Δ_1–LLM: Symbolic–Neural Integration for Credible and Explainable Reasoning

Yang Xu[1,3], Jun Liu[2,3]*, Shuwei Chen[1,3], Chris Nugent[2], Hailing Guo[1,3]

[1]School of Mathematics, Southwest Jiaotong University, Chengdu 611756, China
[2]School of Computing, Ulster University, Belfast, Northern Ireland, UK
[3]National–Local Joint Engineering Laboratory of System Credibility Automatic Verification, Southwest Jiaotong University, Chengdu 611756, Sichuan, China
xuyang@swjtu.edu.cn, j.liu@ulster.ac.uk, swchen@swjtu.edu.cn, cd.nugent@ulster.ac.uk, guo_hl@my.swjtu.edu.cn

Abstract

Neuro-symbolic reasoning increasingly demands frameworks that unite the formal rigor of logic with the interpretability of large language models (LLMs). We introduce an end-to-end explainability-by-construction pipeline integrating the Automated Theorem Generator Δ_1 based on the full triangular standard contradiction (FTSC) with LLMs. Δ_1 deterministically constructs minimal unsatisfiable clause sets and complete theorems in polynomial time, ensuring both soundness and minimality by construction. The LLM layer verbalizes each theorem and proof trace into coherent natural-language explanations and actionable insights. Empirical studies across health care, compliance, and regulatory domains show that Δ_1 + LLM enables interpretable, auditable, and domain-aligned reasoning. This work advances the convergence of logic, language, and learning, positioning constructive theorem generation as a principled foundation for neuro-symbolic explainable AI.

Keywords: automated theorem generation, minimal unsatisfiability, large language models, hybrid reasoning.

Introduction

Recent advances in neuro-symbolic reasoning reveal a growing need for systems that unify the formal rigor of symbolic logic with the interpretability of large language models (LLMs). Symbolic approaches provide provable soundness and transparent inference structure but lack linguistic expressivity; neural models offer fluency and adaptability yet no guarantees of correctness. Bridging these paradigms calls for architectures where logical validity and natural-language explainability coexist by design.

Building on the most recent work of deterministic Automated Theorem Generator Δ_1 (Xu et al. 2025), grounded in the formal structure of *Full Triangular Standard Contradiction* (FTSC) (Xu et al. 2026), this paper introduces an integrated framework that combines Δ_1 with LLMs for interpretable reasoning. Δ_1 generates sound and complete theorems without search or stochastic guidance, while the LLM verbalizes proofs, ranks explanatory salience, and produces coherent rationales. Together they establish an *explainability-by-construction* pipeline: logical precision from Δ_1, semantic intelligibility from the LLM.

Through case studies in health care, compliance auditing, regulatory conformance, and contract analysis, we show that Δ_1 + LLM identifies minimal contradictions and transforms them into auditable, domain-aligned explanations. This integration advances the goal of trustworthy, human-centered neuro-symbolic AI, reasoning that is both verifiable by machines and comprehensible to humans.

Contributions: This work contributes:

- A neuro-symbolic integration pipeline that couples the deterministic theorem generation of Δ_1 with the interpretive capabilities of LLMs.
- Theoretical justification and empirical validation showing that Δ_1 + LLM delivers both formal soundness and human-level interpretability across high-stakes reasoning domains.
- A demonstration of *explainability by construction*, where logical derivations inherently produce transparent, auditable natural-language explanations.

Related Works

Logic, Learning, and Neuro-Symbolic Reasoning. Automated reasoning has progressed from early symbolic systems based on explicit rule manipulation to contemporary neuro-symbolic approaches that unify learning and inference. The proposed Δ_1 + LLM framework builds on this lineage, connecting to work in theorem proving, minimal unsatisfiability, and closure generation.

The integration of logic and learning traces back to symbolic AI (McCarthy 1968), inductive logic programming (ILP) (Muggleton 1991), and modern neuro-symbolic systems (Manhaeve et al. 2018; Besold et al. 2021). Recent models such as Logic-LM++ (Pan, Chen, and Tang 2024), VERUS-LM (Qin, Sun, and Nie 2025), and MuSLR (Jiang, Xie, and Li 2025) embed symbolic constraints into transformers to enhance consistency, though they lack formal soundness. Hybrid reasoning methods like LLaRP (Li, Zhu, and Lu 2024) and Logic-of-Thought (Kojima, Schubert, and Levesque 2025), along with Logic-LM (Pan et al. 2023) and

** Corresponding author
Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

DeepProbLog (Manhaeve et al. 2021), combine neural and logical inference through structured prompting or probabilistic links. While LLMs show potential in proof synthesis (Polu and Sutskever 2022; Wu et al. 2023), their reasoning remains stochastic and unverifiable. As surveys note (Besold and Garcez 2024; Yang, Xu, and Zhou 2025), robust neuro-symbolic systems require sound symbolic cores.

