Certification of prefixed tableau proofs for modal logic

Tomer Libal and **Marco Volpe**

INRIA, Parsifal Team

Can we trust provers?

- Complex software is rarely free of **bugs**.
- Automated theorem provers are complex software - can we **trust** them?

The Provers' Tower of Babel

- Current provers can rarely **share** each other's proofs
- Work has been done for building **bridges** between two specific provers (but even a change in the version number of one prover can cause that bridge to collapse)

Towards proof certification

Motivating questions

1. Can we **trust** provers?
2. Can provers talk a **common language**?

Goal

Provide a **flexible framework** for defining the semantics of a wide range of proof evidences in such a way that:

- **provers** would define the meaning of their own proof evidence;
- trusted proof **checkers** would be able to interpret that meaning and check its formal correctness.

An analogy

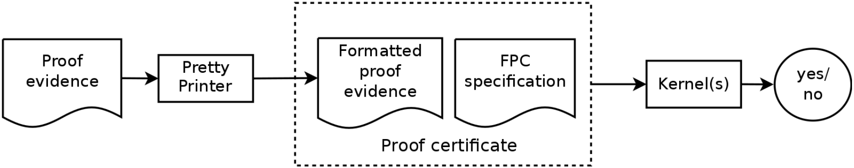
Structural Operational Semantics

1. There are many programming languages.
2. SOS can define the semantics of many of them.
3. Compilers can be built based on the semantics.

Foundational Proof Certificates (FPCs)

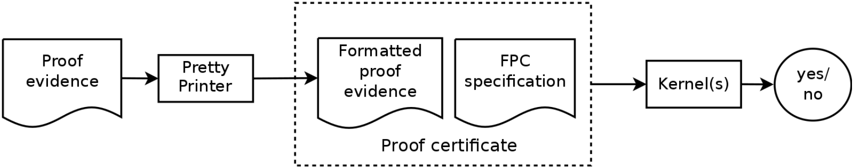
1. There are many forms of proof evidence.
2. FPC can define the semantics of many of them.
3. Checkers can be built based on the semantics.

Foundational Proof Certificates



- **Proof evidence**: The proof output from a prover.
- **Pretty printer**: Some program for properly formatting the proof evidence.
- **FPC specification**: Specification of predicates used to interpret the proof evidence in order to guide the kernel proof search.
- **Kernel**: A trustable low-level calculus, with additional control predicates.

Foundational Proof Certificates (our implementation)



- **Proof evidence**: The proof output from a prover.
- **Pretty printer**: Some (typically OCaml) program for properly formatting the proof evidence (as a λProlog file).
- **FPC specification**: λProlog specification of predicates used to interpret the proof evidence in order to guide the kernel proof search.
- **Kernel**: An encoding of (focused) sequent calculus (LKF + control predicates) as a λProlog program.

Focused proof systems

Let's consider a **sequent calculus** for classical first-order logic (LK).

- Reduce the **search space**.
- Better organize the **structure** of derivations.
- Emphasis on: non-invertible vs. invertible rules.
- Propositional connectives have:
  - a positive version;
  - a negative version.

$$\frac{\vdash \Theta, B_i}{\vdash \Theta, B_1 \vee B_2}\ \vee_F^+ \qquad \frac{\vdash \Theta, B_1, B_2}{\vdash \Theta, B_1 \vee B_2}\ \vee_F^-$$

Focused proof systems

Let's consider a **sequent calculus** for classical first-order logic (LK).

- Reduce the **search space**.
- Better organize the **structure** of derivations.
- Emphasis on: non-invertible vs. invertible rules.
- Propositional connectives have:
  - a positive version;
  - a negative version.
- Polarization of a formula does not affect its **provability**.

