Modularity in Proof Checking

Catherine Dubois

ENSIIE, Samovar, CNRS Évry, France

Proof Checking

- Proof checking = check proofs for correctness (in a given logic formalism)
- Much simpler to verify a proof than finding a proof
  Proof checking may include proof reconstruction
- Batch or interactive process
- Proof checkers : part of interactive provers (e.g. Coq) or independent tools (e.g. DEDUKTI)

A Type Checking Problem

Proof checking $\overset{\text{type theory}}{\longrightarrow}$ Type checking

A proof π is a proof of ϕ iff π is a lambda-term of type ϕ.

Dedukti

Dedukti (`http://dedukti.gforge.inria.fr/`) : a universal proof checker / logical framework developed by Dowek and his group based on $\lambda\Pi$-calculus modulo theory = dependent types *à la* LF + (user-defined) rewriting rules

Dedukti can check proofs from iProverModulo (resolution proofs) :, Zenon modulo (f.o. tableaux proofs), HOL provers (open theory format), Matita and Coq (CIC proofs), FoCaLiZe, thanks to translators.

Modularity

- Computer Science

  Modular program = a set of components/modules with well-defined interfaces and dependencies
  Modularity ⇒ mechanisms to compose/compile/assemble components together to obtain an executable system

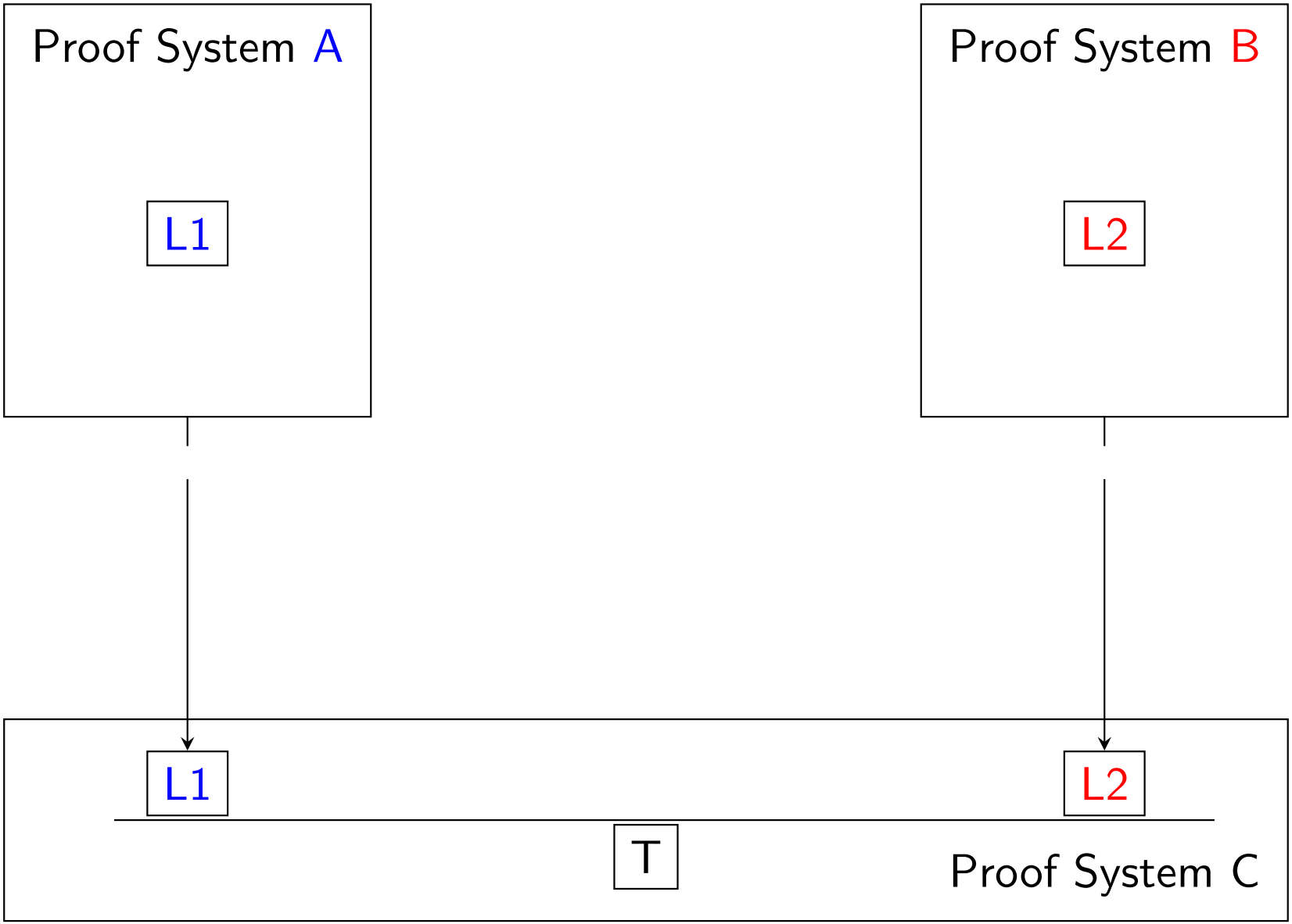
- Proof Checking

  Internal modularity : provides mechanisms to structure large proofs (e.g. modules, type classes, functors, inheritance) Georges's talk

  External modularity : provides mechanisms to interface proofs having different origins and build composite proofs.
  module $\mapsto$ a checked proof - interface $\mapsto \phi$ - body $\mapsto \pi$
  executable system $\mapsto$ composite checked proofs

  External modularity brings some interoperability between proof tools .

