Specification of an Automatic Prover (P3)

J.R. Abrial*, D. Cansell†, Ch. Métayer‡

*jrabrial AT neuf.fr

†dominique.cansel AT loria.fr

‡christophe.metayer AT systerel.fr

Why it is an Interesting Subject

- In the literature, one can see specifications for:
  - programming languages
  - compilers
  - operating systems (rarely)
  - protocols
  - safety critical systems
  - . . .
- One never sees specifications for provers
- The proposed specification is made by successive refinements

A Series of Embedded Provers (1)

A Series of Embedded Provers (1)

- a propositional calculus prover

A Series of Embedded Provers (1)

- a propositional calculus prover

- a first order predicate calculus prover

A Series of Embedded Provers (1)

- a propositional calculus prover

- a first order predicate calculus prover

- an equality prover

A Series of Embedded Provers (1)

- a propositional calculus prover

- a first order predicate calculus prover

- an equality prover

- a set theory prover

A Series of Embedded Provers (1)

- a propositional calculus prover

- a first order predicate calculus prover

- an equality prover

- a set theory prover

- an arithmetic prover (not presented here)

A Series of Embedded Provers (2)

- All such provers are important within a formal method platform

- The Rodin Platform for Event-B: event-b.org

- P3 is not as general as HOL, Isabelle, COQ, ...

- The logics of P3 (above) are all built in

Examples: Propositional Calculus

$$(P \Rightarrow Q) \land (\neg R \Rightarrow \neg Q) \Rightarrow (P \Rightarrow R)$$

Examples: Predicate Calculus

$$(P \Rightarrow Q) \land (\neg R \Rightarrow \neg Q) \Rightarrow (P \Rightarrow R)$$

$$\begin{array}{l}
\quad \forall x, y \cdot P(x) \land Q(y) \Rightarrow (\exists z \cdot R(z) \land S(x, y, z)) \\
\quad \forall x \cdot Q(x) \lor R(x) \\
\quad P(a) \\
\quad \forall y \cdot R(y) \Rightarrow (\exists z \cdot Q(z) \land S(a, y, z)) \\
\Rightarrow \\
\quad \exists x \cdot \forall y \cdot \exists z \cdot S(x, y, z)
\end{array}$$

Examples: Equality

$$(P \Rightarrow Q) \land (\neg R \Rightarrow \neg Q) \Rightarrow (P \Rightarrow R)$$

$$\begin{array}{l}
\quad \forall x, y \cdot P(x) \land Q(y) \Rightarrow (\exists z \cdot R(z) \land S(x, y, z)) \\
\quad \forall x \cdot Q(x) \lor R(x) \\
\quad P(a) \\
\quad \forall y \cdot R(y) \Rightarrow (\exists z \cdot Q(z) \land S(a, y, z)) \\
\Rightarrow \\
\quad \exists x \cdot \forall y \cdot \exists z \cdot S(x, y, z)
\end{array}$$

$$\begin{array}{l}
\quad \forall x \cdot P(x) \land Q(x) \Rightarrow x = a \lor x = b \\
\quad \neg R(a) \\
\quad \forall x \cdot Q(x) \land R(x) \Rightarrow P(x) \\
\Rightarrow \\
\quad \forall x \cdot Q(x) \land R(x) \Rightarrow x = b
\end{array}$$

$$(P \Rightarrow Q) \wedge (\neg R \Rightarrow \neg Q) \Rightarrow (P \Rightarrow R)$$

$$\begin{array}{l} \quad \forall x, y \cdot P(x) \wedge Q(y) \Rightarrow (\exists z \cdot R(z) \wedge S(x, y, z)) \\ \quad \forall x \cdot Q(x) \vee R(x) \\ \quad P(a) \\ \quad \forall y \cdot R(y) \Rightarrow (\exists z \cdot Q(z) \wedge S(a, y, z)) \\ \Rightarrow \\ \quad \exists x \cdot \forall y \cdot \exists z \cdot S(x, y, z) \end{array}$$

$$\begin{array}{l} \quad \forall x \cdot P(x) \wedge Q(x) \Rightarrow x = a \vee x = b \\ \quad \neg R(a) \\ \quad \forall x \cdot Q(x) \wedge R(x) \Rightarrow P(x) \\ \Rightarrow \\ \quad \forall x \cdot Q(x) \wedge R(x) \Rightarrow x = b \end{array}$$

$$\begin{array}{l} \quad p \in U \leftrightarrow S \\ \quad q \in U \leftrightarrow S \\ \quad f \in S \rightarrowtail T \\ \quad p \,;\, f = q \,;\, f \\ \Rightarrow \\ \quad p = q \end{array}$$

Price of Interactive Proofs in Industrial Projects

Price of Interactive Proofs in Industrial Projects

- n: number of lines of generated code

Price of Interactive Proofs in Industrial Projects

- n: number of lines of generated code

- f: proof factor. Typical values are 2 or 3.

  n/f is the number of proofs generated

- n: number of lines of generated code
- f: proof factor. Typical values are 2 or 3.

  n/f is the number of proofs generated
- x: percentage of interactive proofs. Typical values are 2, 5, 10.

  $n.x/100.f$ is the number of interactive proofs generated

Price of Interactive Proofs in Industrial Projects

- n: number of lines of generated code
- f: proof factor. Typical values are 2 or 3.

  n/f is the number of proofs generated
- x: percentage of interactive proofs. Typical values are 2, 5, 10.

  $n.x/100.f$ is the number of interactive proofs generated
- p: number of interactive proofs per man-day. Typical value is 20.

  $n.x/100.f.p$ is the number of man-day for the interactive proofs

- n: number of lines of generated code

- f: proof factor. Typical values are 2 or 3.

  n/f is the number of proofs generated

- x: percentage of interactive proofs. Typical values are 2, 5, 10.

  $n.x/100.f$ is the number of interactive proofs generated

- p: number of interactive proofs per man-day. Typical value is 20.

  $n.x/100.f.p$ is the number of man-day for the interactive proofs

- m: number of man-months to perform the interactive proofs.