Focused proof systems

store (a positive formula to possibly focus on later)

$\vdash \Theta \Uparrow \Gamma$ $\quad$ $t^{-}, f^{-}, \vee^{-}, \wedge^{-}, \forall$

release

$\vdash \Theta \Downarrow A$ $\quad$ $t^{+}, f^{+}, \vee^{+}, \wedge^{+}, \exists$

decide (on a positive formula to focus on)

Focused proof systems

store	(a positive formula to possibly focus on later)
$\vdash \Theta \Uparrow \Gamma$	**NEGATIVE PHASE (invertible)**
release	(change of phase)
$\vdash \Theta \Downarrow A$	**POSITIVE PHASE (non-invertible)**
decide	(on a positive formula to focus on)

Focused proof systems

store (a positive formula to possibly focus on later)

$\vdash \Theta \Uparrow \Gamma$ $t^{-}, f^{-}, \vee^{-}, \wedge^{-}, \forall$

By the way,

release

this is a **BIPOLE**

$\vdash \Theta \Downarrow A$ $t^{+}, f^{+}, \vee^{+}, \wedge^{+}, \exists$

decide (on a positive formula to focus on)

A focused proof system for classical logic (LKF)

Negative introduction rules

$$\frac{}{\vdash \Theta \Uparrow t^-, \Gamma}\ t^- \qquad \frac{\vdash \Theta \Uparrow A, \Gamma \quad \vdash \Theta \Uparrow B, \Gamma}{\vdash \Theta \Uparrow A \wedge^- B, \Gamma}\ \wedge^- \qquad \frac{\vdash \Theta \Uparrow \Gamma}{\vdash \Theta \Uparrow f^-, \Gamma}\ f^- \qquad \frac{\vdash \Theta \Uparrow A, B, \Gamma}{\vdash \Theta \Uparrow A \vee^- B, \Gamma}\ \vee^-$$

$$\frac{\vdash \Theta \Uparrow [y/x]B, \Gamma}{\vdash \Theta \Uparrow \forall x.B, \Gamma}\ \forall\dagger$$

Positive introduction rules

$$\frac{}{\vdash \Theta \Downarrow t^+}\ t^+ \qquad \frac{\vdash \Theta \Downarrow B_1 \quad \vdash \Theta \Downarrow B_2}{\vdash \Theta \Downarrow B_1 \wedge^+ B_2}\ \wedge^+ \qquad \frac{\vdash \Theta \Downarrow B_i}{\vdash \Theta \Downarrow B_1 \vee^+ B_2}\ \vee^+, i \in \{1,2\} \qquad \frac{\vdash \Theta \Downarrow [t/x]B}{\vdash \Theta \Downarrow \exists x.B}\ \exists$$

Identity rules

$$\frac{}{\vdash \neg P_a, \Theta \Downarrow P_a}\ init \qquad \frac{\vdash \Theta \Uparrow B \quad \vdash \Theta \Uparrow \neg B}{\vdash \Theta \Uparrow \cdot}\ cut$$

Structural rules

$$\frac{\vdash \Theta, C \Uparrow \Gamma}{\vdash \Theta \Uparrow C, \Gamma}\ store \qquad \frac{\vdash \Theta \Uparrow N}{\vdash \Theta \Downarrow N}\ release \qquad \frac{\vdash P, \Theta \Downarrow P}{\vdash P, \Theta \Uparrow \cdot}\ decide$$

Labeled proof systems

- **Labeled deduction** approach: we encode in the syntax additional information (e.g. of a semantic nature).

Labels denoting worlds

- Two classes of **formulas**:
  1. Labeled logical formulas, e.g. $x : A$
  2. Relational formulas, e.g. xRy
- The **basic idea** is:
  - each label y refers to a world $\overline{y}$ in the Kripke semantics
  - the relational symbol R refers to the accessibility relation

A labeled sequent system for modal logic

Classical rules

$$\frac{}{x : P, \Gamma \vdash \Delta, x : P}\ init \qquad \frac{x : A, x : B, \Gamma \vdash \Delta}{x : A \wedge B, \Gamma \vdash \Delta}\ L\wedge \qquad \frac{\Gamma \vdash \Delta, x : A \quad \Gamma \vdash \Delta, x : B}{\Gamma \vdash \Delta, x : A \wedge B}\ R\wedge$$

$$\frac{x : A, \Gamma \vdash \Delta \quad x : B, \Gamma \vdash \Delta}{x : A \vee B, \Gamma \vdash \Delta}\ L\vee \qquad \frac{\Gamma \vdash \Delta, x : A, x : B}{\Gamma \vdash \Delta, x : A \vee B}\ R\vee$$