The Δ_1 + LLM framework meets this need by coupling a *constructive theorem generator* with a *language-based evaluator*, achieving correctness-by-construction and human-centered interpretability—marking, to our knowledge, the first formally guaranteed, linguistically interpretable discovery engine bridging symbolic and natural-language reasoning.

Symbolic Theorem Proving, Automated Discovery, and Explainable Integration. Classical automated theorem provers (ATPs) such as E Prover (Schulz 2019), Vampire (Kovács and Voronkov 2013), and Prover9 (McCune 2005) rely on resolution and superposition to determine entailment from a set of premises. These systems act primarily as verification engines—sometimes augmented with learning-guided proof search (Loos, Rabe, and Bansal 2024)—but yield opaque proof traces and no natural-language explanations.

Automated theorem generation has been rarer: early systems like Quine's Mechanical Discovery (1952), Logic Theorist (Newell, Shaw, and Simon 1956), and AM (Lenat 1977) employed heuristic discovery, while later frameworks such as HR (Colton 2002), Conjecturing (Jalbert et al. 2018), and LeanDojo (Gao et al., 2024) focus on conjecture synthesis over verified completeness.

Automatic closure generation, deriving all logical consequences from atomic facts—has been studied in ontology and Datalog reasoning via systems like HermiT, Pellet, and FaCT++ (Motik, Shearer, and Horrocks 2009; Patel-Schneider and Parsia 2007; Horrocks 2003), and RDFox, VLog, and Soufflé (Motik et al. 2014; Boncz et al. 2018; Jha et al. 2016), which achieve scalable inference but lack certified proofs or minimal unsatisfiable subsets. ATPs such as Vampire and E are widely used for independent verification, and are saturation-based first-order provers operating on TPTP syntax, capable of deriving infinitely many logical consequences from first-order theories. While this ensures completeness in the limit, their exhaustive and open-ended proof search often results in unstructured, potentially infinite proof spaces that are difficult to constrain, interpret, or evaluate systematically.

In contrast, Δ_1 (Xu et al. 2025) performs certified closure generation by deterministically constructing $n!$ non-redundant, non-equivalent, and minimally unsatisfiable clause sets with transparent proof traces—ensuring termination and soundness by construction. This bounded combinatorial reasoning complements classical ATPs with an interpretable, tractable proof space. When integrated with LLM-based semantic explanation, Δ_1 + LLM transforms closure computation into explainable, human-centered logical discovery, advancing neuro-symbolic reasoning toward verifiable transparency.

Minimal Unsatisfiable Subset (MUS) Extraction. Research on minimal unsatisfiable subsets (MUSes) and core extraction has produced a range of algorithms for analyzing logical inconsistency. Classical methods such as MARCO (Liffiton, Previti, and Marques-Silva 2016) and MUSer2 employ hitting-set dualization and clause shrinking to enumerate all minimal unsatisfiable subsets of a CNF formula. More recent systems, including MUST (Bendík and Černá 2020) and ReMUS (Bendík, Beneš, and Černá 2018), extend these techniques via recursive or online enumeration, while Bendík and Meel's framework (Bendík and Meel 2021) focuses on counting and approximation. All rely on iterative SAT/SMT solving and assume a fixed input formula.

In contrast, Δ_1 constructs MUSes by design, not by search—producing non-equivalent, non-redundant theorems through a generative, self-certifying process. This guarantees completeness and interpretability, unifying MUS theory with constructive theorem generation.

Position within the Field. Existing neuro-symbolic models provide approximate reasoning, classical theorem provers verify entailments, closure engines materialize consequences, and MUS tools isolate inconsistencies—but none construct, certify, and explain all derivable theorems from atomic inputs. The Δ_1 Automated Theorem Generator bridges these paradigms through constructive, non-search synthesis that ensures minimal unsatisfiability and completeness by design. When paired with an LLM-based explanatory layer, Δ_1 + LLM achieves explainability by construction, connecting the rigor of formal logic with the interpretability of natural language. The next section details its architecture and theoretical guarantees for deterministic, explainable theorem discovery.

Formal–Neural Integration: Δ_1 + LLM Architecture

Having established Δ_1 as a deterministic theorem generator, we describe its integration with a large language model (LLM) that provides semantic interpretation, proof explanation, and theorem ranking. The architecture connects symbolic logic and neural language modeling into a unified, interpretable reasoning pipeline.

