Reasoning with Many-Valued, Spatial and Temporal Logics with SOLE

Alberto Paparella

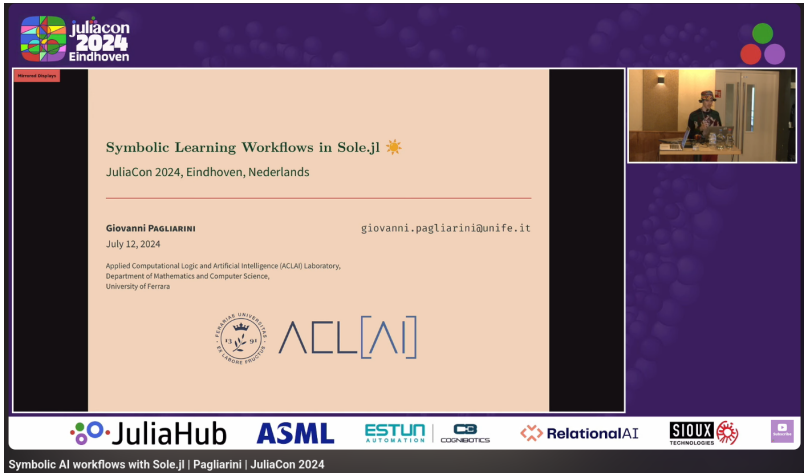
University of Ferrara, Department of Mathematics and Computer Science

Applied Computational Logic and Artificial Intelligence Laboratory

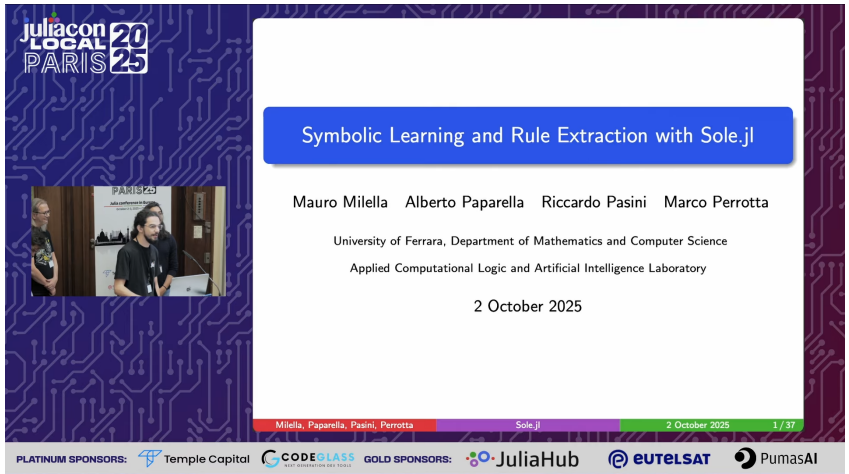
14 August 2026

SOLE at JuliaCon

(a) JuliaCon 2023



(b) JuliaCon 2024



(c) JuliaCon Local 2025

Today's talk

Other conferences:

- Motivation
- Applications
- Mathematical framework
- ...
- **Implementation in Julia (:**

Today's talk:

- New stuff in SOLE

Questions on left column?

Reach out anytime! (:

pprlrt@unife.it

Today's talk

SoleLogics.jl

https://github.com/aclai-lab/SoleLogics.jl

SoleReasoners.jl

https://github.com/aclai-lab/SoleReasoners.jl

using SoleLogics.jl: ManyValuedLogics

A broad family of Many-Valued Logics

A many-valued logic can be interpreted through its corresponding algebra.