External Modularity for Proof Checking



Interoperability

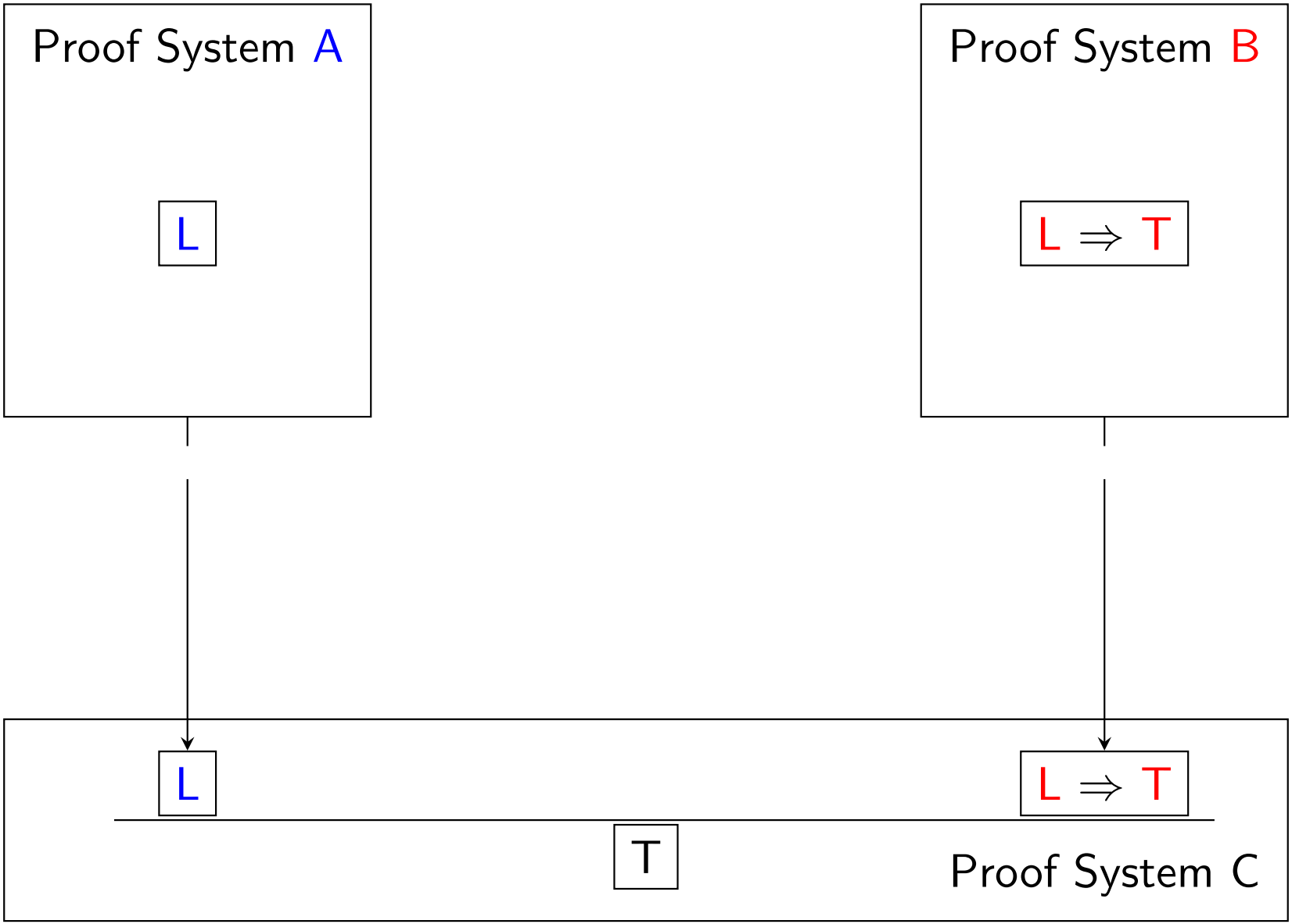
Motivations

- Proof development is *expensive*
  - 4-color theorem, Kepler conjecture, Feit-Thomson theorem
- Proof assistants are *specializing*
  - Counterexamples, proof by reflection, decision procedures, ...

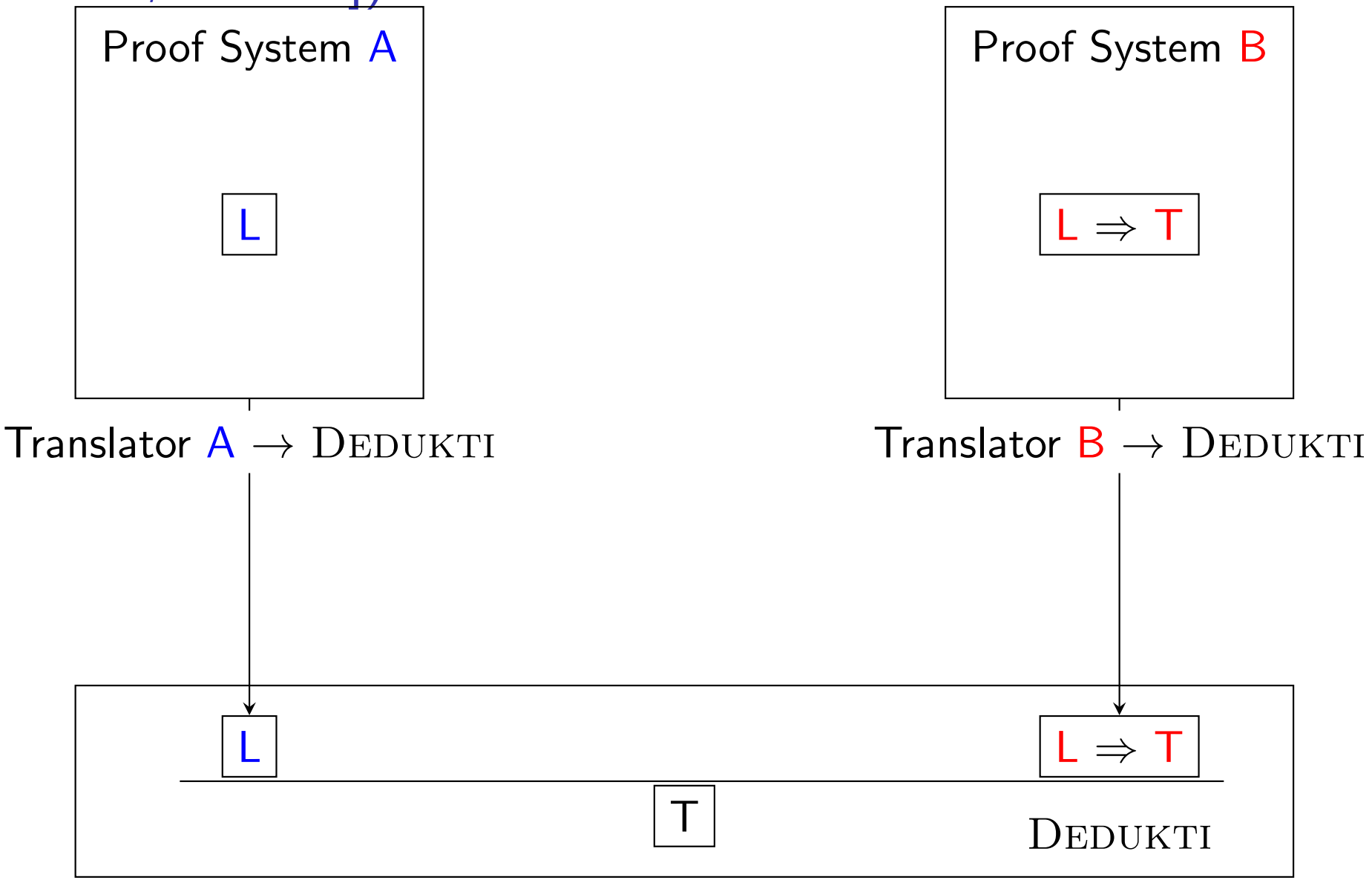
Barriers

- Logical problem :
  We need to combine the logics of A and B in a consistent way.
- Mathematical problem : L and L are not identical
  Theories such as arithmetic are independently defined in System A and System B.
  We need to identify similar concepts.