  $n.x/100.f.p.20$ is the number of man-month for proving

- $m = n.x/100.f.p.20$ is the number of man-months needed for proving

n	$100,000$	$100,000$	$100,000$
f	2	2	2
x	2.5%	5%	10%
p	20	20	20
m	3.12	6.25	12.5

This shows the importance to prove as many automatic proofs as we can

Outline

- Propositional Calculus Prover

- Predicate Calculus Prover

- Equality Prover

- Set Theory prover

- Conclusion

The propositional calculus prover

Approach

- Transforming the predicate P into the sequent

$$\vdash \quad \neg P \Rightarrow \bot$$

- Applying inference rules of the forms

$$\frac{\ldots}{\mathbf{H} \vdash \neg\,(P \ \mathbf{op}\ Q) \Rightarrow R} \qquad \frac{\ldots}{\mathbf{H} \vdash (P \ \mathbf{op}\ Q) \Rightarrow R}$$

where **op** is one of $\wedge$, $\vee$, $\Rightarrow$, and $\Leftrightarrow$

- Applying some rewriting rules to finish up the proof

Propositional Calculus Prover Outline

- Syntax

- Inference rules

- Rewriting rules

- Example

Syntax for the Propositional Calculus Prover

$$
\begin{array}{lcl}
predicate & ::= & \top \\
 & & \bot \\
 & & \neg\, predicate \\
 & & predicate \ \wedge \ predicate \\
 & & predicate \ \vee \ predicate \\
 & & predicate \ \Rightarrow \ predicate \\
 & & predicate \ \Leftrightarrow \ predicate
\end{array}
$$

Priorities and parentheses can be used for managing ambiguities.

Inference Rules for the Propositional Calculus Prover (1)

$$\frac{\vdash \neg P \Rightarrow \bot}{P} \quad \textbf{INI1}$$

$$\frac{\vdash P_1 \Rightarrow (\ \ldots \Rightarrow (P_n \Rightarrow (\neg Q \Rightarrow \bot))\ldots)}{P_1 \wedge \ldots \wedge P_n \Rightarrow Q} \quad \textbf{INI2}$$

$$\frac{\mathbf{H} \vdash \neg Q \Rightarrow R \qquad \mathbf{H} \vdash \neg P \Rightarrow R}{\mathbf{H} \vdash \neg (P \wedge Q) \Rightarrow R} \quad \textbf{AND1} \qquad \frac{\mathbf{H} \vdash P \Rightarrow (Q \Rightarrow R)}{\mathbf{H} \vdash (P \wedge Q) \Rightarrow R} \quad \textbf{AND2}$$

$$\frac{\mathbf{H} \vdash \neg P \Rightarrow (\neg Q \Rightarrow R)}{\mathbf{H} \vdash \neg (P \vee Q) \Rightarrow R} \quad \textbf{OR1} \qquad \frac{\mathbf{H} \vdash P \Rightarrow R \qquad \mathbf{H} \vdash Q \Rightarrow R}{\mathbf{H} \vdash (P \vee Q) \Rightarrow R} \quad \textbf{OR2}$$

$$\frac{\mathbf{H} \vdash P \Rightarrow (\neg\, Q \Rightarrow R)}{\mathbf{H} \vdash \neg\,(P \Rightarrow Q) \Rightarrow R} \quad \textbf{IMP1} \qquad \frac{\mathbf{H} \vdash \neg\, P \Rightarrow R \qquad \mathbf{H} \vdash Q \Rightarrow R}{\mathbf{H} \vdash (P \Rightarrow Q) \Rightarrow R} \quad \textbf{IMP2}$$

$$\frac{\mathbf{H} \vdash P \Rightarrow (\neg\, Q \Rightarrow R) \qquad \mathbf{H} \vdash \neg\, P \Rightarrow (Q \Rightarrow R)}{\mathbf{H} \vdash \neg\,(P \Leftrightarrow Q) \Rightarrow R} \quad \textbf{EQV1}$$

$$\frac{\mathbf{H} \vdash P \Rightarrow (Q \Rightarrow R) \qquad \mathbf{H} \vdash \neg\, P \Rightarrow (\neg\, Q \Rightarrow R)}{\mathbf{H} \vdash (P \Leftrightarrow Q) \Rightarrow R} \quad \textbf{EQV2}$$

Inference Rules for the Propositional Calculus Prover (3)

$$\frac{\mathbf{H} \vdash P \Rightarrow R}{\mathbf{H} \vdash \neg\neg P \Rightarrow R} \quad \textbf{NOT}$$

$$\frac{\mathbf{H}, \neg P \vdash \textbf{simplify}(\mathcal{F}(\neg \top))}{\mathbf{H} \vdash \neg P \Rightarrow \mathcal{F}(P)} \quad \textbf{EVL1} \qquad \frac{\mathbf{H}, P \vdash \textbf{simplify}(\mathcal{F}(\top))}{\mathbf{H} \vdash P \Rightarrow \mathcal{F}(P)} \quad \textbf{EVL2}$$

- in EVL1 and EVL2, P is supposed to be a literal

Rewriting Rules for the Propositional Calculus Prover

$\top \wedge P == P$ **and1** $\qquad P \wedge \top == P$ **and2**

$\neg\top \wedge P == \neg\top$ **and3** $\qquad P \wedge \neg\top == \neg\top$ **and4**

$\top \vee P == \top$ **or1** $\qquad P \vee \top == \top$ **or2**

$\neg\top \vee P == P$ **or3** $\qquad P \vee \neg\top == P$ **or4**

$\top \Rightarrow P == P$ **imp1** $\qquad P \Rightarrow \top == \top$ **imp2**

$\neg\top \Rightarrow P == \top$ **imp3** $\qquad P \Rightarrow \neg\top == \neg P$ **imp4**

$\top \Leftrightarrow P == P$ **eqv1** $\qquad P \Leftrightarrow \top == P$ **eqv2**

$\neg\top \Leftrightarrow P == \neg P$ **eqv3** $\qquad P \Leftrightarrow \neg\top == \neg P$ **eqv4**

Example of Propositional Calculus Proof

$$(P \Rightarrow Q) \wedge (\neg R \Rightarrow \neg Q) \Rightarrow (P \Rightarrow R)$$

INI2

$$(P \Rightarrow Q) \Rightarrow ((\neg R \Rightarrow \neg Q) \Rightarrow (\neg (P \Rightarrow R) \Rightarrow \bot))$$

IMP2

$$\neg P \Rightarrow ((\neg R \Rightarrow \neg Q) \Rightarrow (\neg (P \Rightarrow R) \Rightarrow \bot))$$

EVL1

$$(\neg R \Rightarrow \neg Q) \Rightarrow (\neg (\neg \top \Rightarrow R) \Rightarrow \bot)$$

imp3

$$(\neg R \Rightarrow \neg Q) \Rightarrow (\neg \top \Rightarrow \bot)$$

imp3

$$(\neg R \Rightarrow \neg Q) \Rightarrow \top$$

imp2

$$\top$$

AXM

$$Q \Rightarrow ((\neg R \Rightarrow \neg Q) \Rightarrow (\neg (P \Rightarrow R) \Rightarrow \bot))$$