Modal rules

$$\frac{y : A, x : \Box A, xRy, \Gamma \vdash \Delta}{x : \Box A, xRy, \Gamma \vdash \Delta}\ L\Box \qquad \frac{xRy, \Gamma \vdash \Delta, y : A}{\Gamma \vdash \Delta, x : \Box A}\ R\Box$$

$$\frac{xRy, y : A, \Gamma \vdash \Delta}{x : \Diamond A, \Gamma \vdash \Delta}\ L\Diamond \qquad \frac{xRy, \Gamma \vdash \Delta, x : \Diamond A, y : A}{xRy, \Gamma \vdash \Delta, x : \Diamond A}\ R\Diamond$$

In $R\Box$ and $L\Diamond$, y does not occur in the conclusion.

A prefixed tableau system for modal logic

Classical rules

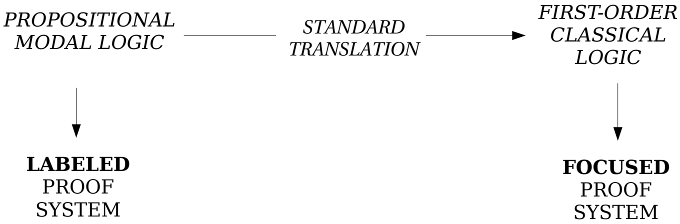
$$\frac{\sigma : A \wedge B}{\sigma : A, \sigma : B} \wedge_F \qquad \frac{\sigma : A \vee B}{\sigma : A \quad \mid \quad \sigma : B} \vee_F$$

Modal rules

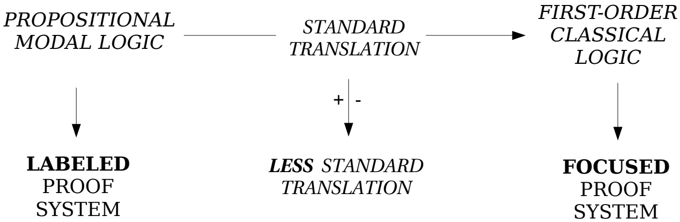
$$\frac{\sigma : \Box A}{\sigma.n : A} \Box_F \qquad \frac{\sigma : \Diamond A}{\sigma.n : A} \Diamond_F$$

In $\Box_F$, $\sigma.n$ is used. In $\Diamond_F$, $\sigma.n$ is new.

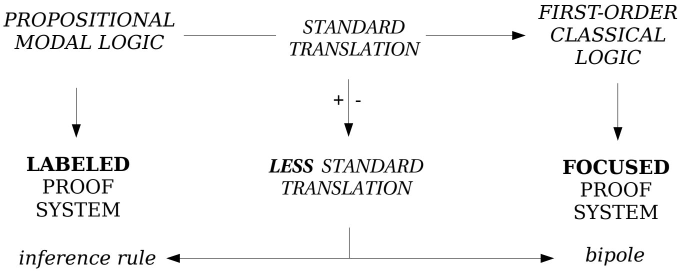
Plus branch **closure rules**, of course.


PROPOSITIONAL MODAL LOGIC
STANDARD TRANSLATION
FIRST-ORDER CLASSICAL LOGIC
LABELED PROOF SYSTEM
FOCUSED PROOF SYSTEM

Labeling and focusing



Labeling and focusing



The standard translation

Modal language $\Rightarrow$ **FO language** with:
- a binary predicate R
- a unary predicate P for each $P \in \mathcal{P}$

$$
\begin{aligned}
ST_x(P) &= P(x) \\
ST_x(A \wedge B) &= ST_x(A) \wedge ST_x(B) \\
ST_x(\Box A) &= \forall y(\neg R(x,y) \vee ST_y(A)) \\
ST_x(\Diamond A) &= \exists y(R(x,y) \wedge ST_y(A))
\end{aligned}
$$

where x is a free variable.

