A proof-theoretical journey through programming, model checking and theorem proving

David Baelde

IT University of Copenhagen

ASL Meeting, Structural Proof Theory Session
Madison, Wisconsin, April 2012

Logic programming

A specification (Γ)

$$
\begin{array}{rl}
\forall k. & app\ nil\ k\ k \\
\forall x \forall l \forall k \forall m. & app\ l\ k\ m \supset app\ (x :: l)\ k\ (x :: m)
\end{array}
$$

Messy sequent calculus proofs

$$
\dfrac{\dfrac{\dfrac{\vdots}{\Gamma, \forall k \forall m.\ app\ [4]\ k\ m \supset app\ [3;4]\ k\ (3 :: m) \vdash app\ [0]\ nil\ [0]}}{\Gamma \vdash app\ [0]\ nil\ [0]}}{\dfrac{\Gamma, app\ nil\ [1;2;3]\ [1;2;3] \vdash app\ [0]\ nil\ [0]}{\Gamma \vdash app\ [0]\ nil\ [0]}}
$$

Logic programming

A specification (Γ)

$$
\begin{array}{rl}
\forall k. & app\ nil\ k\ k \\
\forall x \forall l \forall k \forall m. & app\ l\ k\ m \supset app\ (x :: l)\ k\ (x :: m)
\end{array}
$$

Focused proofs

$$
\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, app\ [0]\ nil\ [0] \vdash app\ [0]\ nil\ [0]} \quad \dfrac{\dfrac{\dfrac{}{\Gamma, app\ nil\ nil\ nil \vdash app\ nil\ nil\ nil}}{\Gamma, \forall k.\ app\ nil\ k\ k \vdash app\ nil\ nil\ nil}}{\Gamma \vdash app\ nil\ nil\ nil}}{\Gamma, app\ nil\ nil\ nil \supset app\ [0]\ nil\ [0] \vdash app\ [0]\ nil\ [0]}}{\Gamma, \forall x \forall k \forall l \forall m.\ \ldots \vdash app\ [0]\ nil\ [0]}}{\Gamma \vdash app\ [0]\ nil\ [0]}
$$

Logic programming

A specification (Γ)

$$\begin{array}{rl} \forall k. & app\ nil\ k\ k \\ \forall x \forall l \forall k \forall m. & app\ l\ k\ m \supset app\ (x :: l)\ k\ (x :: m) \end{array}$$

Focused proofs

$$\dfrac{\dfrac{}{\Gamma \vdash app\ nil\ nil\ nil}\ \forall L, init}{\Gamma \vdash app\ [0]\ nil\ [0]}\ \forall L, \supset L, init$$

Fixed Points

Computation

Rules

$$\frac{\Gamma \vdash B(\mu B)\vec{t}}{\Gamma \vdash \mu B\vec{t}}$$

Specification

$$\begin{aligned} app \stackrel{def}{=} \quad \mu(\lambda A \lambda l \lambda k \lambda m. \quad & (l = nil \wedge k = m) \\ \vee \qquad & (\exists x \exists l' \exists m'.\, l = x :: l' \wedge m = x :: m' \wedge A\ l'\ k\ m')) \end{aligned}$$

Computing

$$\dfrac{\dfrac{}{\vdash [0] = [0]} {=}R \quad \dfrac{}{\vdash [0] = [0]} {=}R \quad \dfrac{}{\vdash app\ nil\ nil\ nil} \mu R, \vee R, {=}R}{\dfrac{\vdash [0] = [0] \wedge [0] = [0] \wedge app\ nil\ nil\ nil}{\vdash app\ [0]\ nil\ [0]} \mu R, \vee R, \exists R} \wedge R$$

Computation

Rules

$$\frac{\Gamma \vdash B(\mu B)\vec{t}}{\Gamma \vdash \mu B\vec{t}}$$

Specification

$$\begin{aligned} app \stackrel{def}{=} \quad \mu(\lambda A \lambda l \lambda k \lambda m. \quad & (l = nil \wedge k = m) \\ \vee \qquad & (\exists x \exists l' \exists m'.\, l = x :: l' \wedge m = x :: m' \wedge A\ l'\ k\ m')) \end{aligned}$$