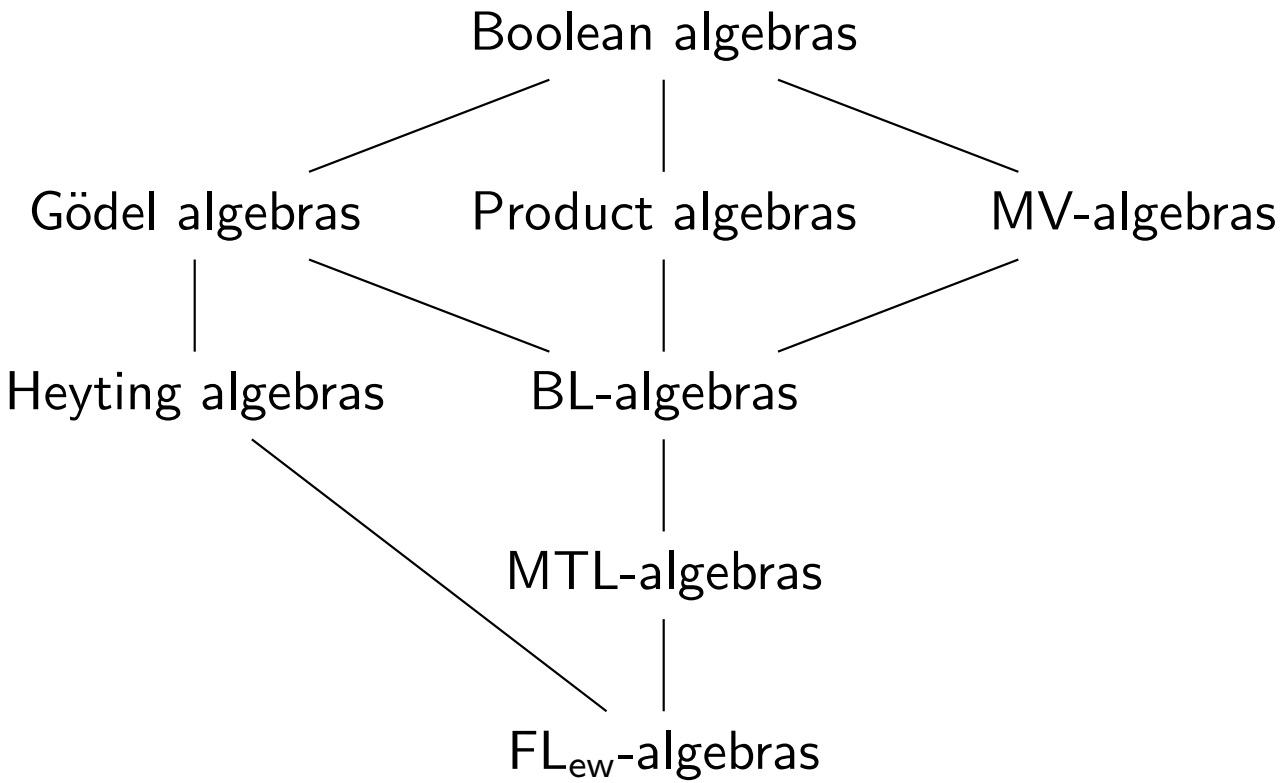


Figure: A partial taxonomy of well-known many-valued algebras.

The sweet spot: $\mathrm{FL_{ew}}$-algebras

Definition

$\mathrm{FL_{ew}}$-algebras

$$\mathcal{A} = \langle A, \cap, \cup, \cdot, 0, 1 \rangle$$

are defined over **bounded commutative residuated lattices**, where:

- $\langle A, \cap, \cup, 0, 1 \rangle$ represents a **bounded complete lattice**
- $\langle A, \cdot, 1 \rangle$ is a **commutative monoid**
- We can define an **implication** $\hookrightarrow$ (the residuation property holds)

$$\alpha \hookrightarrow \beta = \sup\{\gamma \mid \alpha \cdot \gamma \preceq \beta\}$$

$\mathcal{A}$ is a **chain** if $\langle A, \preceq \rangle$ is a total order, **finite** if A is finite.

The sweet spot: $\mathrm{FL}_{\mathrm{ew}}$-algebras

Take-home message:

- Lattice structure $=$ **(partial) order**
- Monoid operator $=$ **conjunction** $(\wedge)$
- Implication is always **derived** from conjunction