EVL2

$$(\neg R \Rightarrow \neg \top) \Rightarrow (\neg (P \Rightarrow R) \Rightarrow \bot)$$

imp4

$$\neg \neg R \Rightarrow (\neg (P \Rightarrow R) \Rightarrow \bot)$$

NOT

$$R \Rightarrow (\neg (P \Rightarrow R) \Rightarrow \bot)$$

EVL2

$$\neg (P \Rightarrow \top) \Rightarrow \bot$$

imp2

$$\neg \top \Rightarrow \bot$$

imp3

$$\top$$

AXM

The first order predicate calculus prover

Approach

- Applying the propositional calculus rules

- Applying some new predicate calculus rewriting and inference rules

- Until one reaches the following sequent:

$$\mathbf{H} \;\vdash\; \perp$$

- Trying then to derive a contradiction within the set of hypotheses **H**

- Sometimes restart the process (proof by cases)

First Order Predicate Calculus Prover Outline

- Syntax
- Rewriting rules
- Inference rules
- Normalisation and Skolemisation
- Mechanism (unit preference strategy)
- Example

This prover is built on top of the previous one

$$
\begin{array}{lcl}
\mathit{predicate} & ::= & \top \\
 & & \bot \\
 & & \neg\ \mathit{predicate} \\
 & & \mathit{predicate}\ \wedge\ \mathit{predicate} \\
 & & \mathit{predicate}\ \vee\ \mathit{predicate} \\
 & & \mathit{predicate}\ \Rightarrow\ \mathit{predicate} \\
 & & \mathit{predicate}\ \Leftrightarrow\ \mathit{predicate} \\
 & & \forall\, \mathit{variables} \cdot \mathit{predicate} \\
 & & \exists\, \mathit{variables} \cdot \mathit{predicate} \\
 & & \\
\mathit{variables} & ::= & \mathit{identifier} \\
 & & \mathit{identifier}, \mathit{variables}
\end{array}
$$

Predicate Calculus Rewriting Rules

$$\forall\, x \cdot \forall\, y \cdot P(x, y) \;==\; \forall\, x, y \cdot P(x, y) \qquad \textbf{grp1}$$

$$\exists\, x \cdot \exists\, y \cdot P(x, y) \;==\; \exists\, x, y \cdot P(x, y) \qquad \textbf{grp2}$$

$$\frac{\mathbf{H} \vdash \neg P \Rightarrow R}{\mathbf{H} \vdash \neg (\forall x \cdot P) \Rightarrow R} \quad \textbf{ALL1}$$

$$\frac{\mathbf{H}, \textbf{normalise}\,(\forall x \cdot P) \vdash R}{\mathbf{H} \vdash (\forall x \cdot P) \Rightarrow R} \quad \textbf{ALL2}$$

$$\frac{\mathbf{H}, \textbf{normalise}\,(\forall x \cdot \neg P) \vdash R}{\mathbf{H} \vdash \neg (\exists x \cdot P) \Rightarrow R} \quad \textbf{XST1}$$

$$\frac{\mathbf{H} \vdash P \Rightarrow R}{\mathbf{H} \vdash (\exists x \cdot P) \Rightarrow R} \quad \textbf{XST2}$$

- In rules **ALL1** and **XST2**, x is supposed to be not free in **H** and **R**

- **normalise** explained in next slide

Outcome of Normalisation: Two Forms

1. The first form corresponds to the following (with $n > 1$):

$$\forall\, x \cdot \neg\, (L_1(x) \;\wedge\; \ldots \;\wedge\; L_i(x) \;\wedge\; \ldots \;\wedge\; L_n(x))$$

   where each predicate $L_i(x)$ is a literal.

2. The second form corresponds to the following:

$$\forall\, x \cdot L(x)$$

   where the predicate $L(x)$ is a literal.

Normalisation Rewriting Rules (1)

Introducing once a Double Negation at the Outermost Level

$$\forall x \cdot P(x) \quad == \quad \forall x \cdot \neg\, \neg\, P(x)$$

Removing Implications and Equivalences

$$P \Rightarrow Q \quad == \quad \neg\, P \vee Q \qquad\qquad P \Leftrightarrow Q \quad == \quad (\neg\, P \vee Q) \wedge (P \vee \neg\, Q)$$

Moving down Negations

$$\neg\, \neg\, P \quad == \quad P \qquad \text{(outside outermost level)}$$

$$\neg\, (P \wedge Q) \quad == \quad \neg\, P \vee \neg\, Q$$

$$\neg\, (P \vee Q) \quad == \quad \neg\, P \wedge \neg\, Q$$

$$\neg\, \forall x \cdot P(x) \quad == \quad \exists x \cdot \neg\, P(x)$$

$$\neg\, \exists x \cdot P(x) \quad == \quad \forall x \cdot \neg\, P(x)$$

Normalisation Rewriting Rules (2)

Moving up Disjunctions

$$P \wedge (Q \vee R) \;==\; (P \wedge Q) \vee (P \wedge R)$$

$$(P \vee Q) \wedge R \;==\; (P \wedge R) \vee (Q \wedge R)$$

$$\exists\, x \cdot P(x) \vee Q(x) \;==\; (\exists\, x \cdot P(x)) \vee (\exists\, x \cdot Q(x))$$

Removing Disjunctions at the Outermost Level

$$\forall\, x \cdot \neg\,(P(x) \vee Q(x)) \;==\; (\forall\, x \cdot \neg\, P(x)) \wedge (\forall\, x \cdot \neg\, Q(x))$$

Removing Existential Quantifications at the Outermost Level

$$\forall x \cdot \neg\,(\ldots \wedge (\exists\, y \cdot P(x,y)) \wedge \ldots) \;==\; \forall x, y \cdot \neg\,(\ldots \wedge P(x,y) \wedge \ldots)$$

Removing Universal Quantifications at the Outermost Level (Skolemisation)

$$\forall x \cdot \neg\,(\ldots \wedge (\forall\, y \cdot P(x,y)) \wedge \ldots) \;==\; \forall x \cdot \neg\,(\ldots \wedge P(x, \mathsf{f}(x)) \wedge \ldots)$$

Example of Normalisation

$$
\begin{array}{l}
\forall\, x, y \cdot P(x) \;\wedge\; Q(y) \;\Rightarrow\; (\exists\, z \cdot R(z) \;\wedge S(x, y, z)) \\
\forall\, x \cdot Q(x) \;\vee\; R(x) \\
P(a) \\
\forall\, y \cdot R(y) \;\Rightarrow\; (\exists\, z \cdot Q(z) \;\wedge\; S(a, y, z)) \\
\Rightarrow \\
\exists\, x \cdot \forall\, y \cdot \exists\, z \cdot S(x, y, z)
\end{array}
$$

