Bimodal TM Logic

A Complete Formalization in Lean 4

Benjamin Brast-McKie

ProofChecker Project

January 13, 2026

Outline

What is TM Logic?

TM = **T**ense + **M**odality

A bimodal propositional logic combining:

S5 Modal Logic

- $\Box\varphi$ — necessarily φ
- $\Diamond\varphi$ — possibly φ
- Metaphysical necessity

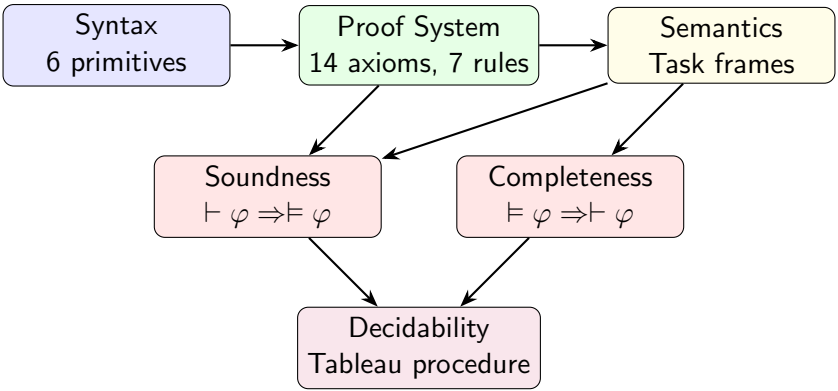
Linear Temporal Logic

- $G\varphi$ — always future
- $H\varphi$ — always past
- $\triangle\varphi$ — eternally (all times)

Key Innovation

Interaction axioms (MF, TF) connect modal and temporal reasoning.

Formalization Overview



Syntax: Six Primitives

Symbol	Lean	Reading
p	`atom "p"`	propositional variable
$\bot$	`bot`	falsity
$\varphi \to \psi$	`imp` φ ψ	implication
$\Box\varphi$	`box` φ	necessity
$H\varphi$	`all_past` φ	always past
$G\varphi$	`all_future` φ	always future

Derived Operators

$\neg\varphi := \varphi \to \bot \qquad \Diamond\varphi := \neg\Box\neg\varphi \qquad \triangle\varphi := H\varphi \wedge \varphi \wedge G\varphi$

The Perpetuity Principles (P1–P6)

Deep connections between necessity and time:

P1: $\Box\varphi \to \triangle\varphi$
Necessary implies eternal

P2: $\triangledown\varphi \to \Diamond\varphi$
Sometime implies possible

P3: $\Box\varphi \to \Box\triangle\varphi$
Necessity of perpetuity

P4: $\Diamond\triangledown\varphi \to \Diamond\varphi$
Possibility of occurrence

P5: $\Diamond\triangledown\varphi \to \triangle\Diamond\varphi$
Persistent possibility

P6: $\triangledown\Box\varphi \to \Box\triangle\varphi$
Occurrent necessity is perpetual

All Six Proven in Lean

✓ Complete machine-checked proofs in `Perpetuity.lean`

Perpetuity Principle P1 in Lean

P1: Necessary implies eternal

$\Box\varphi \rightarrow \triangle\varphi$

```
#check @perpetuity_1
-- perpetuity_1 : ( : Formula)   .box.imp ()

example ( : Formula) :  .box.imp () :=
  perpetuity_1
```

Interpretation

If something is *metaphysically necessary*, it holds at *all times*. Conservation of energy is necessary, so it has always been and will always be true.

Metalogical Results

Result	Statement	Status
Soundness	$(\Gamma \vdash \varphi) \Rightarrow (\Gamma \vDash \varphi)$	✓Proven
Deduction	$(A :: \Gamma \vdash B) \Rightarrow (\Gamma \vdash A \to B)$	✓Proven
Completeness	$\vDash \varphi \Rightarrow \vdash \varphi$	✓Proven
Equivalence	$\vdash \varphi \Leftrightarrow \vDash \varphi$	✓Proven
Decidability	`decide` φ `:` `DecisionResult`	✓Implemented

Main Theorem

`main_provable_iff_valid : Nonempty (⊢ ) ↔ valid`

Derivability and validity coincide for TM logic.

Building Proofs: Modal T Axiom

Goal

Prove the Modal T axiom: $\Box p \to p$

```
/-- The modal T axiom is directly available -/
example (p : String) :
    (Formula.atom p).box.imp (Formula.atom p) := by
  exact DerivationTree.axiom [] _
      (Axiom.modal_t (Formula.atom p))
```

Key Insight

Axioms are constructors of `DerivationTree`. Proofs are data structures, not propositions.

Modus Ponens in Action

Rule

From $\vdash A$ and $\vdash A \to B$, derive $\vdash B$

```
example (p : Formula)
    (h1 :  p.box)
    (h2 :  p.box.imp p.diamond) :
     p.diamond := by
  exact DerivationTree.modus_ponens []
       p.box p.diamond h2 h1
```

Necessitation Rule

From $\vdash \varphi$, derive $\vdash \Box\varphi$

```
example ( : Formula) (h :  ) :  .box := by
  exact DerivationTree.necessitation  h
```

The Deduction Theorem

Theorem

If $A :: \Gamma \vdash B$, then $\Gamma \vdash A \to B$

```
-- Derivation from assumption
example (A : Formula) : [A]  A := by
  exact DerivationTree.assumption [A] A
        (List.mem_singleton_self A)

-- Deduction theorem converts to implication
noncomputable example (A B : Formula)
    (h : [A]  B) :  A.imp B := by
  exact deduction_theorem [] A B h
```

Uses well-founded recursion on derivation height.