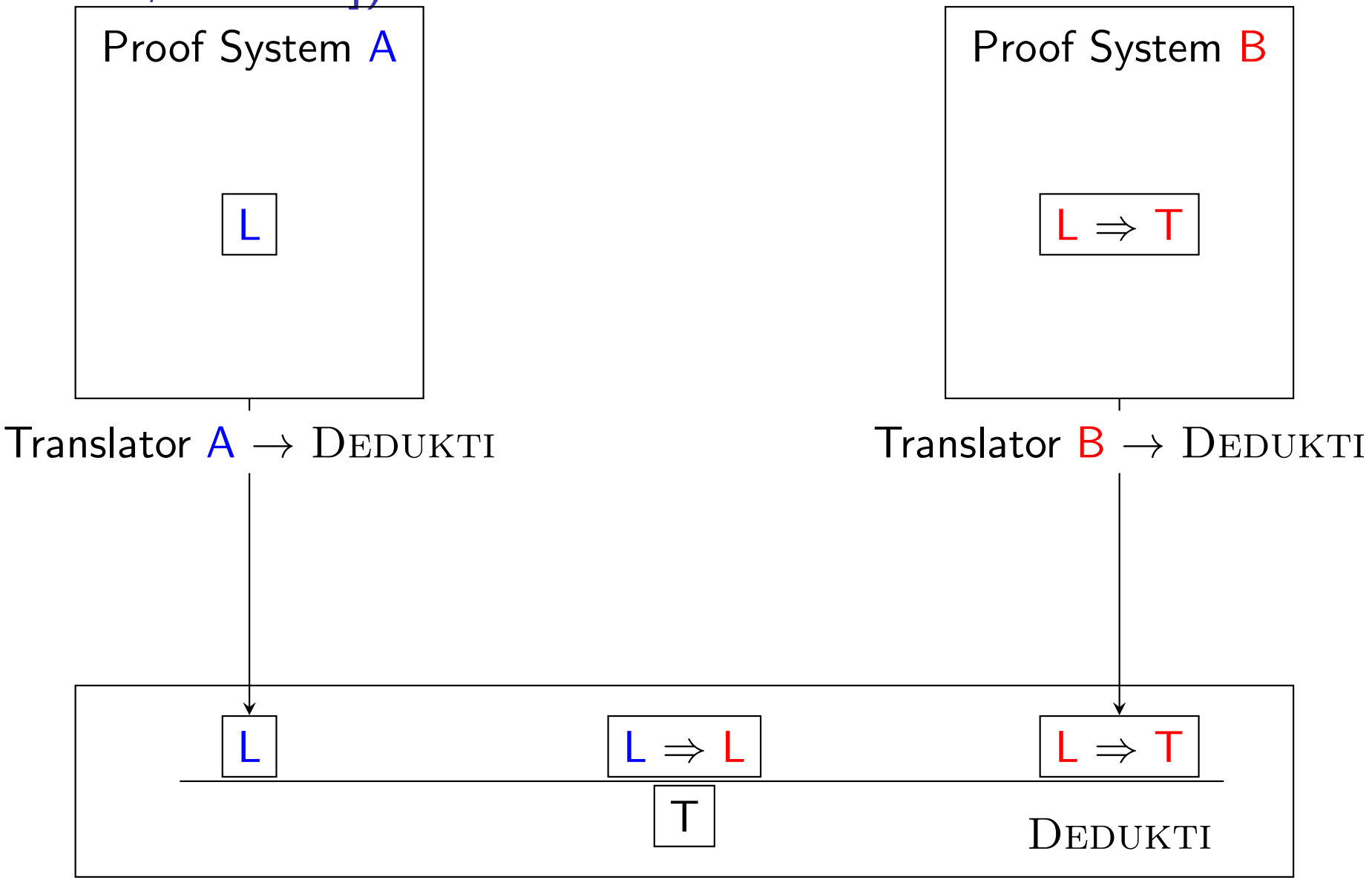
External Modularity for Proof Checking : Refinement

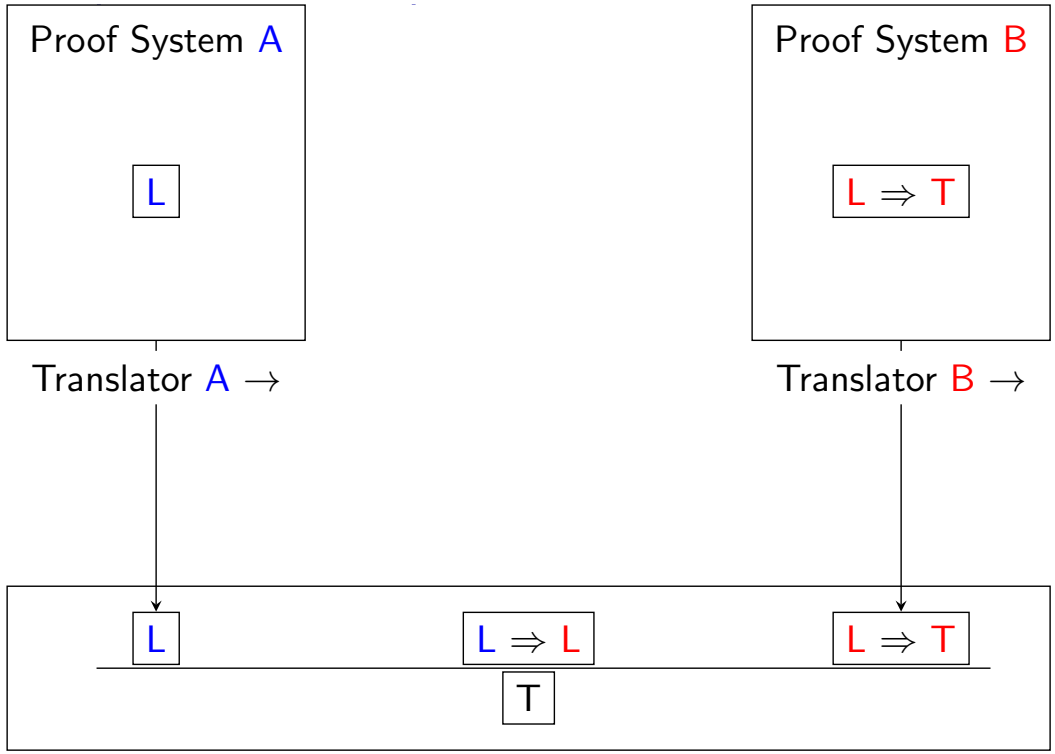


Composite checked proofs with Dedukti ([Cauderlier, Dubois, ITP 17])



Composite checked proofs with Dedukti ([Cauderlier, Dubois, ITP 17])

