

# Reconstructing Propositional Proofs in Type Theory

Jonathan Prieto-Cubides  
Advisor: Andrés Sicard-Ramírez

Master in Applied Mathematics  
Universidad EAFIT  
Medellín, Colombia

November 16, 2017  
Update: December 2, 2017



## Goal

Formalization in type theory, classical propositional derivations generated by the Metis theorem prover.

## Goal

Formalization in type theory, classical propositional derivations generated by the `Metis` theorem prover.

## Topics

- ▶ Automatic reasoning using automatic theorem provers (ATPs) (e.g., `Metis`, `EProver`)
- ▶ Interactive proving using proof-assistants (e.g., `Agda`, `Coq`)
- ▶ Proof-reconstruction for proofs generated by ATPs in proof-assistants

# Research Outcomes

Academic result: paper (work in progress)

Software related results:

- ▶ **Athena**: a translator tool for Metis proofs to Agda in Haskell<sup>1</sup>
- ▶ Agda libraries:
  - ▶ **agda-metis**: Metis prover reasoning for propositional logic<sup>2</sup>
  - ▶ **agda-prop**: intuitionistic propositional logic + PEM<sup>3</sup>
- ▶ Bugs found in Metis: see Issue No. 2, Issue No. 4, and Commit 8a3f11e from the Metis official repository<sup>4</sup>

In parallel, we develop:

- ▶ **Online-ATPs**: a web client for the TPTP world in Haskell<sup>5</sup>
- ▶ **Prop-Pack**: compendium of TPTP problems in classical propositional logic used to test Athena<sup>6</sup>

---

<sup>1</sup><https://github.com/jonaprieto/athena>.

<sup>2</sup><https://github.com/jonaprieto/agda-metis>.

<sup>3</sup><https://github.com/jonaprieto/agda-prop>.

<sup>4</sup><https://github.com/gilith/metis>.

<sup>5</sup><https://github.com/jonaprieto/online-atps>.

<sup>6</sup><https://github.com/jonaprieto/prop-pack>.

# Bug in the Printing of the Proof

Fixed in Metis v2.3 (release 20161108)

$$\varphi := \neg p \wedge (\neg q \Leftrightarrow \neg r) \wedge (\neg p \Leftrightarrow (\neg q \Leftrightarrow \neg r))$$

$$\frac{\frac{\frac{\vdots}{\varphi} \text{ canonicalize}}{\neg p \Leftrightarrow (\neg q \Leftrightarrow \neg r)} \text{ conjunct} \quad \frac{\frac{\frac{\vdots}{\varphi} \text{ canonicalize}}{\neg q \Leftrightarrow \neg r} \text{ conjunct} \quad \frac{\frac{\frac{\vdots}{\varphi} \text{ canonicalize}}{\neg p} \text{ conjunct}}{\text{simplify}}}{\perp}$$

# Bug in the Printing of the Proof

Fixed in Metis v2.3 (release 20161108)

$$\varphi := \neg p \wedge (\neg q \Leftrightarrow \neg r) \wedge (\neg p \Leftrightarrow (\neg q \Leftrightarrow \neg r))$$

$$\frac{\frac{\frac{\vdots}{\varphi} \text{ canonicalize}}{\neg p \Leftrightarrow (\neg q \Leftrightarrow \neg r)} \text{ conjunct} \quad \frac{\frac{\frac{\vdots}{\varphi} \text{ canonicalize}}{\neg q \Leftrightarrow \neg r} \text{ conjunct} \quad \frac{\frac{\frac{\vdots}{\varphi} \text{ canonicalize}}{\neg p} \text{ conjunct}}{\neg p} \text{ simplify}}{\perp}$$

The bug was caused by the conversion of Xor sets to Iff lists. After reporting this, Metis developer fixed the printing of canonicalize inference rule

$$\varphi := \neg p \wedge (\neg q \Leftrightarrow \neg r) \wedge (\neg p \Leftrightarrow (\neg q \Leftrightarrow \textcolor{red}{r}))$$

# Soundness Bug in Splitting Goals

Fixed in Metis v2.3 (release 20170810)