After normalisation and skolemisation, we obtain the following:

$$
\begin{array}{ll}
1: & \forall\, x, y \cdot \neg\,(P(x) \;\wedge\; Q(y) \;\wedge\; \neg\, R(\mathsf{a}(x, y))) \\
2: & \forall\, x, y \cdot \neg\,(P(x) \;\wedge\; Q(y) \;\wedge\; \neg\, S(x, y, \mathsf{a}(x, y))) \\
3: & \forall\, x \cdot \neg\,(\neg\, Q(x) \;\wedge\; \neg\, R(x)) \\
4: & P(a) \\
5: & \forall\, y \cdot \neg\,(R(y) \;\wedge\; \neg\, Q(\mathsf{b}(y))) \\
6: & \forall\, y \cdot \neg\,(R(y) \;\wedge\; \neg\, S(a, y, \mathsf{b}(y))) \\
7: & \forall\, x, z \cdot \neg\, S(x, \mathsf{c}(x), z)
\end{array}
$$

Skolemisation has the effect of cutting hypotheses

General Instantiation Rule

$$\frac{\mathbf{H},\ \forall x \cdot P(x),\ P(E) \vdash \bot}{\mathbf{H},\ \forall x \cdot P(x) \vdash \bot} \qquad \textbf{INS}$$

- The problem is now to discover instantiating expressions E

- In order to derive a contradiction

- We use the "unit preference strategy"

- *The Unit Preference Strategy in Theorem Proving* by L. Wos et al.
  Fall Joint Computer Conference, 1964.

- It consists in diminishing the size of instantiated hypotheses

- A set SLH made of Single Literal Hypotheses:

$$L$$

- A set MLH made of Multiple Literal Hypotheses ($n > 1$):

$$\neg\,(L_1 \;\wedge\; \ldots \;\wedge\; L_n)$$

- A set SUH made of Single Universal Hypotheses:

$$\forall\, x \cdot L(x)$$

- A set MUH made of Multiple Universal Hypotheses:

$$\forall\, x \cdot \neg\,(L_1(x) \;\wedge\; \ldots \;\wedge\; L_n(x))$$

Example

$$
\begin{array}{ll}
1: & \forall x, y \cdot \neg (P(x) \ \wedge \ Q(y) \ \wedge \ \neg R(\mathsf{a}(x, y))) \\
2: & \forall x, y \cdot \neg (P(x) \ \wedge \ Q(y) \ \wedge \ \neg S(x, y, \mathsf{a}(x, y))) \\
3: & \forall x \cdot \neg (\neg Q(x) \ \wedge \ \neg R(x)) \\
4: & P(a) \\
5: & \forall y \cdot \neg (R(y) \ \wedge \ \neg Q(\mathsf{b}(y))) \\
6: & \forall y \cdot \neg (R(y) \ \wedge \ \neg S(a, y, \mathsf{b}(y))) \\
7: & \forall x, z \cdot \neg S(x, \mathsf{c}(x), z)
\end{array}
$$

- MUH is made of 1, 2, 3, 5, and 6.

- SUH is made of 7.

- SLH is made of 4.

Discovering Contradictions

- SLH contains $\perp$.

- SLH contains L and $\neg L$

- SUH contains $\forall x \cdot L(x)$ and $\forall y \cdot \neg L(y)$

- SUH contains $\forall x \cdot L(x)$ and SLH contains $\neg L(E)$

- SUH contains $\forall x \cdot \neg L(x)$ and SLH contains $L(E)$

Entering a New Hypothesis

- In SLH or SUH
    - Check for contradiction (with SLH and SUH)
    - Simplify some MLH or MUH
- In MLH or MUH
    - Check how to simplify it with SLH and SUH

Example of First Order Predicate Calculus Proof (1)

$$\begin{array}{l} \forall x, y \cdot P(x) \wedge Q(y) \Rightarrow (\exists z \cdot R(z) \wedge S(x,y,z)) \\ \forall x \cdot Q(x) \vee R(x) \\ P(a) \\ \forall y \cdot R(y) \Rightarrow (\exists z \cdot Q(z) \wedge S(a,y,z)) \\ \Rightarrow \\ \exists x \cdot \forall y \cdot \exists z \cdot S(x,y,z) \end{array}$$

After normalisation, we obtain the following:

$$\begin{array}{ll} 1: & \forall x, y \cdot \neg (P(x) \wedge Q(y) \wedge \neg R(\mathsf{a}(x,y))) \\ 2: & \forall x, y \cdot \neg (P(x) \wedge Q(y) \wedge \neg S(x,y,\mathsf{a}(x,y))) \\ 3: & \forall x \cdot \neg (\neg Q(x) \wedge \neg R(x)) \\ 4: & P(a) \\ 5: & \forall y \cdot \neg (R(y) \wedge \neg Q(\mathsf{b}(y))) \\ 6: & \forall y \cdot \neg (R(y) \wedge \neg S(a,y,\mathsf{b}(y))) \\ 7: & \forall x, z \cdot \neg S(x,\mathsf{c}(x),z) \end{array}$$

$$
\begin{array}{ll}
1: & \forall x, y \cdot \neg (P(x) \ \wedge \ Q(y) \ \wedge \ \neg R(\mathsf{a}(x, y))) \\
2: & \forall x, y \cdot \neg (P(x) \ \wedge \ Q(y) \ \wedge \ \neg S(x, y, \mathsf{a}(x, y))) \\
3: & \forall x \cdot \neg (\neg Q(x) \ \wedge \ \neg R(x)) \\
4: & P(a) \\
5: & \forall y \cdot \neg (R(y) \ \wedge \ \neg Q(\mathsf{b}(y))) \\
6: & \forall y \cdot \neg (R(y) \ \wedge \ \neg S(a, y, \mathsf{b}(y))) \\
7: & \forall x, z \cdot \neg S(x, \mathsf{c}(x), z)
\end{array}
$$

We obtain the following instantiations:

$$
\begin{array}{llr}
8: & \forall y \cdot \neg (Q(y) \ \wedge \ \neg R(\mathsf{a}(a, y))) & (1, 4) \\
9: & \forall y \cdot \neg (Q(y) \ \wedge \ \neg S(a, y, \mathsf{a}(a, y))) & (2, 4) \\
10: & \forall x \cdot \neg (P(x) \ \wedge \ Q(\mathsf{c}(x))) & (2, 7) \\
11: & \neg R(\mathsf{c}(a)) & (6, 7) \\
12: & \neg Q(\mathsf{c}(a)) & (9, 7) \\
13: & Q(\mathsf{c}(a)) & (3, 11)
\end{array}
$$

Contradiction between 12 and 13.