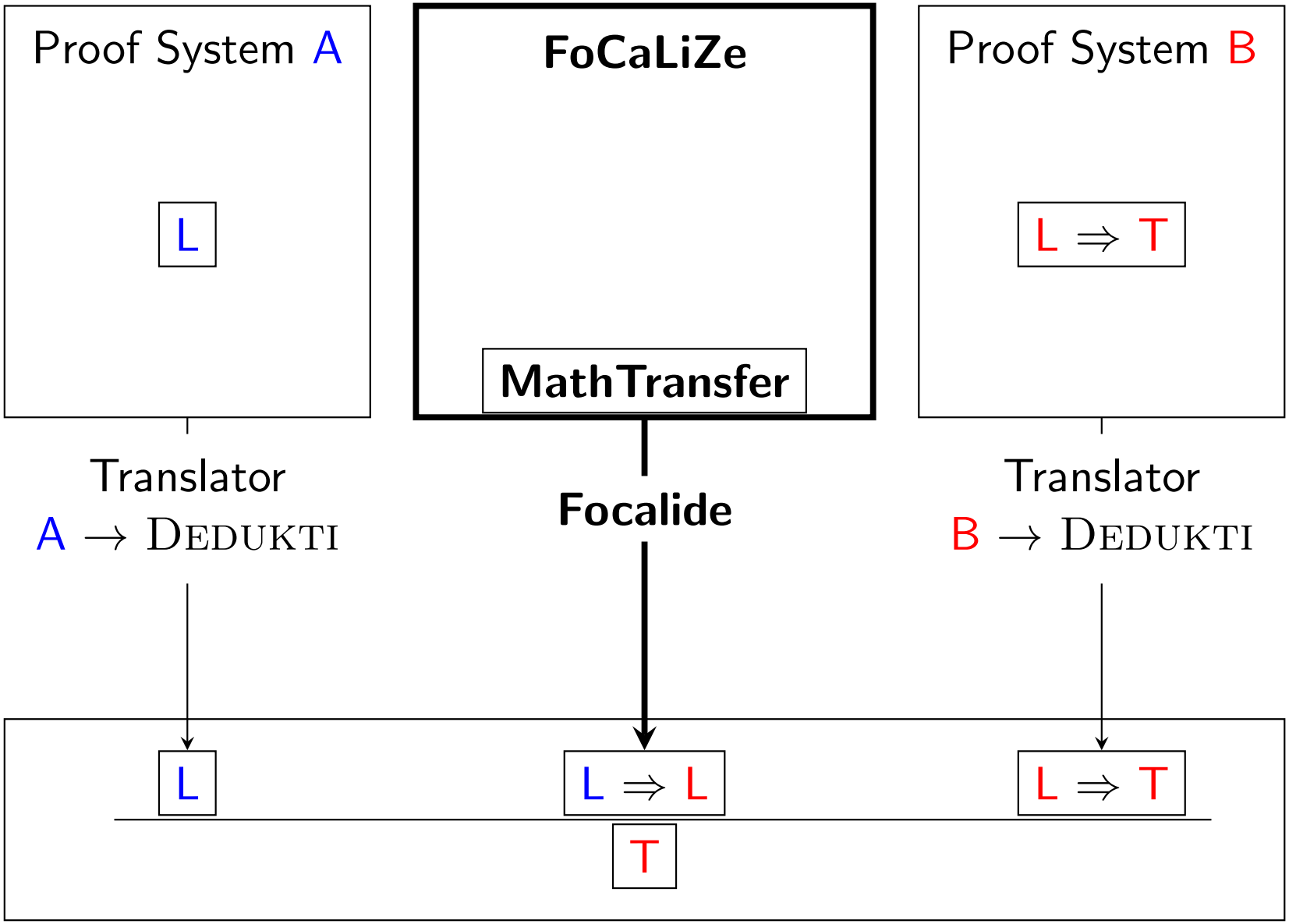




$\longrightarrow$ Theorem Transfer (automate reasoning modulo isomorphism)

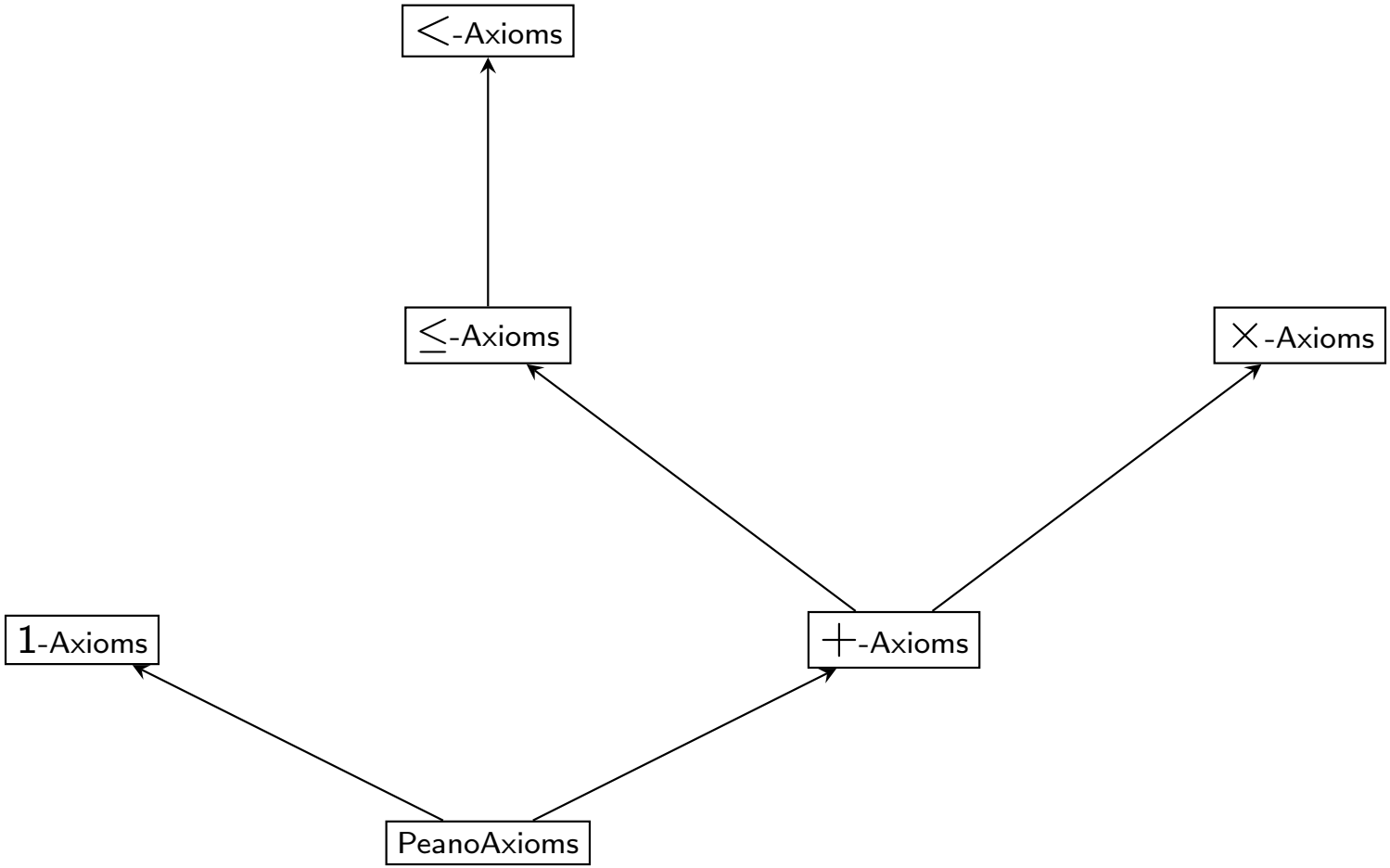
Definition : let A, B, two isomorphic structures, ϕ_A a formula on A, ϕ_B the corresponding formula on B, $\phi_A \Rightarrow \phi_B$: a transfer theorem,

`transfer` tactic in Isabelle (Huffman, Kuncar), `transfer` tactic in Coq (Zimmermann, Herbelin), `transfer` tactic in Dedukti (Cauderlier)