Consider this TPTP problem

```
$ cat issue.tptp
fof(goal, conjecture,
    (~ (p <=> q)) <=> ((p => ~ q) & (q => ~p))).
```

Metis found a proof of this problem and it is not a tautology.

```
$ metis issue.tptp
SZS status Theorem for issue.tptp
```

| p | q | $\neg (p \Leftrightarrow q) \Leftrightarrow ((p \supset \neg q) \wedge (q \supset \neg p))$ |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T | T | ⊥   | T | T | T | T | T | ⊥ | ⊥ | T | ⊥ | T | ⊥ | ⊥ | T |
| T | ⊥ | T   | T | ⊥ | ⊥ | T | T | T | ⊥ | T | ⊥ | T | ⊥ | ⊥ | T |
| ⊥ | T | T   | ⊥ | ⊥ | T | T | ⊥ | T | ⊥ | T | T | T | T | T | ⊥ |
| ⊥ | ⊥ | ⊥   | ⊥ | T | ⊥ | ⊥ | T | T | ⊥ | T | ⊥ | T | T | T | ⊥ |

# Soundness Bug in Splitting Goals

Fixed in Metis v2.3 (release 20170810)

We detected the bug by reconstructing the `strip` inference rule. The bug was solved changing this formula:

$$\neg (p \Leftrightarrow q) \Leftrightarrow ((p \Rightarrow \neg q) \wedge (q \Rightarrow \neg p))$$

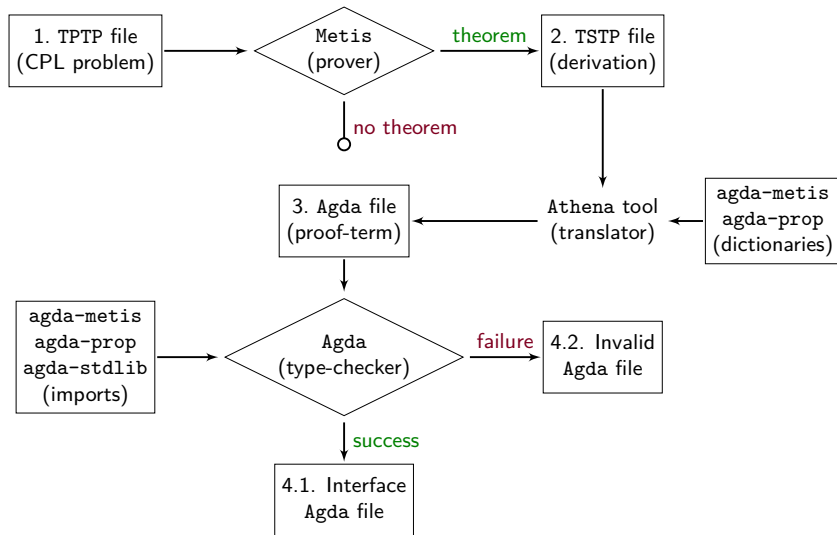
by the following tautology

$$\neg (p \Leftrightarrow q) \Leftrightarrow ((p \Rightarrow \neg q) \wedge (\neg q \Rightarrow p))$$

| p | q | $\neg (p \Leftrightarrow q) \Leftrightarrow ((p \Rightarrow \neg q) \wedge (\neg q \Rightarrow p))$ |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T | T | ⊥   | T | T | T | T | T | ⊥ | ⊥ | T | ⊥ | ⊥ | T | T | T |
| T | ⊥ | T   | T | ⊥ | ⊥ | T | T | T | ⊥ | T | T | ⊥ | T | T |   |
| ⊥ | T | T   | ⊥ | ⊥ | T | ⊥ | T | ⊥ | T | T | ⊥ | T | T | ⊥ |   |
| ⊥ | ⊥ | ⊥   | ⊥ | T | ⊥ | ⊥ | T | T | ⊥ | ⊥ | T | ⊥ | ⊥ | ⊥ |   |



# Proof Reconstruction: Overview



# Inference Rules of Metis

TSTP derivations by Metis exhibit the following inferences:

| Metis rule   | Purpose                                  |
|--------------|--|
| strip        | Strip a goal into subgoals               |
| conjunct     | Takes a formula from a conjunction       |
| resolve      | A general form of the resolution theorem |
| canonicalize | Normalization of the formula             |
| clausify     | Performs clausification                  |
| simplify     | Simplify definitions and theorems        |