The equality prover

Approach

- Apply propositional and predicate calculus rules

- Use specific equality rules

Equality Prover Outline

- Syntax

- Inference rules

- "One point" rule

- Example

Syntax for the Equality Prover

This prover is built on top of the previous one

$$
\begin{array}{lcl}
predicate & ::= & \top \\
 & & \bot \\
 & & \neg\, predicate \\
 & & predicate \;\wedge\; predicate \\
 & & predicate \;\vee\; predicate \\
 & & predicate \;\Rightarrow\; predicate \\
 & & predicate \;\Leftrightarrow\; predicate \\
 & & \forall\, variables \cdot predicate \\
 & & \exists\, variables \cdot predicate \\
 & & expression = expression \\
 & & \\
variables & ::= & identifier \\
 & & identifier, variables \\
 & & \\
expression & ::= & identifier \\
 & & expression \mapsto expression
\end{array}
$$

Equality Inference Rules

$$\frac{}{\mathbf{H} \vdash \neg\,(E{=}E) \Rightarrow P} \quad \textbf{EQL2} \qquad \frac{\mathbf{H} \vdash P}{\mathbf{H} \vdash E{=}E \Rightarrow P} \quad \textbf{EQL1}$$

$$\frac{\mathcal{H}(F) \vdash \mathcal{P}(F)}{\mathcal{H}(x) \vdash x{=}F \Rightarrow \mathcal{P}(x)} \quad \textbf{EQL3} \qquad \frac{\mathcal{H}(F) \vdash \mathcal{P}(F)}{\mathcal{H}(x) \vdash F{=}x \Rightarrow \mathcal{P}(x)} \quad \textbf{EQL4}$$

where x is a constant which is not free in F

Other Equality Rules

- Equality between pairs

- Applying an equality between expressions

For universally quantified predicates:

$$\begin{array}{l} \forall\, y, \ldots, x, \ldots, z \cdot P(y, \ldots, x, \ldots, z) \\ \qquad x = E \\ \qquad Q(y, \ldots, x, \ldots, z) \\ \qquad \Rightarrow \\ \qquad R(y, \ldots, x, \ldots, z) \end{array} \quad == \quad \begin{array}{l} \forall\, y, \ldots, z \cdot P(y, \ldots, E, \ldots, z) \\ \qquad Q(y, \ldots, E, \ldots, z) \\ \qquad \Rightarrow \\ \qquad R(y, \ldots, E, \ldots, z) \end{array}$$

For existentially quantified predicates:

$$\begin{array}{l} \exists\, y, \ldots, x, \ldots, z \cdot P(y, \ldots, x, \ldots, z) \\ \qquad x = E \\ \qquad Q(y, \ldots, x, \ldots, z) \end{array} \quad == \quad \begin{array}{l} \exists\, y, \ldots, z \cdot P(y, \ldots, E, \ldots, z) \\ \qquad Q(y, \ldots, E, \ldots, z) \end{array}$$

where variable x is not free in E

Practical "One Point" Rules

Applied during normalisation at the outermost level BEFORE skolemisation

$$\forall\, y, \ldots, x, \ldots, z \cdot \neg\, (P(x, \ldots) \;\wedge\; \ldots \;\wedge\; x = E \;\wedge\; \ldots \;\wedge\; Q(x, \ldots))$$
$$==$$
$$\forall\, y, \ldots, z \cdot \neg\, (P(E, \ldots) \;\wedge\; \ldots \;\wedge\; Q(E, \ldots))$$

Example of Proof with the Equality Prover

$$
\begin{array}{l}
\quad \forall x \cdot P(x) \;\wedge\; Q(x) \;\Rightarrow\; x = a \;\vee\; x = b \\
\quad \neg\, R(a) \\
\quad \forall x \cdot Q(x) \;\wedge\; R(x) \;\Rightarrow\; P(x) \\
\Rightarrow \\
\quad \forall x \cdot Q(x) \;\wedge\; R(x) \;\Rightarrow\; x = b
\end{array}
$$

The normalisation yields the following:

$$
\begin{array}{ll}
1: & \forall x \cdot \neg\,(P(x) \;\wedge\; Q(x) \;\wedge x \neq a \;\wedge\; x \neq b) \\
2: & \neg\, R(a) \\
3: & \forall x \cdot \neg\,(Q(x) \;\wedge\; R(x) \;\wedge\; \neg\, P(x)) \\
4: & Q(x) \\
5: & R(x) \\
6: & x \neq b
\end{array}
$$

$$
\begin{array}{ll}
1: & \forall x \cdot \neg (P(x) \ \wedge \ Q(x) \ \wedge x \neq a \ \wedge \ x \neq b) \\
2: & \neg R(a) \\
3: & \forall x \cdot \neg (Q(x) \ \wedge \ R(x) \ \wedge \ \neg P(x)) \\
4: & Q(x) \\
5: & R(x) \\
6: & x \neq b
\end{array}
$$

Instantiations yield:

$$
\begin{array}{llr}
7: & P(x) & (4,5,3) \\
8: & x = a & (1,7,4,6) \\
9: & R(a) & (5,8)
\end{array}
$$

9 contradicts 2.

The set theory prover

Approach

- Introducing the membership operator $\in$

- Translating membership predicates $E \in S$ as much as possible

- Performing a predicate calculus proof of the translated predicate

- Set theory specific mechanisms

- Using the set theory presented in:

  Modeling in Event-B by J-R. Abrial. CUP (2010)

Set Theory Prover Outline

- Syntax
- Axioms of set theory
- Operators of set theory
- Examples of translation
- Exploiting types
- Example
- Instantiating set quantified variables (2nd order)
- Partial translations

Set Theory Prover Syntax

- This prover is built on top of the previous one

$$
\begin{array}{lcl}
predicate & ::= & \top \\
 & & \bot \\
 & & \neg\, predicate \\
 & & predicate \;\wedge\; predicate \\
 & & predicate \;\vee\; predicate \\
 & & predicate \;\Rightarrow\; predicate \\
 & & predicate \;\Leftrightarrow\; predicate \\
 & & \forall\, variables \cdot predicate \\
 & & \exists\, variables \cdot predicate \\
 & & expression = expression \\
 & & expression \;\in\; expression \\
 & & \\
variables & ::= & identifier \\
 & & identifier, variables \\
 & & \\
expression & ::= & identifier \\
 & & expression \mapsto expression \\
 & & expression \times expression \\
 & & \mathbb{P}(expression) \\
 & & \{\, variables \cdot predicate \mid expression \,\}
\end{array}
$$

Predicates of the form $E \in S$ are translated as indicated

Operator	Predicate	Rewritten
Cartesian product	$E \mapsto F \in S \times T$	$E \in S \;\wedge\; F \in T$
Power set	$E \in \mathbb{P}(S)$	$\forall x \cdot x \in E \;\Rightarrow\; x \in S$
Set comprehension	$E \in \{\, x \cdot P \mid F \,\}$	$\exists x \cdot P \;\wedge\; F = E$
Set equality	$S = T$	$S \in \mathbb{P}(T) \;\wedge\; T \in \mathbb{P}(S)$