For any modal formula A, any model $\mathcal{M}$ and any world w:

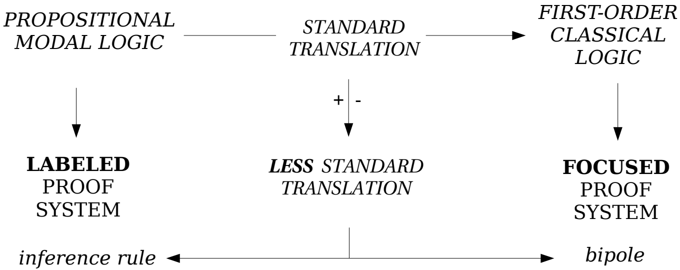
$$\mathcal{M}, w \models A \qquad \text{iff} \qquad \mathcal{M} \models ST_x(A)[x \leftarrow w]$$

Our translation $[\cdot]$

$$
\begin{array}{rcl}
ST_x(P) & = & P(x) \\
[\mathbf{x} : \mathbf{P}] & = & \mathbf{P(x)} \\
 & & \\
ST_x(A \wedge B) & = & ST_x(A) \wedge ST_x(B) \\
[\mathbf{x} : \mathbf{A} \wedge \mathbf{B}] & = & \boldsymbol{\partial}^{+}([\mathbf{x} : \mathbf{A}]) \wedge^{-} \boldsymbol{\partial}^{+}([\mathbf{x} : \mathbf{B}]) \\
 & & \\
ST_x(\Box A) & = & \forall y(\neg R(x, y) \vee ST_y(A)) \\
[\mathbf{x} : \Box \mathbf{A}] & = & \forall \mathbf{y}(\neg \mathbf{R(x, y)} \vee^{-} \boldsymbol{\partial}^{+}([\mathbf{y} : \mathbf{A}])) \\
 & & \\
ST_x(\Diamond A) & = & \exists y(R(x, y) \wedge ST_y(A)) \\
[\mathbf{x} : \Diamond \mathbf{A}] & = & \exists \mathbf{y}(\mathbf{R(x, y)} \wedge^{+} \boldsymbol{\partial}^{-}(\boldsymbol{\partial}^{+}([\mathbf{y} : \mathbf{A}])))
\end{array}
$$

Delay operators (∂^{+}, ∂^{-}) force a formula to be positive or negative.

Theorem of adequacy



LKF + control predicates

$$\frac{\vdash \Theta \Downarrow [t/x]B}{\vdash \Theta \Downarrow \exists x.B}\ \exists$$

$$\frac{\Xi' \vdash \Theta \Downarrow [t/x]B \quad \text{exists}_e(\Xi, t, \Xi')}{\Xi \vdash \Theta \Downarrow \exists x.B}$$

Typically, in an FPC specification, the information about t will be contained in Ξ.

- e.g., $\Xi = \{t, t_1, \ldots, t_n\}$ and $\Xi' = \{t_1, \ldots, t_n\}$.

The augmented focused system LKF^a

INVERTIBLE RULES

$$\frac{\Xi' \vdash \Theta \Uparrow A, \Gamma \quad \Xi'' \vdash \Theta \Uparrow B, \Gamma \quad \mathsf{andNeg}_c(\Xi, \Xi', \Xi'')}{\Xi \vdash \Theta \Uparrow A \wedge^- B, \Gamma}$$

$$\frac{\Xi' \vdash \Theta \Uparrow A, B, \Gamma \quad \mathsf{orNeg}_c(\Xi, \Xi')}{\Xi \vdash \Theta \Uparrow A \vee^- B, \Gamma} \qquad \frac{(\Xi' y) \vdash \Theta \Uparrow [y/x]B, \Gamma \quad \mathsf{all}_c(\Xi, \Xi')}{\Xi \vdash \Theta \Uparrow \forall x.B, \Gamma} \dagger$$

FOCUSED RULES

$$\frac{\Xi' \vdash \Theta \Downarrow B_1 \quad \Xi'' \vdash \Theta \Downarrow B_2 \quad \mathsf{andPos}_e(\Xi, \Xi', \Xi'')}{\Xi \vdash \Theta \Downarrow B_1 \wedge^+ B_2}$$

$$\frac{\Xi' \vdash \Theta \Downarrow B_i \quad \mathsf{orPos}_e(\Xi, \Xi', i)}{\Xi \vdash \Theta \Downarrow B_1 \vee^+ B_2} \qquad \frac{\Xi' \vdash \Theta \Downarrow [t/x]B \quad \mathsf{exists}_e(\Xi, t, \Xi')}{\Xi \vdash \Theta \Downarrow \exists x.B}$$