# Proposition Type in Agda

A data type for formulas

```
data PropFormula : Set where
  Var : Fin n → Prop
  T    : Prop
  ⊥    : Prop
  _∧_  : (φ ψ : Prop) → Prop
  _∨_  : (φ ψ : Prop) → Prop
  _⇒_  : (φ ψ : Prop) → Prop
  _⇔_  : (φ ψ : Prop) → Prop
  ¬_   : (φ : Prop)   → Prop
```

# Inference Rules For Propositional Logic I

Intuitionistic Propositional Logic + PEM ( $\Gamma \vdash \varphi \vee \neg \varphi$ )

$$\frac{}{\Gamma, \varphi \vdash \varphi} \text{ assume}$$

$$\frac{}{\Gamma \vdash \top} \top\text{-intro}$$

$$\frac{}{\Gamma \vdash \varphi \vee \neg \varphi} \text{ PEM}$$

# Inference Rules For Propositional Logic II

$$\frac{\Gamma \vdash \perp}{\Gamma \vdash \varphi} \perp\text{-elim}$$

$$\frac{\Gamma, \varphi \vdash \perp}{\Gamma \vdash \neg \varphi} \neg\text{-intro}$$

$$\frac{\Gamma \vdash \neg \varphi \quad \Gamma \vdash \varphi}{\Gamma \vdash \perp} \neg\text{-elim}$$

# Inference Rules For Propositional Logic III

$$\frac{\Gamma \vdash \varphi \quad \Gamma \vdash \psi}{\Gamma \vdash \varphi \wedge \psi} \wedge\text{-intro}$$

$$\frac{\Gamma \vdash \varphi \wedge \psi}{\Gamma \vdash \varphi} \wedge\text{-proj}_1$$

$$\frac{\Gamma \vdash \varphi \wedge \psi}{\Gamma \vdash \psi} \wedge\text{-proj}_2$$

# Inference Rules For Propositional Logic IV

$$\frac{\Gamma \vdash \varphi}{\Gamma \vdash \varphi \vee \psi} \vee\text{-intro}_1$$

$$\frac{\Gamma \vdash \psi}{\Gamma \vdash \varphi \vee \psi} \vee\text{-intro}_2$$

$$\frac{\Gamma, \varphi \vdash \gamma \quad \Gamma, \psi \vdash \gamma}{\Gamma, \varphi \vee \psi \vdash \gamma} \vee\text{-elim}$$

$$\frac{\Gamma, \varphi \vdash \psi}{\Gamma \vdash \varphi \Rightarrow \psi} \Rightarrow\text{-intro}$$

$$\frac{\Gamma \vdash \varphi \Rightarrow \psi \quad \Gamma \vdash \varphi}{\Gamma \vdash \psi} \Rightarrow\text{-elim}$$

# Other Rules

- ▶ Weakening: to extend the hypotheses with additional formulas

$$\frac{\Gamma \vdash \varphi}{\Gamma, \psi \vdash \varphi} \text{ weaken}$$

- ▶ The RAA rule is the formulation of the principle of proof by contradiction:

$$\frac{\Gamma, \neg \varphi \vdash \perp}{\Gamma \vdash \varphi} \text{ RAA}$$



# Syntactical Consequence Relation in Agda

- ▶ Inductive family  $\_ \vdash \_$  with two indexes: a set of propositions  $\Gamma$  (the premises) and a proposition  $\varphi$  (the conclusion)

## Example

In [8] we define  $\_ \vdash \_$  as follows

```
data _⊢_ : (Γ : Ctxt)(φ : Prop) → Set
...
^⊢-intro
  : ∀ {Γ} {φ ψ}
  → Γ ⊢ φ → Γ ⊢ ψ
  → Γ ⊢ φ ∧ ψ

^⊢-proj1
  : ∀ {Γ} {φ ψ}
  → Γ ⊢ φ ∧ ψ
  → Γ ⊢ φ

^⊢-proj2
  : ∀ {Γ} {φ ψ}
  → Γ ⊢ φ ∧ ψ
  → Γ ⊢ ψ
...
```