Variable x is not free in E and S

Predicates of the form $E \in S$ are translated as indicated

Operator	Predicate	Rewritten
Inclusion	$S \subseteq T$	$S \in \mathbb{P}(T)$
Union	$E \in S \cup T$	$E \in S \quad \vee \quad E \in T$
Intersection	$E \in S \cap T$	$E \in S \quad \wedge \quad E \in T$
Difference	$E \in S \setminus T$	$E \in S \quad \wedge \quad \neg\,(E \in T)$
Extension	$E \in \{a, \ldots, b\}$	$E = a \quad \vee \quad \ldots \quad \vee \quad E = b$
Empty set	$E \in \varnothing$	$\perp$

Predicates of the form $E \in S$ are translated as indicated

Operator	Predicate	Rewritten
Binary relations	$r \in S \leftrightarrow T$	$r \subseteq S \times T$
Converse	$E \mapsto F \in r^{-1}$	$F \mapsto E \in r$
Relational Image	$F \in r[U]$	$\exists x \cdot x \in U \ \wedge \ x \mapsto F \in r$
Forward composition	$E \mapsto F \in f \mathbin{;} g$	$\exists x \cdot E \mapsto x \in f \ \wedge \ x \mapsto F \in g$

Variable x is not free in E, F, U, r, f, and g

Functional Operators

Predicates of the form $E \in S$ are translated as indicated

Operator	Predicate	Rewritten
Identity	$E \mapsto F \in \mathrm{id}$	$E = F$
Set of all partial functions	$f \in S \pfun T$	$f \in S \rel T \quad \wedge \quad f^{-1} \mathbin{;} f \subseteq \mathrm{id}$
Set of all total functions	$f \in S \tfun T$	$f \in S \pfun T \quad \wedge \quad S = \mathrm{dom}\,(f)$
Set of all partial injections	$f \in S \pinj T$	$f \in S \pfun T \quad \wedge \quad f^{-1} \in T \pfun S$
Set of all total injections	$f \in S \tinj T$	$f \in S \tfun T \quad \wedge \quad f^{-1} \in T \pfun S$
Set of all partial surjections	$f \in S \psur T$	$f \in S \pfun T \quad \wedge \quad T = \mathrm{ran}\,(f)$
Set of all total surjections	$f \in S \tsur T$	$f \in S \tfun T \quad \wedge \quad T = \mathrm{ran}\,(f)$
Set of all bijections	$f \in S \tbij T$	$f \in S \tinj T \quad \wedge \quad f \in S \tsur T$

Example of Translation (1)

The following predicate:

$$r \in S \leftrightarrow T \ \land \ a \subseteq S \ \land \ b \subseteq S \ \Rightarrow \ r[a \cup b] = r[a] \cup r[b]$$

is translated to:

$$\forall x, y \cdot x \mapsto y \in r \ \Rightarrow \ x \in S \ \land \ y \in T$$

$$\forall x \cdot x \in a \ \Rightarrow \ x \in S$$

$$\forall x \cdot x \in b \ \Rightarrow \ x \in S$$

$$\Rightarrow$$

$$\begin{aligned} \forall x \cdot \ & (\exists x0 \cdot (x0 \in a \ \lor \ x0 \in b) \ \land \ x0 \mapsto x \in r) \\ \Leftrightarrow & \\ & (\exists x0 \cdot x0 \in a \ \land \ x0 \mapsto x \in r) \ \lor \ (\exists x0 \cdot x0 \in b \ \land \ x0 \mapsto x \in r) \end{aligned}$$

Example of Translation (2)

The following predicate:

$$f \in S \rightarrowtail T \;\land\; p \in U \leftrightarrow S \;\land\; q \in U \leftrightarrow S \;\land\; p \,;\, f = q \,;\, f \;\Rightarrow\; p = q$$

is translated to:

$$\begin{array}{l}
\forall\, x, y \cdot x \mapsto y \in f \;\Rightarrow\; x \in S \;\land\; y \in T \\
\forall\, x, x0, x1 \cdot x \mapsto x0 \in f \;\land\; x \mapsto x1 \in f \;\Rightarrow\; x0 = x1 \\
\forall\, x \cdot \exists\, x0 \cdot x \mapsto x0 \in f \\
\forall\, x, x0, x1 \cdot x0 \mapsto x \in f \;\land\; x1 \mapsto x \in f \;\Rightarrow\; x0 = x1
\end{array}$$

$$\forall\, x, y \cdot x \mapsto y \in p \;\Rightarrow\; x \in U \;\land\; y \in S$$

$$\forall\, x, y \cdot x \mapsto y \in q \;\Rightarrow\; x \in U \;\land\; y \in S$$

$$\begin{array}{l}
\forall\, x, x0 \cdot (\exists\, x1 \cdot x \mapsto x1 \in p \;\land\; x1 \mapsto x0 \in f) \\
\qquad\quad \Leftrightarrow \\
\qquad\quad (\exists\, x1 \cdot x \mapsto x1 \in q \;\land\; x1 \mapsto x0 \in f)
\end{array}$$

$$\Rightarrow$$

$$\forall\, x, y \cdot x \mapsto y \in p \;\Leftrightarrow\; x \mapsto y \in q$$

Given the following statement:

$$
\begin{array}{l}
\quad r \in S \leftrightarrow T \\
\quad a \subseteq S \\
\quad b \subseteq T \\
\Rightarrow \\
\quad r[a] \subseteq b \;\Leftrightarrow\; a \subseteq S \setminus r^{-1}[T \setminus b]
\end{array}
$$

we can determine the types of its components as follows:

$$
\begin{array}{l}
\textbf{type}(r) = \mathbb{P}(S \times T) \\
\textbf{type}(S) = \mathbb{P}(S) \\
\textbf{type}(T) = \mathbb{P}(T) \\
\textbf{type}(a) = \mathbb{P}(S) \\
\textbf{type}(b) = \mathbb{P}(T)
\end{array}
$$

They are all determined from the carrier sets S and T

Defining carrier sets as basic types:

$$
\begin{array}{l}
\quad cs : S \quad T \\
\quad r \in S \leftrightarrow T \\
\quad a \subseteq S \\
\quad b \subseteq T \\
\Rightarrow \\
\quad r[a] \subseteq b \;\Leftrightarrow\; a \subseteq S \setminus r^{-1}[T \setminus b]
\end{array}
$$

Syntax for types:

$$
\begin{array}{lll}
type & ::= & carrier\_set \\
 & & type \times type \\
 & & \mathbb{P}(type)
\end{array}
$$

Exploiting Types

- Because of typing, set theoretic statements are richer than pure Predicate Calculus statements