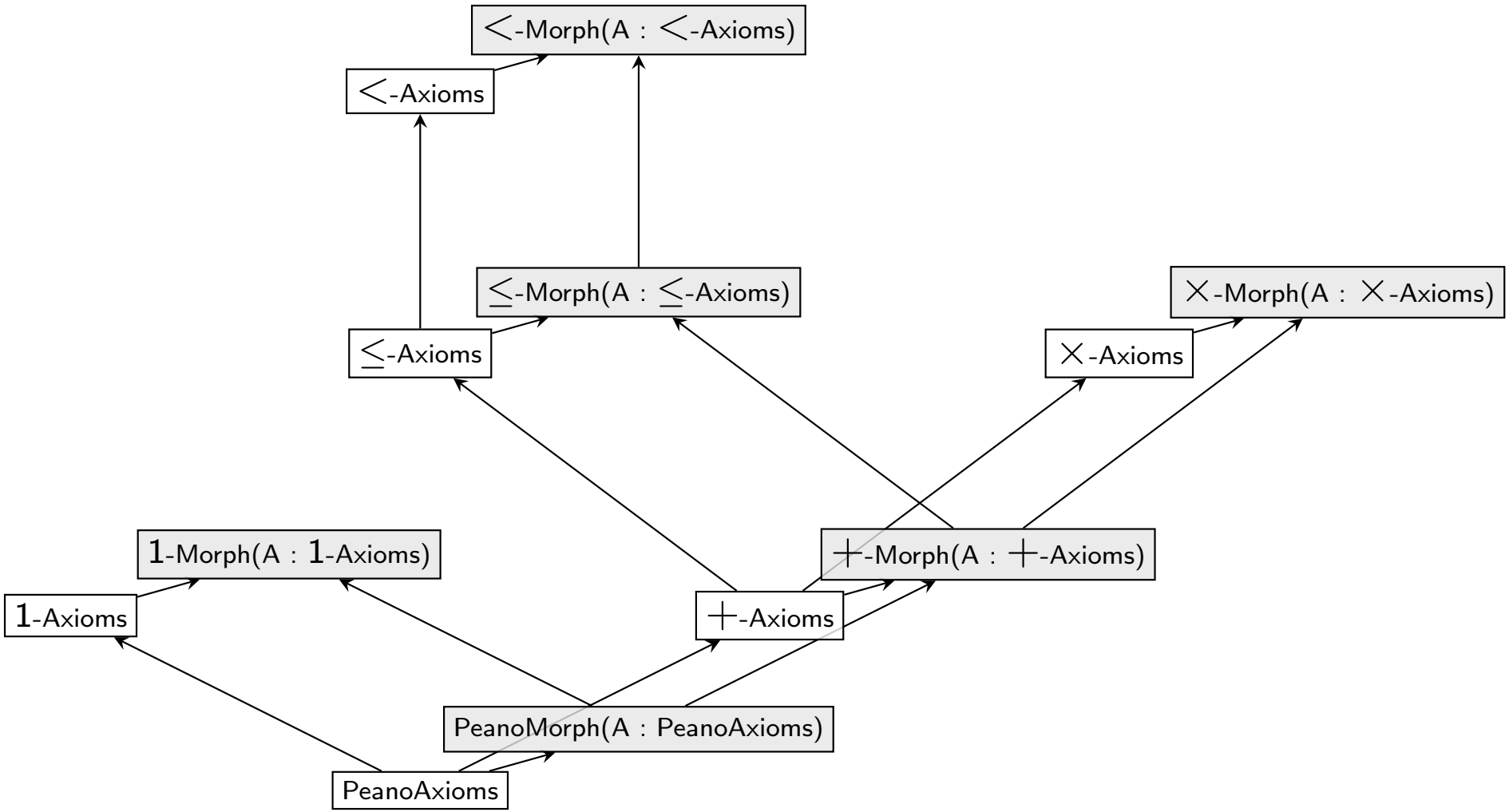


MathTransfer = a FoCaLiZe library of transfer theorems about natural number arithmetic (`https://gitlab.math.univ-paris-diderot.fr/cauderlier/math_transfer`)