# Reconstructing Metis Rules in Type Theory

Let `metisRule` be a `Metis` inference rule. We define the function `metisRule` in type theory which has the following pattern<sup>7</sup>:

`metisRule` : `Premise`  $\rightarrow$  `Conclusion`  $\rightarrow$  `Prop`

$$\text{metisRule } \varphi \ \psi = \begin{cases} \psi, & \text{if the conclusion } \psi \text{ can be derived by applying} \\ & \text{certain inference rules to the premise } \varphi; \\ \varphi, & \text{otherwise;} \end{cases}$$

To justify all transformations done by the `metisRule` rule, we prove its soundness with a theorem like the following:

If  $\Gamma \vdash \varphi$  then  $\Gamma \vdash \text{metisRule } \varphi \ \psi$ .

---

<sup>7</sup>`Premise` and `Conclusion` as synonyms of the `Prop` type to describe in the function types the role of the arguments

# Example

The `clausify` rule transforms a formula into its clausal normal form.

## Example

In the following TSTP derivation by Metis, we see how `clausify` transforms the `norm0` formula to get `norm1` formula.

```
fof(norm0,  $\neg p \vee (q \wedge r)$  ...  
fof(norm1,  $(\neg p \vee q) \wedge (\neg p \vee r)$ , inf(clausify, norm0)).
```

## Theorem

*Let  $\psi$  : Conclusion. If  $\Gamma \vdash \varphi$  then  $\Gamma \vdash \text{clausify } \varphi \ \psi$ , where*

`clausify` : Premise  $\rightarrow$  Conclusion  $\rightarrow$  Prop

$$\text{clausify } \varphi \ \psi = \begin{cases} \psi, & \text{if } \varphi \equiv \psi; \\ \text{reorder}_{\wedge \vee} (\text{cnf } \varphi) \ \psi, & \text{otherwise.} \end{cases}$$

# Sketch of the Metis Algorithm

---

**Algorithm 1** Metis refutation strategy

---

**procedure** METIS

**input:** the goal and a set of *premises*  $a_1, \dots, a_n$

**output:** maybe a derivation when  $a_1, \dots, a_n \vdash \text{goal}$ , otherwise nothing.

strip the goal into a list of *subgoals*  $s_i$

**for** each subgoal  $s_i$  **do**

try to find by a refutation for  $\neg s_i$ :

    apply clausification for the negated subgoal  $\neg s_i$

**if** a premise  $a_j$  is relevant **then**

        apply clausification to  $a_j$

**end if**

        application of Metis inference rules

**if** a contradiction can be derived from the assumptions **then**

        keep the refutation and continue with the others subgoals

**else**

        exit without a proof.

**end if**

**end for**

print the conjecture and the premises

print each refutation for each negated subgoal

**end procedure**

---

# Some Challenges

- ▶ Formalization
  - ▶ Understanding the `Metis` reasoning without a proper documentation or description from the `Metis` author
  - ▶ Terminating of functions that reconstruct `Metis` inference rules
  - ▶ Intuitionistic logic implementation
- ▶ Software related
  - ▶ Parsing of TSTP derivations
  - ▶ Printing valid Agda files

# Complete Example

The problem<sup>8</sup>:

$$(p \Rightarrow q) \wedge (q \Rightarrow p) \vdash (p \vee q) \Rightarrow (p \wedge q)$$

In TPTP syntax:

```
fof(a1, axiom, (p => q) ^ (q => p)).  
fof(goal, conjecture, (p v q) => (p ^ q)).
```

Its TSTP solution using Metis:

```
fof(a1, axiom, (p => q) ^ (q => p)).  
fof(goal, conjecture, (p v q) => (p ^ q)).  
fof(s1, (p v q) => p, inf(strip, goal)).  
fof(s2, ((p v q) ^ p) => q, inf(strip, goal)).  
...
```