IDENTITY RULES

$$\frac{\Xi' \vdash \Theta \Uparrow B \quad \Xi'' \vdash \Theta \Uparrow \neg B \quad \mathit{cut}_e(\Xi, \Xi', \Xi'', B)}{\Xi \vdash \Theta \Uparrow \cdot}\ \mathit{cut} \qquad \frac{\langle l, \neg P_a \rangle \in \Theta \quad \mathit{initial}_e(\Xi, l)}{\Xi \vdash \Theta \Downarrow P_a}\ \mathit{init}$$

STRUCTURAL RULES

$$\frac{\Xi' \vdash \Theta \Uparrow N \quad \mathit{release}_e(\Xi, \Xi')}{\Xi \vdash \Theta \Downarrow N}\ \mathit{release} \qquad \frac{\Xi' \vdash \Theta, \langle l, C \rangle \Uparrow \Gamma \quad \mathit{store}_c(\Xi, C, l, \Xi')}{\Xi \vdash \Theta \Uparrow C, \Gamma}\ \mathit{store}$$

$$\frac{\Xi' \vdash \Theta \Downarrow P \quad \langle l, P \rangle \in \Theta \quad \mathit{decide}_e(\Xi, l, \Xi')}{\Xi \vdash \Theta \Uparrow \cdot}\ \mathit{decide}$$

FPC specifications (1)

A proof is punctually represented by specifying:

1. at each step, on which formula we apply a rule (*decide*-predicate);
2. in the case of a $\Diamond$-formula, with respect to which label ($\exists$-predicate);
3. in the case of an initial, with respect to which complementary literal (*init*-predicate).

This gives rise to a punctual FPC specification:

- it allows for reconstructing the proof in a very faithful way;
- it might be not very concise.

FPC specifications (2)

We can only require some essential information:

1. a mapping between $\Box$- and $\Diamond$- formulas ($\exists$-predicate);
2. a mapping between complementary literals (*init*-predicate).

This gives rise to an essential FPC specification:

- it leaves the checker free of doing some not-driven reconstruction;
- it is less faithful but also more concise.

FPC specifications

The essential specification can be used also to check proofs in, e.g., free variable (FV) tableaux (where punctual is not possible).

$$
\begin{array}{c}
1 : \Diamond\neg p \lor \Diamond\neg q \\
| \\
1 : \Box(p \land q) \\
\begin{array}{cc}
1 : \Diamond\neg p & 1 : \Diamond\neg q \\
| & | \\
1.1 : \neg p & 1.2 : \neg q \\
| & | \\
1.1 : p \land q & 1.2 : p \land q \\
| & | \\
1.1 : p & 1.2 : p \\
| & | \\
1.1 : q & 1.2 : q
\end{array}
\end{array}
\qquad
\begin{array}{c}
1 : \Diamond\neg p \lor \Diamond\neg q \\
| \\
1 : \Box(p \land q) \\
| \\
1.x : p \land q \\
| \\
1.x : p \\
| \\
1.x : q \\
\begin{array}{cc}
1 : \Diamond\neg p & 1 : \Diamond\neg q \\
| & | \\
1.1 : \neg p & 1.2 : \neg q \\
| & | \\
x \mapsto 1 & x \mapsto 2
\end{array}
\end{array}
$$

Tests

Formalism	**Prover**	**Punctual FPC**	**Essential FPC**
Labeled sequents	by hand	V	V
Prefixed tableaux	ModLEAN-TAP	V	V
FV-tableaux	ModLEAN-TAP	X	V

Conclusion

In this paper

- Application of the use of a general framework for **proof checking/certification** to **modal logics**.
- Two different specifications for **prefixed tableau** proofs.

Current and future work

- Extension to modal logics represented by **geometric** frame properties.
- Extension to other formalisms:
  - **“unlabeled” sequent systems**;
  - **nested sequent systems**;
  - **hypersequent systems**;
  - **resolution methods**.

Thank you!