Computing

$$\frac{}{\vdash app\ [0]\ nil\ [0]}\ \mu R, \vee R, \exists R, = R$$

Finite reasoning

Rules

$$\frac{\Gamma, B(\mu B)\vec{t} \vdash P}{\Gamma, \mu B\vec{t} \vdash P} \qquad \frac{\Gamma \vdash B(\mu B)\vec{t}}{\Gamma \vdash \mu B\vec{t}}$$

Reasoning by computing

$$\dfrac{\dfrac{\overline{x :: l = nil, k = nil \vdash \bot} \quad \overline{x :: l = x :: l', nil = x :: m', app\ l'\ k\ m' \vdash \bot}}{app\ (x :: l)\ k\ nil \vdash \bot}}{\vdash \forall x, l, k.\ app\ (x :: l)\ k\ nil \supset \bot}$$

More examples: connectedness, path unicity, (bi)simulation…
for finite systems.

Finite reasoning

Rules

$$\frac{\Gamma, B(\mu B)\vec{t} \vdash P}{\Gamma, \mu B\vec{t} \vdash P} \qquad \frac{\Gamma \vdash B(\mu B)\vec{t}}{\Gamma \vdash \mu B\vec{t}}$$

Reasoning by computing

$$\dfrac{\dfrac{\vdots}{\vdash node\ C} \qquad \dfrac{\ldots \quad \dfrac{\vdots}{\vdash path\ C\ N_i} \quad \ldots}{\vdash \forall N.\ node\ N \supset path\ C\ N}}{\vdash \exists C.\ node\ C \wedge \forall N.\ node\ N \supset path\ C\ N}$$

More examples: connectedness, path unicity, (bi)simulation...
for finite systems.

Infinity (identity)

Rules

$$\frac{\Gamma, B(\mu B)\vec{t} \vdash P}{\Gamma, \mu B\vec{t} \vdash P} \qquad \frac{\Gamma \vdash B(\mu B)\vec{t}}{\Gamma \vdash \mu B\vec{t}}$$

$$\frac{}{\Gamma, \mu B\vec{t} \vdash \mu B\vec{t}}$$

Infinity (identity)

Rules

$$\frac{\Gamma, B(\mu B)\vec{t} \vdash P}{\Gamma, \mu B\vec{t} \vdash P} \qquad \frac{\Gamma \vdash B(\mu B)\vec{t}}{\Gamma \vdash \mu B\vec{t}}$$

$$\frac{\Gamma, \mu B\vec{t} \vdash P}{\Gamma, \mu B\vec{t} \vdash P} \qquad \frac{}{\Gamma, \mu B\vec{t} \vdash \mu B\vec{t}}$$

Infinity (identity)

Rules

$$\frac{\Gamma, B(\mu B)\vec{t} \vdash P}{\Gamma, \mu B\vec{t} \vdash P} \qquad \frac{\Gamma \vdash B(\mu B)\vec{t}}{\Gamma \vdash \mu B\vec{t}}$$

$$\frac{\Gamma, \mu B\vec{t} \vdash P}{\Gamma, \mu B\vec{t} \vdash P} \qquad \frac{}{\Gamma, \mu B\vec{t} \vdash \mu B\vec{t}}$$

Example

$$\dfrac{\dfrac{\dfrac{\dfrac{}{nat\ x \vdash nat\ x}}{nat\ x \vdash nat\ (s^{10}\ x)}}{nat\ x \vdash nat\ (s^{10}\ x)}}{nat\ (s^{3}\ x) \vdash nat\ (s^{10}\ x)}$$

Infinity (induction)

Rules

$$\frac{\Gamma, S\vec{t} \vdash P \quad BS\vec{x} \vdash S\vec{x}}{\Gamma, \mu B\vec{t} \vdash P} \qquad \frac{\Gamma \vdash B(\mu B)\vec{t}}{\Gamma \vdash \mu B\vec{t}}$$

$$\frac{\Gamma, \mu B\vec{t} \vdash P}{\Gamma, \mu B\vec{t} \vdash P} \qquad \frac{}{\Gamma, \mu B\vec{t} \vdash \mu B\vec{t}}$$