---

<sup>8</sup>Problem No. 13 in Disjunction Section in [7]

```

fof(premise, axiom, (p q) ^ (q p)).
fof(goal, conjecture, (p v q) (p ^ q)).
fof(s0, (p v q) p, inf(strip, goal)).
fof(s1, ((p v q) ^ p) q, inf(strip, goal)).
fof(neg0, ¬ ((p v q) p), inf(negate, s0)).
fof(n00, (¬ p v q) ^ (¬ q v p), inf(canonicalize, premise)).
fof(n01, ¬ q v p, inf(conjunct, n00)).
fof(n02, ¬ p ^ (p v q), inf(canonicalize, neg0)).
fof(n03, p v q, inf(conjunct, n02)).
fof(n04, ¬ p, inf(conjunct, n02)).
fof(n05, q, inf(simplify, [n03, n04])).
cnf(r00, ¬ q v p, inf(canonicalize, n01)).
cnf(r01, q, inf(canonicalize, n05)).
cnf(r02, p, inf(resolve, q, [r01, r00])).
cnf(r03, ¬ p, inf(canonicalize, n04)).
cnf(r04, ⊥, inf(resolve, p, [r02, r03])).
fof(neg1, ¬ ((p v q) ^ p) q, inf(negate, s1)).
fof(n10, ¬ q ^ p ^ (p v q), inf(canonicalize, neg1)).
fof(n11, (¬ p v q) ^ (¬ q v p), inf(canonicalize, premise)).
fof(n12, ¬ p v q, inf(conjunct, n11)).
fof(n13, ⊥, inf(simplify, [n10, n12])).
cnf(r10, ⊥, inf(canonicalize, n13)).

```

# TSTP Refutation of Subgoal No. 1

```
fof(premise, axiom, (p  $\supset$  q)  $\wedge$  (q  $\supset$  p)).  
fof(goal, conjecture, (p  $\vee$  q)  $\supset$  (p  $\wedge$  q)).  
fof(s0, (p  $\vee$  q)  $\supset$  p, inf(strip, goal)).  
...  
fof(neg0,  $\neg$  ((p  $\vee$  q)  $\supset$  p), inf(negate, s0)).  
fof(n00, ( $\neg$  p  $\vee$  q)  $\wedge$  ( $\neg$  q  $\vee$  p), inf(canonicalize, premise)).  
fof(n01,  $\neg$  q  $\vee$  p, inf(conjunct, n00)).  
fof(n02,  $\neg$  p  $\wedge$  (p  $\vee$  q), inf(canonicalize, neg0)).  
fof(n03, p  $\vee$  q, inf(conjunct, n02)).  
fof(n04,  $\neg$  p, inf(conjunct, n02)).  
fof(n05, q, inf(simplify, [n03, n04])).  
cnf(r00,  $\neg$  q  $\vee$  p, inf(canonicalize, n01)).  
cnf(r01, q, inf(canonicalize, n05)).  
cnf(r02, p, inf(resolve, q, [r01, r00])).  
cnf(r03,  $\neg$  p, inf(canonicalize, n04)).  
cnf(r04,  $\bot$ , inf(resolve, p, [r02, r03])).
```



# Refutation Tree for $s_0$

```

fof(premise, axiom, (p  q) ∧ (q  p)).
...
fof(n00, (¬ p ∨ q) ∧ (¬ q ∨ p), inf(canonicalize, premise)).
fof(n01, ¬ q ∨ p, inf(conjunct, n00)).
...

```

$$(\mathcal{D}_1) \quad \frac{\frac{\frac{\Gamma \vdash (p \Rightarrow q) \wedge (q \Rightarrow p)}{\Gamma, \neg s_1 \vdash (p \Rightarrow q) \wedge (q \Rightarrow p)} \text{axiom premise}}{\Gamma, \neg s_1 \vdash (\neg p \vee q) \wedge (\neg q \vee p)} \text{weaken}}{\Gamma, \neg s_1 \vdash \neg q \vee p} \text{canonicalize} \text{conjunct}$$

```

...
fof(s1, (p ∨ q) ⇒ p, inf(strip, goal)).
fof(neg1, ¬ ((p ∨ q) ⇒ p), inf(negate, s1)).
...
fof(n02, ¬ p ∧ (p ∨ q), inf(canonicalize, neg1)).
fof(n03, p ∨ q, inf(conjunct, n02)).
fof(n04, ¬ p, inf(conjunct, n02)).
...

