LEO-II

- Equational Reasoning
- Termindexing
- Handling of Definitions

Cooperat. with Specialist Reasoners

- Monadic Second-Order Logic, Prop. Logic, Arithmetic, . . .
- 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

Logic System Interrelationships, Ontology Reasoning (SUMO, CYC), Formal Methods, CL, . . .

... there is much left to be done!

LEO-II

- Equational Reasoning
- Termindexing
- Handling of Definitions

Cooperat. with Specialist Reasoners

- Monadic Second-Order Logic, Prop. Logic, Arithmetic, ...
- 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

Logic System Interrelationships, Ontology Reasoning (SUMO, CYC), Formal Methods, CL, ...

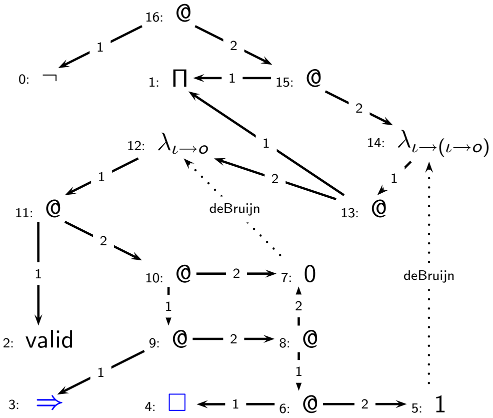
More Information on LEO-II

- 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: $\neg\forall R.\forall A.(\mathbf{valid}(\Box_R\, A \Rightarrow A))$

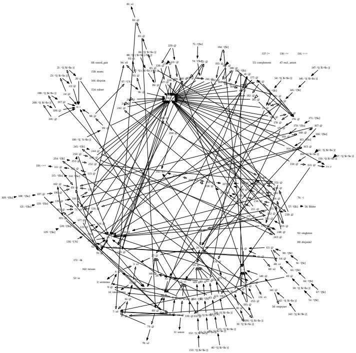


Term graph videos: `http://www.ags.uni-sb.de/~leo/art`

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

Limitations

LEO-II cannot prove the following example:

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

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

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:

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

- 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

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