Example (Derived rules for *nat*)

$$nat\ x \stackrel{def}{=} \mu(\lambda N \lambda x.\ x = 0 \lor \exists y.\ x = s\ y \land N\ y)x$$

$$\frac{}{\Gamma \vdash nat\ 0} \qquad \frac{\Gamma \vdash nat\ x}{\Gamma \vdash nat\ (s\ x)}$$

$$\frac{\vdash P\ 0 \quad P\ y \vdash P\ (s\ y) \quad \Gamma, P\ x \vdash G}{\Gamma, nat\ x \vdash G}$$

Infinity (coinduction)

Rules

$$\frac{\Gamma \vdash S\vec{t} \quad S\vec{x} \vdash BS\vec{x}}{\Gamma \vdash \nu B\vec{t}} \qquad \frac{\Gamma, B(\nu B)\vec{t} \vdash P}{\Gamma, \nu B\vec{t} \vdash P}$$

$$\frac{\Gamma \vdash \nu B\vec{t}}{\Gamma \vdash \nu B\vec{t}} \qquad \frac{}{\Gamma, \nu B\vec{t} \vdash \nu B\vec{t}}$$

Example (Derived rules for *sim*)

$$sim \stackrel{def}{=} \nu(\lambda S \lambda p \lambda q.\ \forall \alpha \forall p'.step\ p\ \alpha\ p' \supset \exists q'.step\ q\ \alpha\ q' \wedge S\ p'\ q')$$

$$\frac{\Gamma \vdash step\ p\ \alpha\ p' \quad \Gamma, step\ q\ \alpha\ q', sim\ p'\ q' \vdash P}{\Gamma, sim\ p\ q \vdash P}$$

$$\frac{\Gamma \vdash R\ p\ q \quad R\ p\ q, step\ p\ \alpha\ p' \vdash \exists q'.\ step\ q\ \alpha\ q' \wedge R\ p'\ q'}{\Gamma \vdash sim\ p\ q}$$

Fixed Points in Proof Theory

Foundations

- Natural generic rules, various ambient calculi
- Completeness of focused systems [Baelde & Miller ’07]
- Cut elimination [Baelde ’10]
- Game semantics for μLJ proofs [Clairambault ’09]

Related Work

- Definitions (SH 93, MM 00, MT 03)
- Type theory (Mendler 91, Matthes 99, Paulin)
- Cyclic proofs (…Santocanale 01, Brotherston 05)
- μ-calculus, Kleene algebras…

Applications

Abella & Tac

- Interactive theorem provers for μLJ
- Extensions for reasoning about binding (esp. Abella)
- Tac: automated focused (co)inductive theorem proving

Bedwyr

- "model checking" over syntactic specifications
- finite behavior proofs, "prolog + exhaustive case analyses"
- example: bisimulation checker for π, spi (Miller & Tiu, Tiu)
- tabling and cyclic proofs

Proof & Verification

…not “proof $\otimes$ verification”.

Motivations

Practical

- Independently checkable certificates
- Not too ad-hoc, composable: proofs
- Compute: run a certificate on examples (synthesis)
- Interoperate: mix automatic and interactive theorem proving, certify abstraction and verify it, combine partial correctness and termination...

Fundamental

- Completeness, decidability results, proof structures
- More algebraic viewpoint on automata techniques

Model-checking

Verification

- Does a system satisfy a specification?
- $M \models S$
- Often translated to automata inclusion $[M] \subseteq [S]$

How do you prove an inclusion?

$$[M]x \vdash [S]x$$

What is the structure of inclusion?

NFA: Definitions

Non-deterministic finite automata

- Alphabet $\Sigma = \{\alpha, \beta, \gamma, \ldots\}$
- Finite set of states
- Distinguished initial and final states
- Transition relation $s \rightarrow^{\alpha} q$

Definition

If Q is a set of states,
$Q \rightarrow^{\alpha} Q'$ iff each state of Q' is reachable from Q.
In other words, $Q' \subseteq \alpha^{-1} Q$.