```

$$(\mathcal{D}_2) \quad \frac{\frac{\overline{\Gamma, \neg s_1 \vdash \neg s_1} \text{ assume}}{\Gamma, \neg s_1 \vdash \neg p \wedge (p \vee q)} \text{ canonicalize}}{\Gamma, \neg s_1 \vdash p \vee q} \text{ conjunct}$$

$$(\mathcal{D}_3) \quad \frac{\frac{\frac{\overline{\Gamma, \neg s_1 \vdash \neg s_1} \text{ assume } \neg s_1}{\Gamma, \neg s_1 \vdash \neg p \wedge (p \vee q)} \text{ canonicalize}}{\Gamma, \neg s_1 \vdash \neg p} \text{ conjunct}$$

$$(\mathcal{D}_4) \quad \frac{\frac{\mathcal{D}_2}{\Gamma, \neg s_1 \vdash p \vee q} \quad \frac{\mathcal{D}_3}{\Gamma, \neg s_1 \vdash \neg p}}{\Gamma, \neg s_1 \vdash q} \text{ simplify}$$

$$(\mathcal{R}_1) \quad \frac{\frac{\frac{\mathcal{D}_1}{\Gamma, \neg s_1 \vdash \neg q \vee p} \quad \frac{\mathcal{D}_4}{\Gamma, \neg s_1 \vdash q}}{\Gamma, \neg s_1 \vdash p} \text{ resolve } q \quad \frac{\mathcal{D}_3}{\Gamma, \neg s_1 \vdash \neg p}}{\Gamma, \neg s_1 \vdash \perp} \text{ resolve } p$$

$$\frac{\Gamma, \neg s_1 \vdash \perp}{\Gamma \vdash s_1} \text{ RAA}$$

## Tree for the Subgoal No. 2: $((p \vee q) \wedge p) \Rightarrow q$

```

fof(s2, ((p ∨ q) ∧ p) ⇒ q, inf(strip, goal)).
fof(neg2, ¬ (((p ∨ q) ∧ p) ⇒ q), inf(negate, s2)).
fof(n10, ¬ q ∧ p ∧ (p ∨ q), inf(canonicalize, neg2)).
fof(n11, (¬ p ∨ q) ∧ (¬ q ∨ p), inf(canonicalize, a1)).
fof(n12, ¬ p ∨ q, inf(conjunct, n11)).
fof(n13, ⊥, inf(simplify, [n10, n12])).
cnf(r10, ⊥, inf(canonicalize, n13)).

```

$$\begin{array}{c}
 \text{axiom } a_1 \\
 \hline
 \Gamma \vdash (p \Rightarrow q) \wedge (q \Rightarrow p) \\
 \hline
 \text{weaken} \\
 \hline
 \Gamma, \neg s_2 \vdash (p \Rightarrow q) \wedge (q \Rightarrow p) \\
 \hline
 \text{canonicalize} \\
 \hline
 \Gamma, \neg s_2 \vdash (\neg p \vee q) \wedge (\neg q \vee p) \\
 \hline
 \text{conjunct} \\
 \hline
 \Gamma, \neg s_2 \vdash \neg p \vee q \\
 \hline
 \text{simplify} \\
 \hline
 \Gamma, \neg s_2 \vdash \perp \\
 \hline
 \text{RAA} \\
 \hline
 \Gamma \vdash s_2
 \end{array}$$

$\frac{\Gamma, \neg s_2 \vdash \neg s_2 \quad \text{assume } (\neg s_2)}{\Gamma, \neg s_2 \vdash \neg q \wedge p \wedge (p \vee q)} \text{ canonicalize}$

$(\mathcal{R}_2) \quad \frac{\Gamma, \neg s_2 \vdash \neg q \wedge p \wedge (p \vee q) \quad \Gamma, \neg s_2 \vdash \neg p \vee q}{\Gamma, \neg s_2 \vdash \perp} \text{ canonicalize}$

# Summarizing the Example

The problem was:

$$(p \Rightarrow q) \wedge (q \Rightarrow p) \vdash (p \vee q) \Rightarrow (p \wedge q)$$