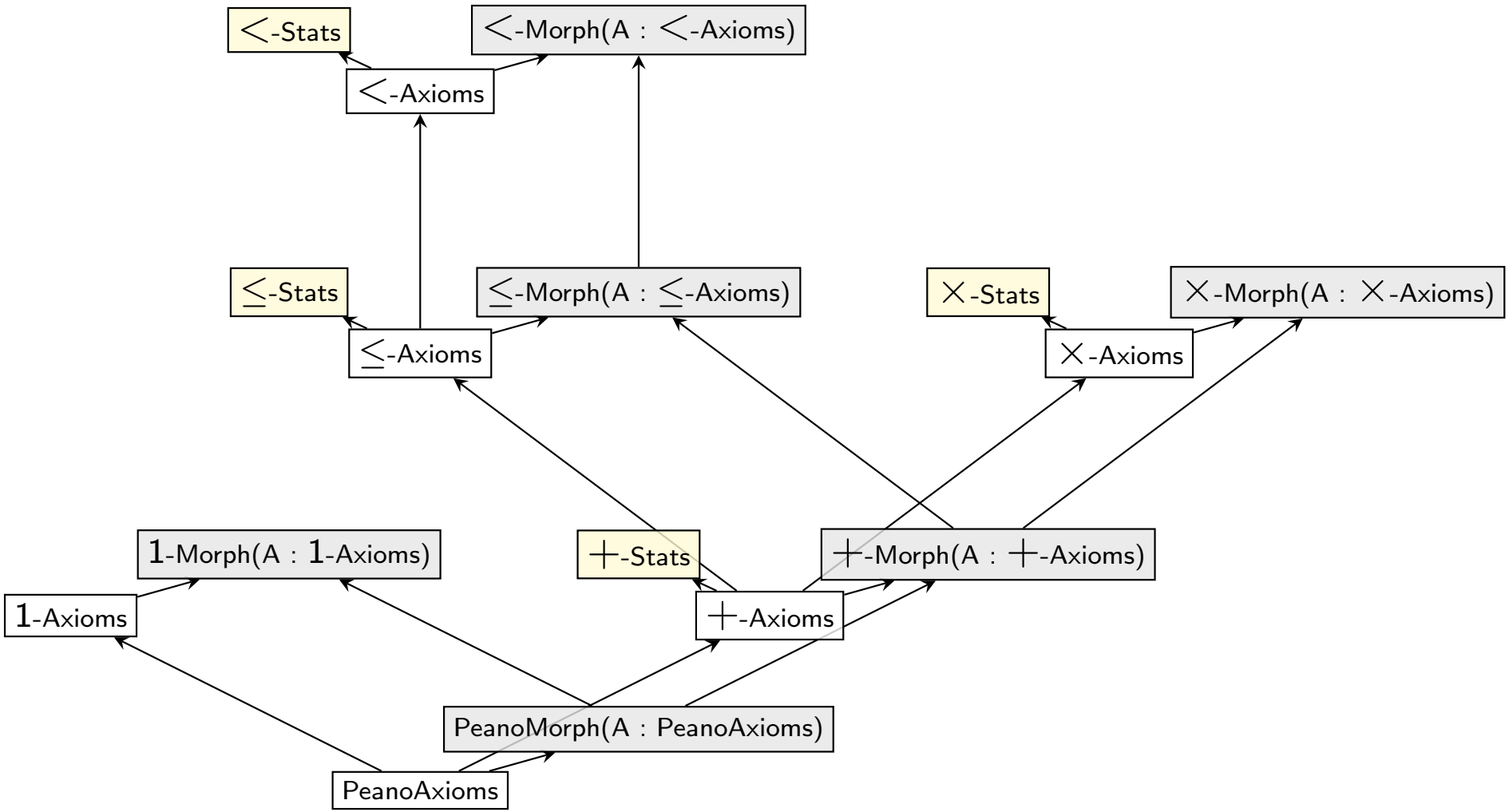
MathTransfer



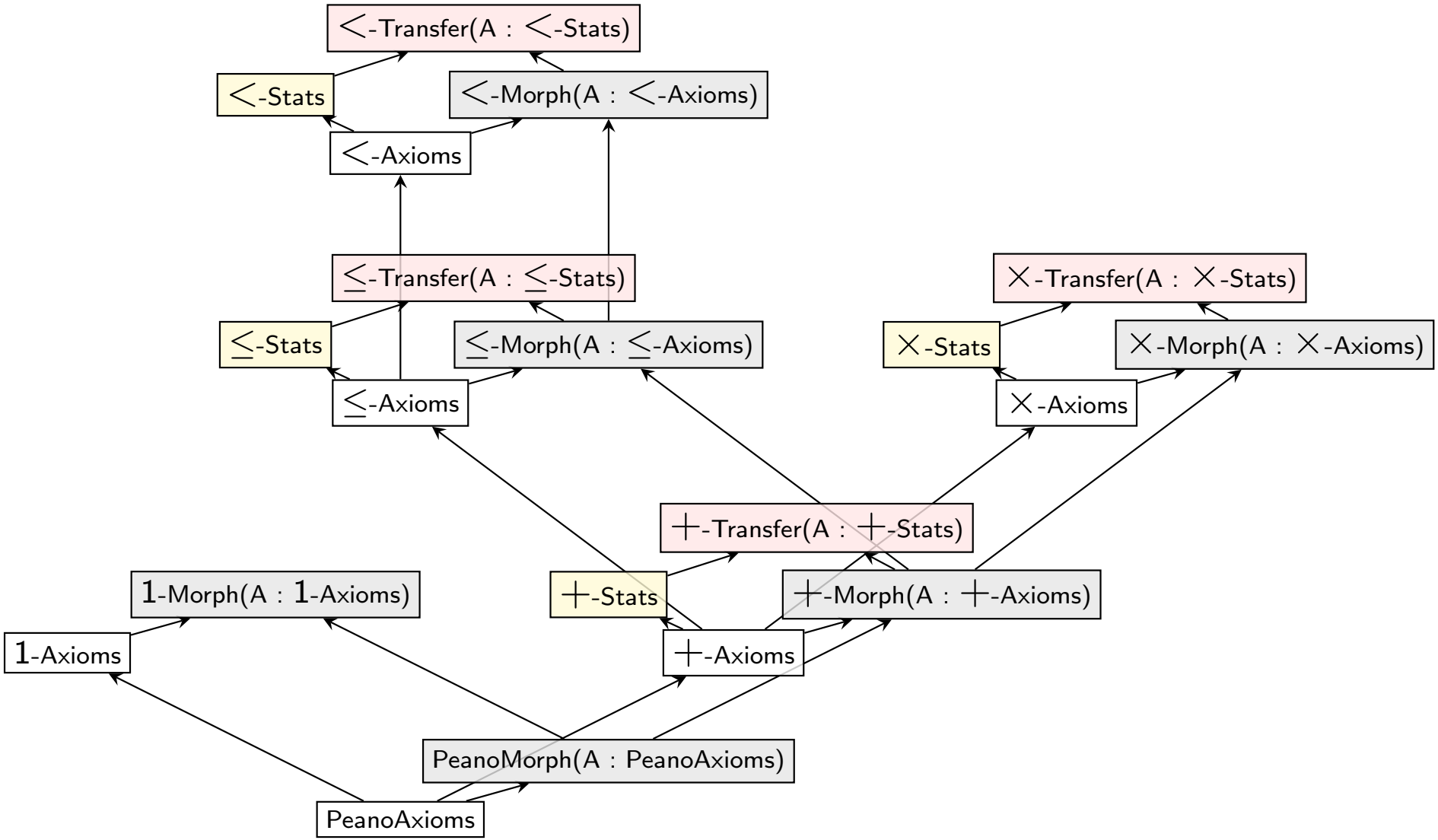
MathTransfer



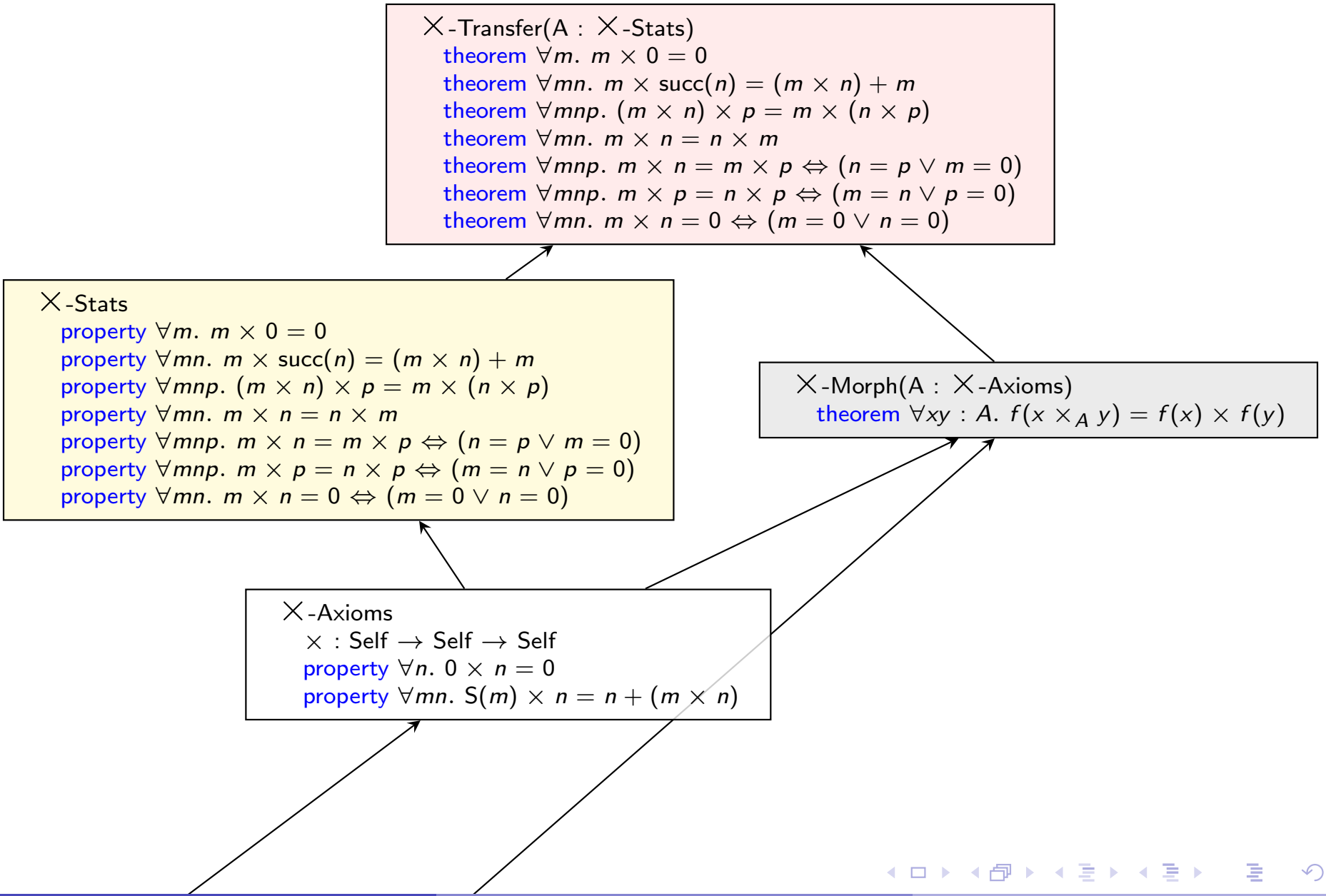
MathTransfer



MathTransfer