- Instantiating a variable with an expression requires that they have both the same type

- Two effects:
    - avoiding wrong instantiations
    - allowing more instantiations

$$
\begin{array}{l}
\quad cs : S \;\; T \;\; U \\
\quad f \in S \mathrel{+\!\!\!\rightarrow} T \\
\quad g \in T \mathrel{+\!\!\!\rightarrow} U \\
\Rightarrow \\
\quad f \,;\, g \in S \mathrel{+\!\!\!\rightarrow} U
\end{array}
$$

The translation yields:

$$
\begin{array}{l}
\quad \forall\, x, x0, x1 \cdot x \mapsto x0 \in f \;\wedge\; x \mapsto x1 \in f \;\Rightarrow\; x1 = x0 \\
\quad \forall\, x, x0, x1 \cdot x \mapsto x0 \in g \;\wedge\; x \mapsto x1 \in g \;\Rightarrow\; x1 = x0 \\
\Rightarrow \\
\quad \forall\, x, x0, x1 \cdot \;\begin{array}{l}
\exists\, x1 \cdot x \mapsto x1 \in f \;\wedge x1 \mapsto x0 \in g \\
\exists\, x0 \cdot x \mapsto x0 \in f \;\wedge x0 \mapsto x1 \in g \\
\Rightarrow \\
x1 = x0
\end{array}
\end{array}
$$

$$
\begin{array}{l}
\quad \forall x, x0, x1 \cdot x \mapsto x0 \in f \;\wedge\; x \mapsto x1 \in f \;\Rightarrow\; x1 = x0 \\
\quad \forall x, x0, x1 \cdot x \mapsto x0 \in g \;\wedge\; x \mapsto x1 \in g \;\Rightarrow\; x1 = x0 \\
\Rightarrow \\
\quad \forall x, x0, x1 \cdot \begin{array}[t]{l} \exists x1 \cdot x \mapsto x1 \in f \;\wedge x1 \mapsto x0 \in g \\ \exists x0 \cdot x \mapsto x0 \in f \;\wedge x0 \mapsto x1 \in g \\ \Rightarrow \\ x1 = x0 \end{array}
\end{array}
$$

The normalisation and skolemisation yields:

$$
\begin{array}{ll}
1: & \forall x, x0, x1 \cdot \neg\,(x \mapsto x0 \in f \;\wedge\; x \mapsto x1 \in f \;\wedge\; x1 \neq x0) \\
2: & \forall x, x0, x1 \cdot \neg\,(x \mapsto x0 \in g \;\wedge\; x \mapsto x1 \in g \;\wedge\; x1 \neq x0) \\
3: & \mathsf{a} \mapsto \mathsf{d} \in f \\
4: & \mathsf{d} \mapsto \mathsf{b} \in g \\
5: & \mathsf{a} \mapsto \mathsf{e} \in f \\
6: & \mathsf{e} \mapsto \mathsf{c} \in g \\
7: & \mathsf{b} \neq \mathsf{c}
\end{array}
$$

$$
\begin{array}{ll}
1: & \forall x, x0, x1 \cdot \neg\,(x \mapsto x0 \in f \;\wedge\; x \mapsto x1 \in f \;\wedge\; x0 \neq x1) \\
2: & \forall x, x0, x1 \cdot \neg\,(x \mapsto x0 \in g \;\wedge\; x \mapsto x1 \in g \;\wedge\; x0 \neq x1) \\
3: & \mathsf{a} \mapsto \mathsf{d} \in f \\
4: & \mathsf{d} \mapsto \mathsf{b} \in g \\
5: & \mathsf{a} \mapsto \mathsf{e} \in f \\
6: & \mathsf{e} \mapsto \mathsf{c} \in g \\
7: & \mathsf{b} \neq \mathsf{c}
\end{array}
$$

We obtain the following successive instantiations:

$$
\begin{array}{lll}
8: & \forall x1 \cdot \neg\,(\mathsf{a} \mapsto x1 \in f \;\wedge\; \mathsf{d} \neq x1) & (1,3) \\
9: & \forall x1 \cdot \neg\,(\mathsf{e} \mapsto x1 \in g \;\wedge\; \mathsf{c} \neq x1) & (2,6) \\
10: & \mathsf{d} = \mathsf{e} & (8,5) \\
11: & \mathsf{e} \mapsto \mathsf{b} \notin g & (9,7) \\
12: & \mathsf{d} \mapsto \mathsf{b} \notin g & (10,11)
\end{array}
$$

12 contradicts 4.

Two Important Extensions

- Instantiating set quantified variables: 2nd order statements

- Partial translation of set theoretic statements

- Both extensions proposed by Dominique Cansell

Instantiating Set Quantified Variables. An Example

$$
\begin{array}{l}
\quad cs : S \\
\quad r \in S \leftrightarrow S \\
\quad \forall p \cdot p \subseteq r^{-1}[p] \;\Rightarrow\; p = \varnothing \\
\quad \forall x \cdot r[\{x\}] \subseteq q \;\Rightarrow\; x \in q \\
\quad x \in S \\
\Rightarrow \\
\quad x \in q
\end{array}
$$

The normalisation and skolemisation yields the following :

$$
\begin{array}{ll}
1: & \forall p, x \cdot \neg\,(\mathsf{a}(p) \notin p \;\wedge\; x \in p) \\
2: & \forall p, x, x0 \cdot \neg\,(x0 \in p \;\wedge\; \mathsf{a}(p) \mapsto x0 \in r \;\wedge\; x \in p) \\
3: & \forall x \cdot \neg\,(x \mapsto \mathsf{b}(x) \notin r \;\wedge\; x \notin q) \\
4: & \forall x \cdot \neg\,(\mathsf{b}(x) \in q \;\wedge\; x \notin q) \\
5: & x \notin q
\end{array}
$$

- p is a set quantified variable: its type is $\mathbb{P}(S)$

$$
\begin{array}{ll}
\mathbf{1:} & \forall\, p, x \cdot \neg\,(\mathsf{a}(p) \notin p \;\wedge\; x \in p) \\
\mathbf{2:} & \forall\, p, x, x0 \cdot \neg\,(x0 \in p \;\wedge\; \mathsf{a}(p) \mapsto x0 \in r \;\wedge\; x \in p) \\
\mathbf{3:} & \forall\, x \cdot \neg\,(x \mapsto \mathsf{b}(x) \notin r \;\wedge\; x \notin q) \\
\mathbf{4:} & \forall\, x \cdot \neg\,(\mathsf{b}(x) \in q \;\wedge\; x \notin q) \\
\mathbf{5:} & x \notin q
\end{array}
$$