Its TSTP solution using Metis was:

```
fof(a1, axiom, (p ⇒ q) ∧ (q ⇒ p)).  
fof(goal, conjecture, (p ∨ q) ⇒ (p ∧ q)).  
fof(s1, (p ∨ q) ⇒ p, inf(strip, goal)).  
fof(s2, ((p ∨ q) ∧ p) ⇒ q, inf(strip, goal)).  
...
```

The proof is:

$$\frac{\frac{\Gamma \vdash (s_1 \wedge s_2) \Rightarrow \text{goal}}{\text{strip}} \quad \frac{\frac{\mathcal{R}_1}{\Gamma \vdash s_1} \quad \frac{\mathcal{R}_2}{\Gamma \vdash s_2}}{\Gamma \vdash s_1 \wedge s_2} \wedge\text{-intro}}{\Gamma \vdash \text{goal}} \Rightarrow\text{-elim}$$

(Live example using Agda and Athena)

# Future Work

Further research directions include, but are not limited to:

- ▶ improve the performance of the `canonicalize` rule
- ▶ extend the proof-reconstruction presented in this paper to
  - ▶ support the identity theory
  - ▶ support other ATPs for propositional logic like EProver or Z3.  
See Kanso's Ph.D. thesis [5]
  - ▶ support Metis first-order proofs

# Related Work

In type theory:

- ▶ Kanso in [5] reconstructs in Agda propositional proofs generated by EProver and Z3
- ▶ Foster and Struth in [2] describe proof-reconstruction in Agda for equational logic of Waldmeister prover
- ▶ Bezem, Hendriks, and Nivelle in [1] transform a proof produced by the first-order prover Bliksem in a Coq proof-term

In classical logic:

- ▶ Paulson and Susanto in [6] introduce SledgeHammer, a tool able to reconstruct proofs of well-known ATPs: EProver, Vampire, among others using SystemOnTPTP server
- ▶ Hurd in [3] integrates the first-order resolution prover Gandalf prover for HOL proof-assistant
- ▶ Kaliszyk and Urban in [4] reconstruct proofs of different ATPs for HOL Light

# References I



Marc Bezem, Dimitri Hendriks, and Hans de Nivelle. Automated Proof Construction in Type Theory Using Resolution. *Journal of Automated Reasoning* 29.3-4 (2002), pp. 253–275. DOI: 10.1023/A:1021939521172 (cit. on p. 31).



Simon Foster and Georg Struth. Integrating an Automated Theorem Prover in Agda. In: *NASA Formal Methods (NFM 2011)*. Ed. by Mihael Bobaru et al. Vol. 6617. Lecture Notes in Computer Science. Springer, 2011, pp. 116–130. DOI: 10.1007/978-3-642-20398-5\_10 (cit. on p. 31).



Joe Hurd. Integrating Gandalf and HOL. In: *Theorem Proving in Higher Order Logics (TPHOLs 2001)*. Ed. by Yves Bertot, Gilles Dowek, Laurent Théry, and Christine Paulin. Vol. 1690. Lecture Notes in Computer Science. Springer, 2001, pp. 311–321. DOI: 10.1007/3-540-48256-3\_21 (cit. on p. 31).



# References II



Cezary Kaliszyk and Josef Urban. PRoCH: Proof Reconstruction for HOL Light. In: Automated Deduction (CADE-24). Ed. by Maria Paola Bonacina. Vol. 7898. Lecture Notes in Artificial Intelligence. Springer, 2013, pp. 267–274. DOI: 10.1007/978-3-642-38574-2\_18 (cit. on p. 31).



Karim Kanso. Agda as a Platform for the Development of Verified Railway Interlocking Systems. PhD thesis. Department of Computer Science. Swansea University, 2012. URL: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.310.1502> (cit. on pp. 30, 31).



Lawrence C. Paulson and Kong Woei Susanto. Source-level Proof Reconstruction For Interactive Theorem Proving. In: TPHOLs. Vol. 4732. Springer. 2007, pp. 232–245 (cit. on p. 31).



Jonathan Prieto-Cubides. A Collection of Propositional Problems in TPTP Format. June 2017. DOI: 10.5281/ZENODO.817997 (cit. on p. 22).