Tableau-Based Decision

The `decide` function returns:

- **valid proof** — formula is valid, with proof term
- **invalid counter** — formula is invalid, with countermodel
- **timeout** — resources exhausted

API

- `decide : Formula → DecisionResult`
- `isValid : Formula → Bool`
- `isSatisfiable : Formula → Bool`
- `isTautology : Formula → Bool`

Valid Formulas

Modal T: $\Box p \to p$

```
def formula_modal_t :=
  (atom "p").box.imp
  (atom "p")

#eval decide formula_modal_t
-- valid (proof)
```

Modal K: $\Box(p \to q) \to (\Box p \to \Box q)$

```
def formula_modal_k :=
  let p := atom "p"
  let q := atom "q"
  (p.imp q).box.imp
  (p.box.imp q.box)

#eval decide formula_modal_k
-- valid (proof)
```

Proof Extraction

When valid, the decision procedure extracts a `DerivationTree` proof term.

Invalid Formulas

Converse of T: $p \to \Box p$

```
def formula_converse_t :=
  (atom "p").imp
  (atom "p").box

#eval decide formula_converse_t
-- invalid (countermodel)
```

True things need not be necessary

Possibility to actuality: $\Diamond p \to p$

```
def formula_poss_to_act :=
  (atom "p").diamond.imp
  (atom "p")

#eval decide formula_poss_to_act
-- invalid (countermodel)
```

Possible things need not be actual

Countermodel Extraction

When invalid, the procedure extracts a finite countermodel witnessing falsity.

Laws of Nature

Conservation of Energy

Physical laws are necessary truths that hold at all times.

```
def conservation_of_energy := atom "conservation_of_energy"

-- If conservation is necessary, it holds eternally
example :  conservation_of_energy.box.imp
           (conservation_of_energy) :=
  perpetuity_1 conservation_of_energy
```

Interpretation

P1 captures the intuition that *necessary truths are eternal*.

Astronomical Events

Lunar Eclipse

Events that happen sometimes illustrate temporal possibility.

```
def lunar_eclipse := atom "lunar_eclipse"

-- If an eclipse sometimes occurs, it is possible
example :  (lunar_eclipse).imp lunar_eclipse.diamond :=
  perpetuity_2 lunar_eclipse
```

Interpretation

P2 captures: *temporal existence entails modal possibility*. If something happens at some time, it must be possible.

Mathematical Truths

$2 + 2 = 4$

Mathematical truths are paradigmatic necessary truths.

```
def two_plus_two_equals_four := atom "2+2=4"

-- Mathematical truths are necessarily eternal
example :  two_plus_two_equals_four.box.imp
           (two_plus_two_equals_four).box :=
  perpetuity_3 two_plus_two_equals_four
```

Interpretation

P3 captures: *necessity transfers to perpetuity*. What is necessary is necessarily eternal.

The Deepest Principle: P6

Occurrent Necessity is Perpetual

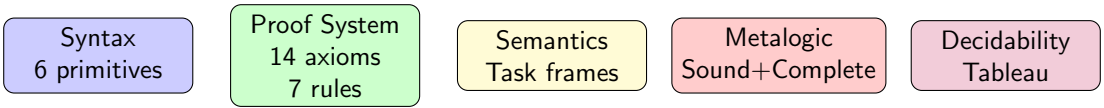
$\bigtriangledown\Box\varphi \rightarrow \Box\bigtriangleup\varphi$

```
-- If necessity ever occurs, it's necessarily eternal
noncomputable example (truth : Formula) :
     ((truth.box)).imp (truth).box :=
  perpetuity_6 truth
```

Philosophical Significance

If it's ever the case that something is necessary, then it's *necessarily eternal*.
Necessity, once realized, cannot be undone.

What We Built



Proven Results

- ✓ Soundness
- ✓ Deduction Theorem
- ✓ Weak Completeness
- ✓ Equivalence Theorem
- ✓ P1–P6 Perpetuity

Implemented

- ✓ Decision Procedure
- ✓ Proof Extraction
- ✓ Countermodel Generation
- ✓ Batch Validation

The Main Theorem

`main_provable_iff_valid`

$$\text{Nonempty}(\vdash \varphi) \Leftrightarrow \text{valid } \varphi$$

Significance

Syntax meets semantics: What can be derived is exactly what is true in all models. This is the crowning achievement of the formalization.

Machine-Verified

Every theorem, lemma, and definition is verified by Lean 4's type checker. No gaps, no hand-waving, no `sorry`.

Resources

Code

- `Demo.lean` — Interactive tour
- `Perpetuity.lean` — P1–P6
- `Soundness.lean` — Soundness proof
- `Completeness.lean` — Completeness proof
- `DecisionProcedure.lean` — Tableau

Documentation

- `BimodalReference.pdf`
- `README.md`
- `QUICKSTART.md`
- `PROOF_PATTERNS.md`

Questions?