Some examples of lattice structures

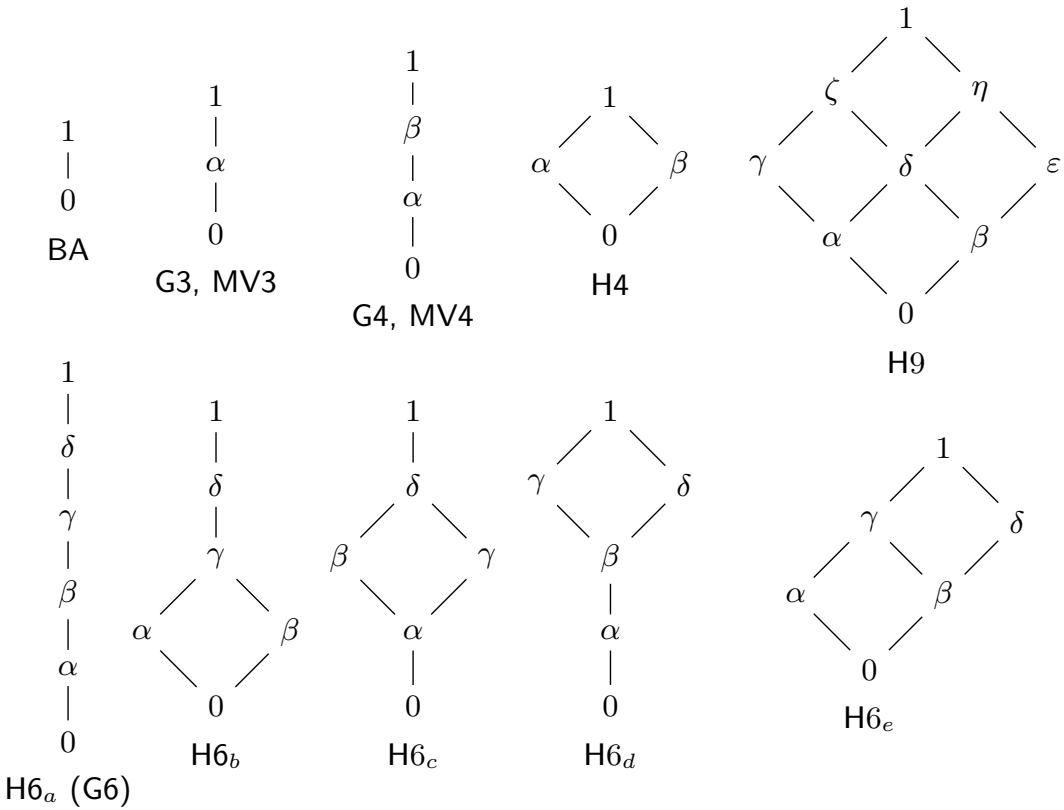


Figure: Examples of (finite) lattice structures (i.e., partial orders).

Some examples of monoid operators

We can use the monoid operator to model continuous t-norms:

- Łukasiewicz t-norm:

$$x * y = max(0, x + y - 1)$$

- Gödel t-norm:

$$x * y = min(x, y)$$

- Product t-norm:

$$x * y = x \cdot y \quad \text{(product of reals)}$$

Some examples of (derived) implications

For $x \leq y, x \Rightarrow y = 1$.

For $x > y$,

- Łukasiewicz implication:

$$x \Rightarrow y = \ y - x$$

- Gödel implication:

$$x \Rightarrow y = y$$

- Goguen implication:

$$x \Rightarrow y = y/x \quad \text{(residuum of product conjunction)}$$

using SoleReasoners.jl

Many-Valued Multi-Modal Logic

Definition

Let $\mathcal{A}P$ be a set of propositional letters, $\mathcal{A}$ a complete $\mathrm{FL}_{\mathrm{ew}}$-algebra. Well-formed formulas of *Multi-Modal Logic* $\mathrm{FL}_{\mathrm{ew}}$-$K_n$ are obtained as:

$$\varphi ::= \alpha \mid p \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid \varphi \to \psi \mid \langle X_i \rangle \varphi \mid [X_i]\varphi,$$

for $1 \leq i \leq n$, $p \in \mathcal{A}P$, and $\alpha \in \mathcal{A}$.

- We need a function $\widetilde{V}$ to evaluate each operator (**semantics**)
- $\langle X_i \rangle$ reads "it can be the case", $[X_i]$ reads "it must be the case"
- Useful for treating temporal and spatial logics!

The α-satisfiability problem

Definition

A formula φ of a many-valued logic is α-*satisfied* by some model $\widetilde{M}$ iff:

$$\widetilde{V}(\varphi) \succeq \alpha.$$

Definition

A formula φ is α-*satisfiable* iff there exists a model in which it is α-satisfied; resp., a formula is α-*valid* if it is α-satisfied in every model.

To check (α)-satisfiability or validity, we use an **analytic tableau**.