# References III



[Jonathan Prieto-Cubides](#). A Library for Classical Propositional Logic in Agda. 2017. DOI: [10.5281/zenodo.398852](https://doi.org/10.5281/zenodo.398852) (cit. on p. 17).

# TPTP Syntax

Thousands of Problems for Theorem Provers

- ▶ Is a language<sup>9</sup> to encode problems
- ▶ Is the input of the ATPs
- ▶ Annotated formulas with the form  
language(name, role, formula).

language FOF or CNF

name to identify the formula within the problem

role axiom, definition, hypothesis, conjecture

formula formula in TPTP format

---

<sup>9</sup><http://www.cs.miami.edu/~tptp/TPTP/SyntaxBNF.html>.

# Metis Theorem Prover

Metis is an automatic theorem prover for first-order logic with equality.

- ▶ Open source implemented
- ▶ Reads problems in TPTP format
- ▶ Outputs *detailed* proofs in TSTP format
- ▶ For the propositional logic, Metis has only three inference rules:

$$\frac{}{\Gamma \vdash \varphi_1 \vee \dots \vee \varphi_n} \text{ axiom } \varphi_1, \dots, \varphi_n$$

$$\frac{}{\Gamma \vdash \varphi \vee \neg \varphi} \text{ assume } \varphi$$

$$\frac{\Gamma \vdash \varphi_1 \vee \dots \vee l \vee \dots \vee \varphi_n \quad \Gamma \vdash \psi_1 \vee \dots \vee \neg l \vee \dots \vee \psi_m}{\Gamma \vdash \varphi_1 \vee \dots \vee \varphi_n \vee \psi_1 \vee \dots \vee \psi_m} \text{ resolve } l$$

# TSTP Syntax

A TSTP derivation<sup>10</sup>

- ▶ Is a **D**irected **A**cyclic **G**raph where
  - `leaf` is a formula from the TPTP input
  - `node` is a formula inferred from parent formula
  - `root` the final derived formula
- ▶ Is a list of annotated formulas with the form

```
language(name, role, formula, source [,useful info]).
```

where `source` typically is an inference record

```
inference(rule, useful info, parents).
```

---

<sup>10</sup><http://www.cs.miami.edu/~tptp/TPTP/QuickGuide/Derivations.html>.

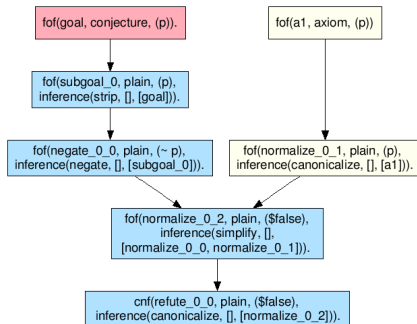
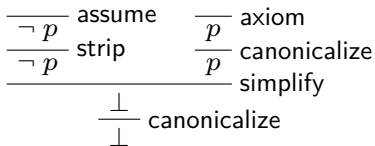
# Another TSTP Example

- Proof found by Metis for the problem  $p \vdash p$

```
$ metis --show proof problem.tptp
fof(a, axiom, p).
fof(goal, conjecture, p).
fof(subgoal_0, plain, p),
    inference(strip, [], [goal])).
fof(negate_0_0, plain, ~ p,
    inference(negate, [], [subgoal_0])).
fof(normalize_0_0, plain, ~ p,
    inference(canonicalize, [], [negate_0_0])).
fof(normalize_0_1, plain, p,
    inference(canonicalize, [], [a])).
fof(normalize_0_2, plain, $false,
    inference(simplify, [],
        [normalize_0_0, normalize_0_1])).
cnf(refute_0_0, plain, $false,
    inference(canonicalize, [], [normalize_0_2])).
```

# DAG Example

By refutation, we proved  $p \vdash p$ :



# Athena tool

Is an Haskell program that translates proofs given by Metis in TSTP format to Agda code

- ▶ Parsing of TSTP language
- ▶ Creation and analysis of **DAG** derivations
- ▶ Analysis of inference rules used in the TSTP derivation
- ▶ Agda code generation

| Library    | Purpose  |
|------------|--|
| agda-prop  | axioms and theorems of classical propositional logic |
| agda-metis | versions of the inference rules used by Metis        |