Structure of inclusion

Definition (Multi-simulation)

A multi-simulation between two automata (A, T, I, F) and (B, T', I', F') is a relation $\mathfrak{R} \subseteq A \times \wp(B)$ such that whenever $p\mathfrak{R}Q$:

- if p is final, then there must be a final state in Q;
- for any α and p' such that $p \to^{\alpha} p'$
  there exists Q' such that $Q \to^{\alpha} Q'$ and $p'\mathfrak{R}Q'$.

Multi-simulations are post-fixed points.
There is a greatest one: call it multi-similarity.

Proposition (Multi-similarity is inclusion)

$\mathcal{L}(p) \subseteq \mathcal{L}(Q)$ if and only if $p\mathfrak{R}Q$ for some multi-simulation $\mathfrak{R}$.

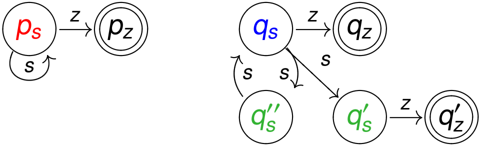
Example: $\forall x.\ nat\ x \supset even\ x \vee odd\ x$

Consider the following two automata:

State p_0 is included in q_0. Proof:

$$\mathfrak{R} = \{(p_0, \{q_0\}), (p_1, \{q_1, q_1'\}), (p_2, \{q_2\})\}$$

Example: $\forall x.\ nat\ x \supset \exists h.\ half\ x\ h$



Proof of $\mathcal{L}(p_s) \subseteq \mathcal{L}(q_s)$:

$$\mathfrak{R} = \{(p_s, \{q_s\}), (p_s, \{q_s', q_s''\}), (p_z, \{q_z\}), (p_z, \{q_z'\})\}$$

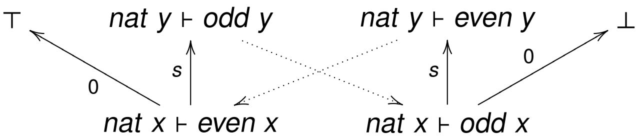
Extended cyclic proofs / tabled search

$$
\dfrac{
\boxed{\dfrac{\dfrac{}{\vdash \mathit{even}\ 0} \qquad \dfrac{\dfrac{\infty}{\mathit{nat}\ y \vdash \mathit{odd}\ y}}{\mathit{nat}\ y \vdash \mathit{even}\ (sy)}}{\mathit{nat}\ x \vdash \mathit{even}\ x}}
\ \oplus\
\boxed{\dfrac{\dfrac{\bot}{\vdash \mathit{odd}\ 0} \qquad \dfrac{\dfrac{\infty}{\mathit{nat}\ y \vdash \mathit{even}\ y}}{\mathit{nat}\ y \vdash \mathit{odd}\ (sy)}}{\mathit{nat}\ x \vdash \mathit{odd}\ x}}
}{\mathit{nat}\ x \vdash \mathit{even}\ x \oplus \mathit{odd}\ x}
$$

Extended cyclic proofs / tabled search

$$\dfrac{\boxed{\dfrac{\dfrac{}{\vdash \mathit{even}\ 0} \qquad \dfrac{\dfrac{\infty}{\mathit{nat}\ y \vdash \mathit{odd}\ y}}{\mathit{nat}\ y \vdash \mathit{even}\ (sy)}}{\mathit{nat}\ x \vdash \mathit{even}\ x}} \oplus \boxed{\dfrac{\dfrac{\bot}{\vdash \mathit{odd}\ 0} \qquad \dfrac{\dfrac{\infty}{\mathit{nat}\ y \vdash \mathit{even}\ y}}{\mathit{nat}\ y \vdash \mathit{odd}\ (sy)}}{\mathit{nat}\ x \vdash \mathit{odd}\ x}}}{\mathit{nat}\ x \vdash \mathit{even}\ x \oplus \mathit{odd}\ x}$$

This is not quite a proof but realizes one:
the underlying automata covers all cases, *i.e.,* contains *nat*.



Semi-decidability, generating invariants and μLJ proofs

Conclusion

Proof theory of fixed points

- Very rich logics
- Precise proof theoretical analysis
- Wider range of applications, supported by focusing

More proof & verification

- Extend: Büchi, tree and alternating automata
- Automated (co)inductive reasoning, loop schemes in Bedwyr