Quantified variable x and constant x have the same type, we obtain:

$$
\mathbf{6:} \forall\, p \cdot \neg\,(\mathsf{a}(p) \notin p \;\wedge\; x \in p)
$$

$$
\begin{array}{ll}
\mathbf{1:} & \forall p, x \cdot \neg\,(\mathsf{a}(p) \notin p \;\wedge\; x \in p) \\
\mathbf{2:} & \forall p, x, x0 \cdot \neg\,(x0 \in p \;\wedge\; \mathsf{a}(p) \mapsto x0 \in r \;\wedge\; x \in p) \\
\mathbf{3:} & \forall x \cdot \neg\,(x \mapsto \mathsf{b}(x) \notin r \;\wedge\; x \notin q) \\
\mathbf{4:} & \forall x \cdot \neg\,(\mathsf{b}(x) \in q \;\wedge\; x \notin q) \\
\mathbf{5:} & x \notin q \\
\mathbf{6:} & \forall p \cdot \neg\,(\mathsf{a}(p) \notin p \;\wedge\; x \in p)
\end{array}
$$

- Suppose that we can instantiate p with $\{x|P(x)\}$ in 6.

- Then the predicate $x \in p$ in 6 becomes $P(x)$.

- By instantiating p with $\{x \mid x \notin q\}$ in 6, we obtain (thanks to 5):

$$\mathbf{7:} \quad \mathsf{a}(Q) \notin q$$

where Q denotes the set $\{x \mid x \notin q\}$.

$$
\begin{array}{ll}
1: & \forall p, x \cdot \neg\, (\mathsf{a}(p) \notin p \ \wedge \ x \in p) \\
2: & \forall p, x, x0 \cdot \neg\, (x0 \in p \ \wedge \ \mathsf{a}(p) \mapsto x0 \in r \ \wedge \ x \in p) \\
3: & \forall x \cdot \neg\, (x \mapsto \mathsf{b}(x) \notin r \ \wedge \ x \notin q) \\
4: & \forall x \cdot \neg\, (\mathsf{b}(x) \in q \ \wedge \ x \notin q) \\
5: & x \notin q \\
6: & \forall p \cdot \neg\, (\mathsf{a}(p) \notin p \ \wedge \ x \in p) \\
7: & \mathsf{a}(Q) \notin q
\end{array}
$$

More instantiations:

$$
\begin{array}{lll}
8: & \mathsf{a}(Q) \mapsto \mathrm{b}(\mathsf{a}(Q)) \in r & (3, 7) \\
9: & \mathsf{b}(\mathsf{a}(Q)) \in q & (8, 5, 2) \\
10: & \mathsf{a}(Q) \in q & (9, 4)
\end{array}
$$

10 contradicts 7.

$$
\begin{array}{l}
\quad \boldsymbol{cs} : \boldsymbol{S} \\
\quad \boldsymbol{f} \subseteq \mathbb{P}(\boldsymbol{S}) \\
\quad \boldsymbol{M} \cup \boldsymbol{A} \in \boldsymbol{f} \\
\quad \forall\, \boldsymbol{X}, \boldsymbol{Y} \cdot \boldsymbol{X} \in \boldsymbol{f} \;\wedge\; \boldsymbol{X} \subseteq \boldsymbol{Y} \;\Rightarrow\; \boldsymbol{Y} \in \boldsymbol{f} \\
\Rightarrow \\
\quad \boldsymbol{M} \cup (\boldsymbol{A} \cup \boldsymbol{B}) \in \boldsymbol{f}
\end{array}
$$

The translation yields:

$$
\begin{array}{l}
\quad \boldsymbol{M} \cup \boldsymbol{A} \in \boldsymbol{f} \\
\quad \forall\, \boldsymbol{X}, \boldsymbol{Y} \cdot \boldsymbol{X} \in \boldsymbol{f} \;\wedge\; (\forall\, \boldsymbol{x} \cdot \boldsymbol{x} \in \boldsymbol{X} \;\Rightarrow\; \boldsymbol{x} \in \boldsymbol{Y}) \;\Rightarrow \boldsymbol{Y} \in \boldsymbol{f} \\
\Rightarrow \\
\quad \boldsymbol{M} \cup (\boldsymbol{A} \cup \boldsymbol{B}) \in \boldsymbol{f}
\end{array}
$$

$\boldsymbol{M} \cup \boldsymbol{A} \in \boldsymbol{f}$ and $\boldsymbol{M} \cup (\boldsymbol{A} \cup \boldsymbol{B}) \in \boldsymbol{f}$ cannot be translated.

We continue with the proof. Normalisation yields:

$$\begin{array}{ll} \mathbf{1:} & M \cup A \in f \\ \mathbf{2:} & \forall X, Y \cdot \neg\,(\mathsf{a}(X,Y) \notin X \;\wedge\; X \in f \;\wedge\; Y \notin f) \\ \mathbf{3:} & \forall X, Y \cdot \neg\,(\mathsf{a}(X,Y) \in Y \;\wedge\; X \in f \;\wedge\; Y \notin f) \\ \mathbf{4:} & M \cup (A \cup B) \notin f \end{array}$$

We obtain the following instantiations:

$$\begin{array}{l} T \in M \cup A \\ T \notin M \cup (A \cup B) \end{array}$$

where T stands for $\mathsf{a}(M \cup A, M \cup (A \cup B)$.

- These are put down in the goal (see next slide)

$$\boldsymbol{T} \notin \boldsymbol{M} \cup (\boldsymbol{A} \cup \boldsymbol{B}) \Rightarrow (\boldsymbol{T} \in \boldsymbol{M} \cup \boldsymbol{A} \Rightarrow \bot)$$

and then translated yielding:

$$\boldsymbol{T} \notin \boldsymbol{M} \wedge \boldsymbol{T} \notin \boldsymbol{A} \wedge \boldsymbol{T} \notin \boldsymbol{B} \Rightarrow (\boldsymbol{T} \in \boldsymbol{M} \vee \boldsymbol{T} \in \boldsymbol{A} \Rightarrow \bot)$$

We obtain the following hypotheses:

$$\begin{array}{ll} \mathbf{5:} & \boldsymbol{T} \notin \boldsymbol{M} \\ \mathbf{6:} & \boldsymbol{T} \notin \boldsymbol{A} \\ \mathbf{7:} & \boldsymbol{T} \notin \boldsymbol{B} \end{array}$$

This results in the following goal:

$$\neg \top \vee \neg \top \Rightarrow \bot$$

reducing to $\top$.

Other questions

- Constructing an independent proof checker

- Detecting which hypotheses are used in a proof

Conclusion

- We presented a series of embedded provers

- Implementation (in Java) is an on-going project

- Development so far is encouraging.

- Exercises of predicate calculus are all proved in:

  Mathematical Logic: Applications and Theory

  by J.E. Rubin. Saunders College Publishing (1990)