MathTransfer

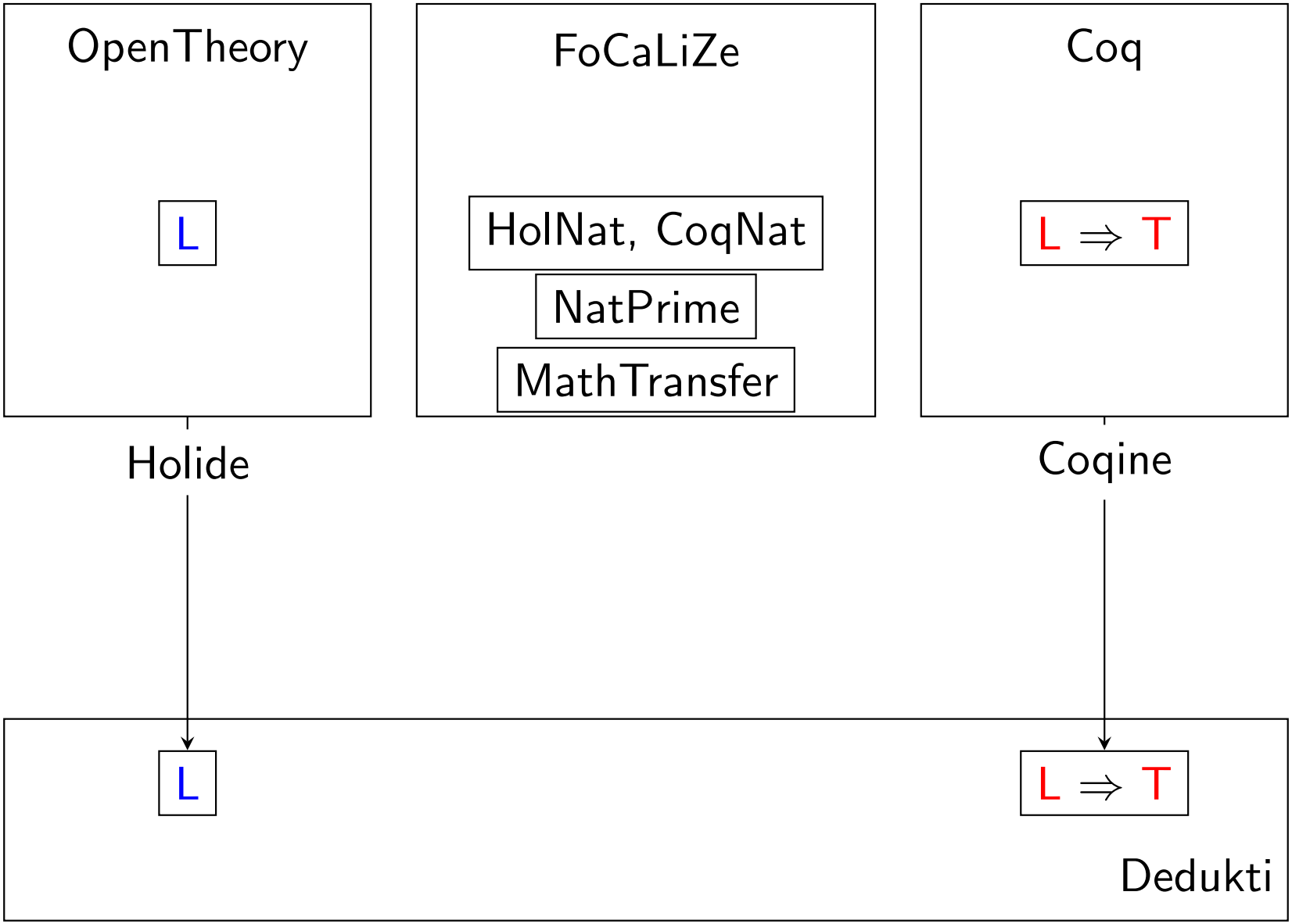


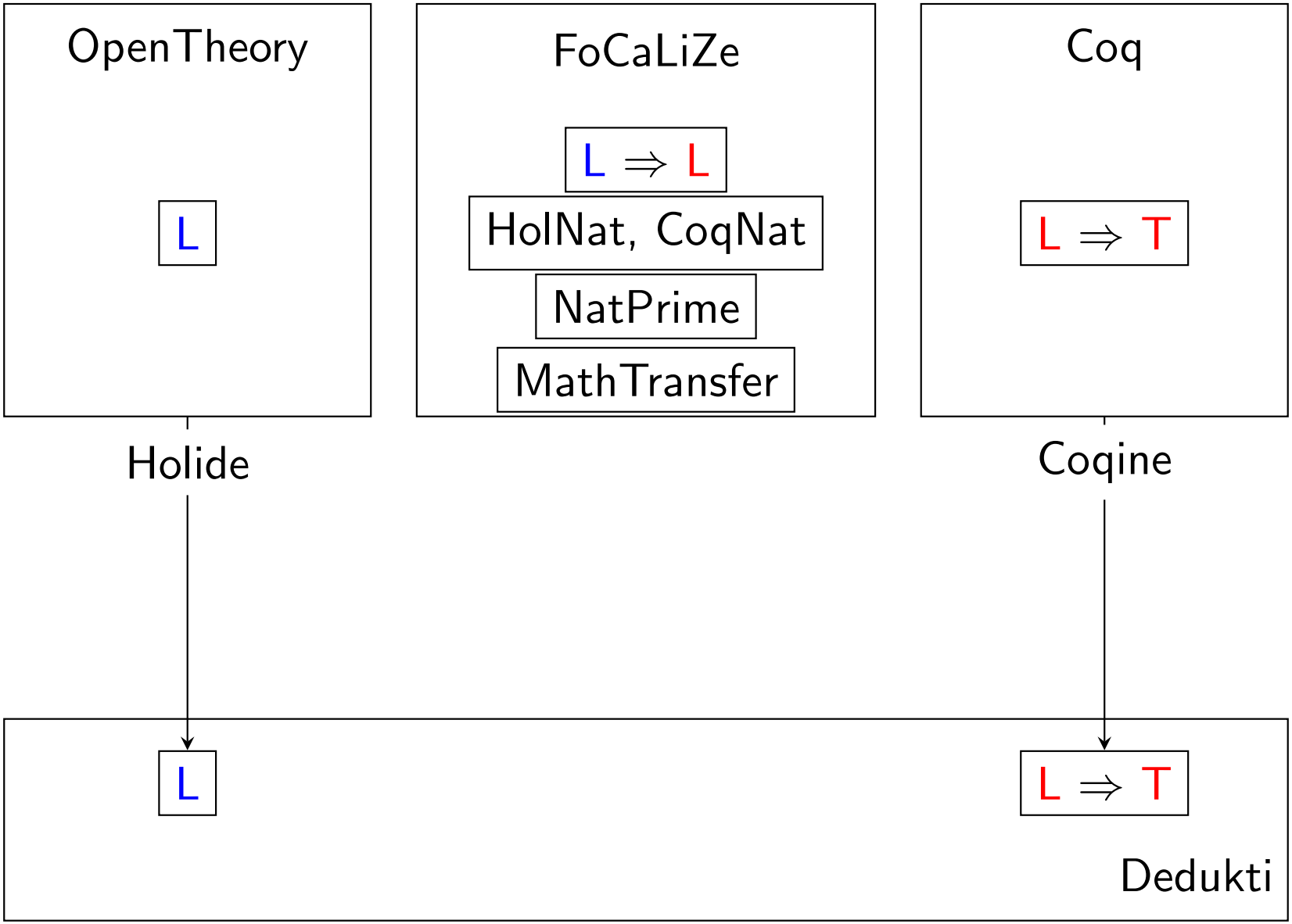
Case Study : a composite checked proof of correctness of Eratosthenes Sieve