Example: $\Diamond(p \wedge \alpha) \wedge \alpha$ is α-satisfiable in G3

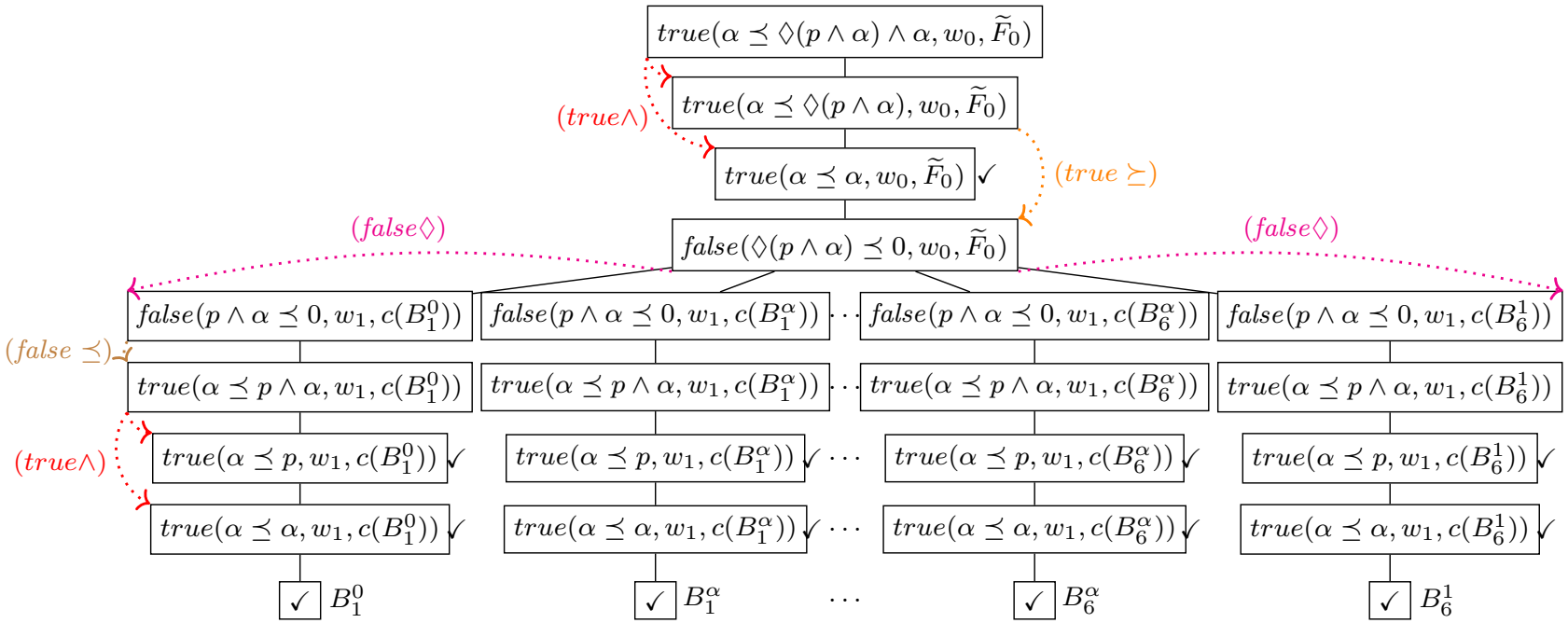


Figure: We only needed to fully expand one branch to prove α-satisfiability; here, we choose to show all possible branches for example purposes.