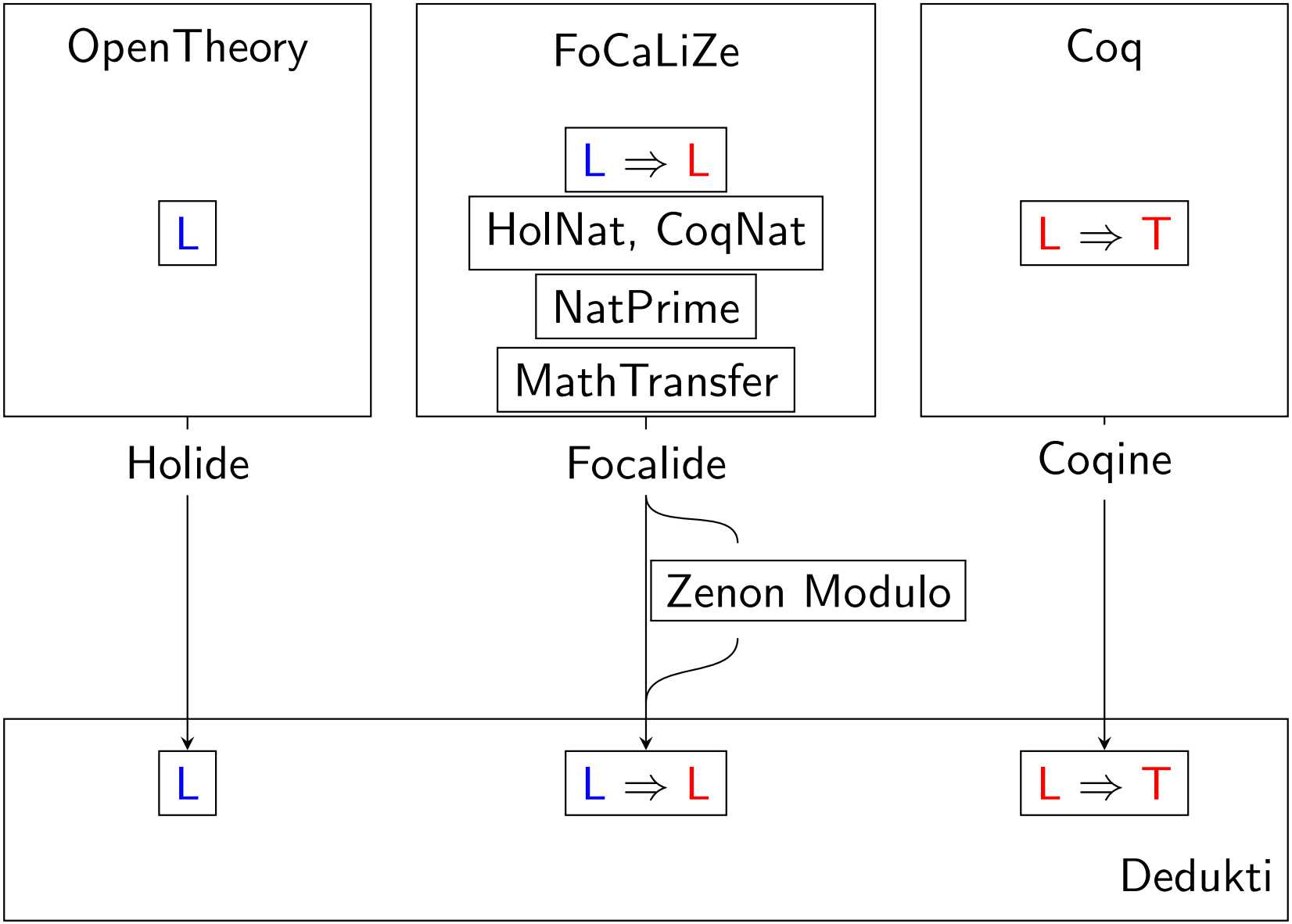
- A = HOL (OpenTheory)
- B = Coq
- T = correctness of the Sieve of Eratosthenes
- L = prime divisor lemma

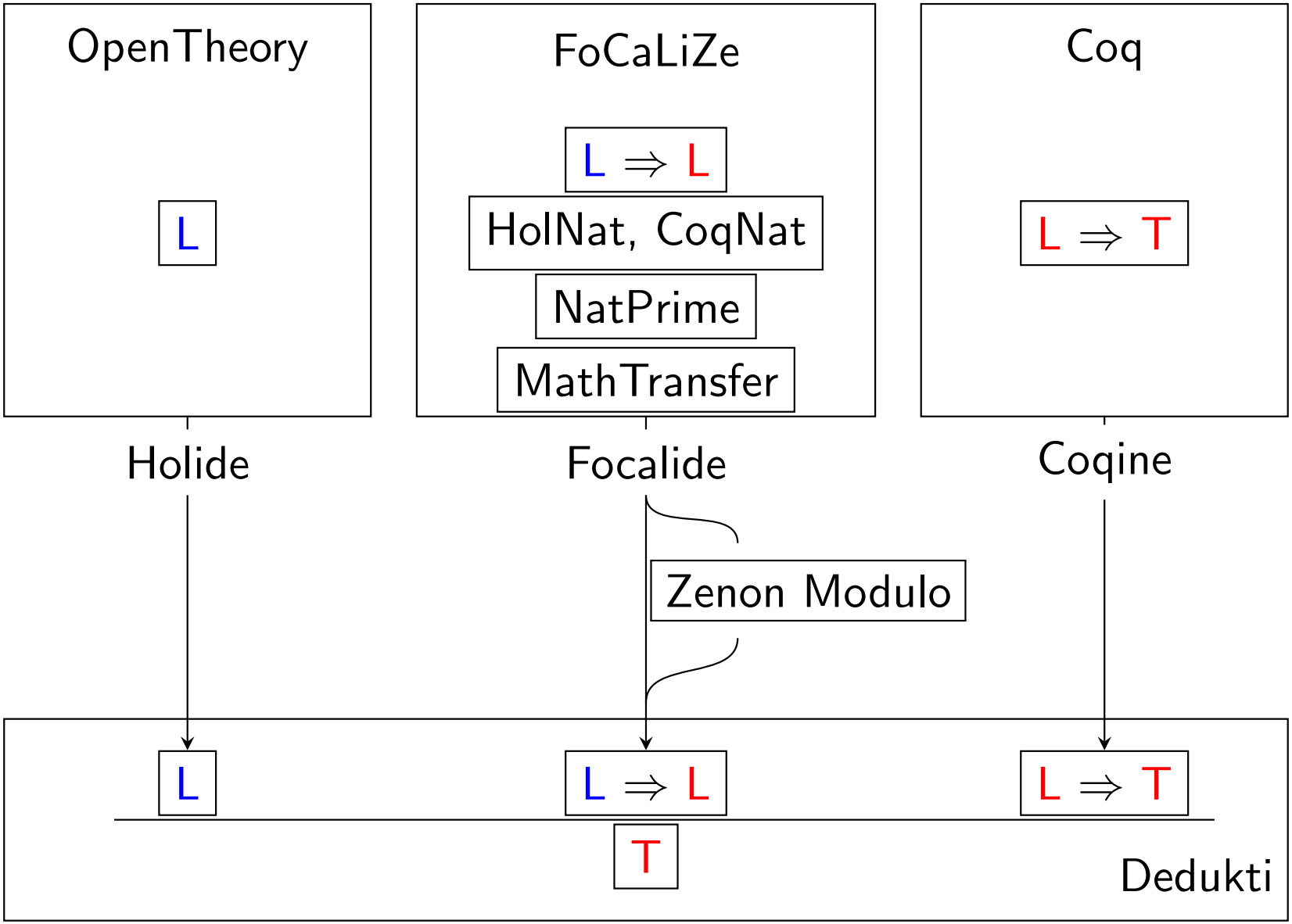
$$\mathsf{L} := \forall n \neq 1.\ \exists p.\ \mathsf{prime}(p) \wedge p \mid n$$

proved in HOL/OpenTheory `natural-prime` library


OpenTheory
L
FoCaLiZe
HolNat, CoqNat
NatPrime
MathTransfer
Coq
L ⇒ T
Holide
Coqine
L
L ⇒ T
Dedukti


OpenTheory
L
FoCaLiZe
L ⇒ L
HolNat, CoqNat
NatPrime
MathTransfer
Coq
L ⇒ T
Holide
Coqine
L
L ⇒ T
Dedukti


OpenTheory
L
FoCaLiZe
L ⇒ L
HolNat, CoqNat
NatPrime
MathTransfer
Coq
L ⇒ T
Holide
Focalide
Coqine
Zenon Modulo
L
L ⇒ L
L ⇒ T
Dedukti


OpenTheory
L
FoCaLiZe
L ⇒ L
HolNat, CoqNat
NatPrime
MathTransfer
Coq
L ⇒ T
Holide
Focalide
Coqine
Zenon Modulo
L
L ⇒ L
T
L ⇒ T
Dedukti

What is your view of
modular proof checking ?

MathTransfer in numbers

- 11 operations (0, S, 1, bit0, bit1, pred, $+$, $\times$, $\leq$, -, ¡)
- 84 transfer theorems
- 69 species
- 1771 lines, 74KB
- 1.7MB generated Dedukti code (71% from Zenon Modulo and the transfer tactic)