Example: $\Diamond(p \wedge \alpha) \wedge \alpha$ is not α-satisfiable in MV3

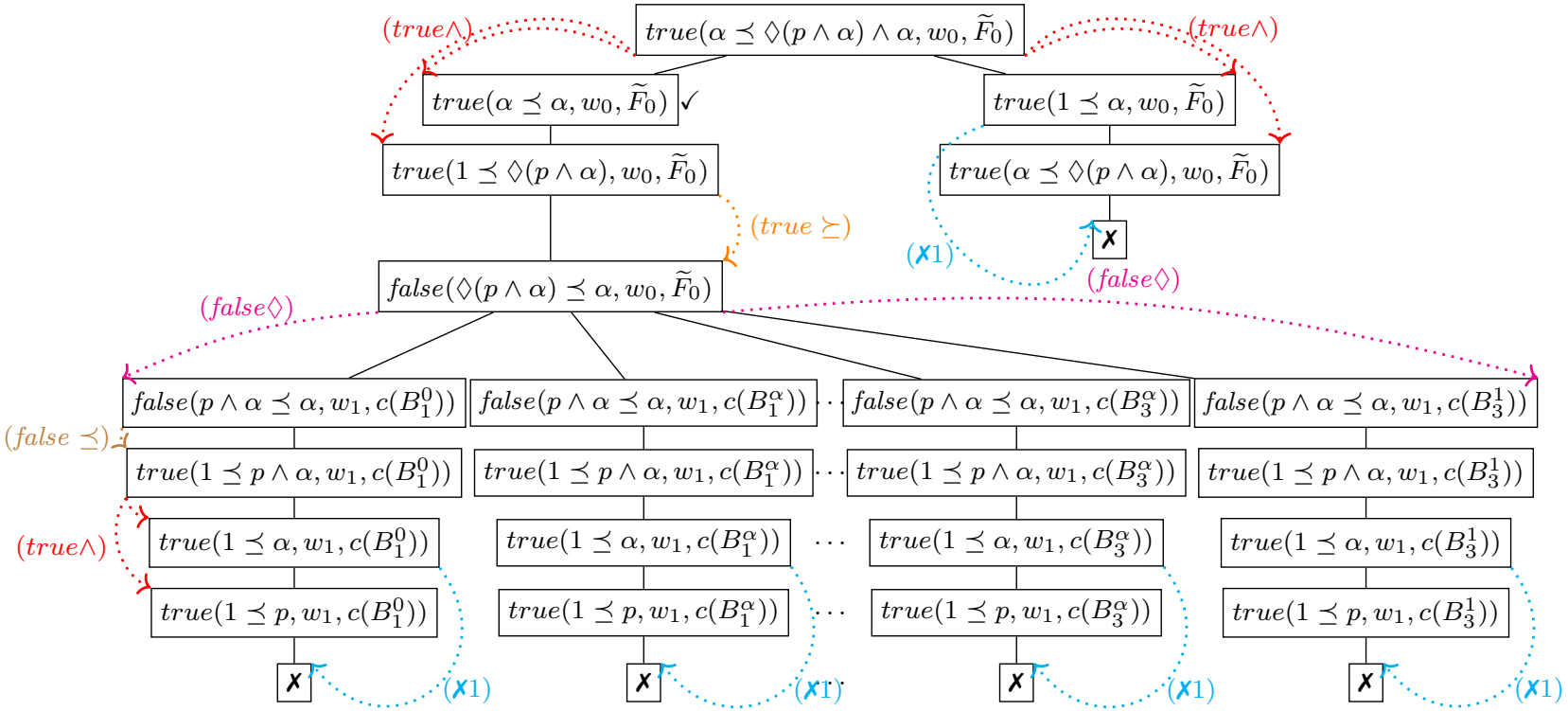


Figure: Applying the rules of the tableau system results in closing all branches.

Reasoning with SOLE

Live demo

Reasoning with SOLE

```
julia> using SoleLogics

julia> using SoleLogics.ManyValuedLogics

julia> using SoleReasoners

julia> using SoleLogics: LTLFP_F

julia> using SoleLogics.ManyValuedLogics: G3, Ł3, α

julia> p = Atom("p")
Atom{String}: p

julia> diamondF = diamond(LTLFP_F)
DiamondRelationalConnective{SoleLogics._GreaterRel}: ⟨>⟩

julia> φ = ∧(diamondF(∧(p,α)),α)
SyntaxBranch: ⟨>⟩(p ∧ α) ∧ α

julia> alphasat(MVLTLFPTableau, α, φ, G3)
true

julia> alphasat(MVLTLFPTableau, α, φ, Ł3)
false
```

Figure: How to code the previous example using SOLE.

Limitations and future work

Conclusions

Limitations

- The reasoning tool only supports **finite** many-valued logics
- Most real-work applications only need (very simple) fuzzy logics

Future work

- Reasoning support for **continuous**-chains (i.e., **BL-chains**)
- **Ad hoc** reasoning tools for simpler logics
- Advanced techniques for fuzzy and many-valued **model checking**

Thank you for your attention!

Questions?