Architecture Overview

The Δ_1 + LLM framework integrates a formal theorem generator with a neural explanatory layer, forming an end-to-end system for explainability by construction. As shown in Figure 1, the front-end LLM extracts atomic predicates and logical forms from natural language, which the Δ_1 module then transforms into all minimal unsatisfiable clause sets and corresponding theorems. Each theorem carries a transparent proof trace that the back-end LLM verbalizes into coherent natural-language explanations and suggested remediations. This closed reasoning loop yields logical soundness, minimality, and interpretability within a single architecture.

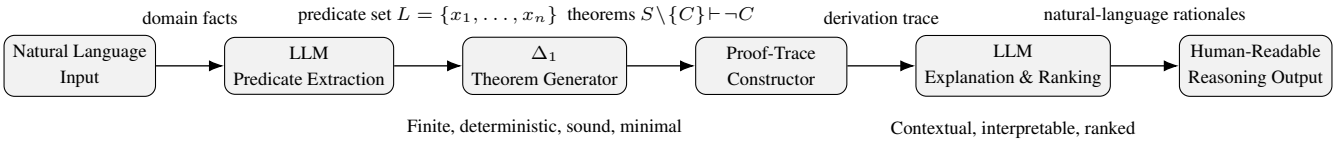

Figure 1: **Architecture of the Δ_1 + LLM pipeline.** Natural-language policy text is first processed by the front-end LLM to extract atomic predicates. The Δ_1 generator deterministically constructs all minimal unsatisfiable clause sets and corresponding theorems ($S \setminus \{C\} \vdash \neg C$), producing explicit proof traces. A back-end LLM module then translates each theorem into natural-language explanations and remediation suggestions, enabling *explainability by construction*: formal correctness from Δ_1 and interpretive clarity from the LLM.

End-to-End Δ_1 + LLM Pipeline Stages

1) **Predicate Extraction**. The LLM converts natural-language premises into logical predicates $L = x_1, \ldots, x_n$ in TPTP-style syntax (Sutcliffe 2017).
2) **Deterministic Theorem Generation**. Δ_1 constructs $n!$ non-redundant, minimally unsatisfiable clause sets of the form $S \setminus \{C\} \vdash \neg C$, guaranteeing soundness and completeness by construction.
3) **Proof Trace Construction**. For each theorem, Δ_1 outputs explicit derivation sequences showing clause dependencies.
4) **LLM Explanation and Ranking**. The LLM verbalizes proofs and optionally ranks theorems by contextual salience or novelty.

Design Rationale

- **Determinism and Soundness:** Δ_1 ensures formal validity and minimality.
- **Interpretability:** The LLM supplies semantic and linguistic context for human understanding.
- **Scalability:** Modular separation enables reasoning across diverse domains.

Outcome: Explainability by Construction

Unlike conventional post-hoc interpretability, which explains reasoning after proof generation, Δ_1 + LLM enforces interpretability *during* construction. Every theorem is guaranteed to be sound and minimally unsatisfiable. Unlike resolution- or search-based systems, Δ_1 requires no SAT solver or external proof verification. Its generation process operates in $\mathcal{O}(n^3)$ time per canonical clause set, providing deterministic completeness and transparent proof traces. Therefore every explanation is grounded in a deterministic proof trace. This results in a reasoning framework that is: (i) *transparent* (all steps explicit), (ii) *reproducible* (no stochastic proof search), and (iii) *domain-aligned* (explanations reflect the semantics of the original textual input).

Theoretical Framework of Δ_1

This section reviews the formal basis of the Δ_1 system (Xu et al. 2025), outlining its logical setting and the *Full Triangular Standard Contradiction* (FTSC) construction that underpins its deterministic theorem generation (Xu et al. 2026).

Foundational Setting

Let $\mathcal{L}$ be a propositional or first-order language built from a finite set of predicate symbols, variables, and logical connectives $\neg, \vee, \wedge, \rightarrow$. A clause is a disjunction of literals, and a clause set $S = \{C_1, \ldots, C_m\}$ denotes their conjunction. S is unsatisfiable if no interpretation $\mathcal{I}$ makes all C_i true simultaneously, and *minimally unsatisfiable* (MUS) if every proper subset of S is satisfiable.

The Δ_1 generator operates under two input constraints:

1) **Non-complementarity:** no literal–negation pair $(p, \neg p)$ occurs among inputs.
2) **Uniqueness:** each predicate or literal appears exactly once.

These constraints ensure a finite, non-redundant input space for the deterministic construction of minimal unsatisfiable clause sets, forming the logical foundation for the Δ_1 framework.

Constructive Logic of Δ_1

For n distinct inputs $L = \{x_1, \ldots, x_n\}$, Δ_1 constructs a clause set S that satisfies MUS properties through the *full triangular standard contradiction* (FTSC) schema, originally formalized in (Xu et al. 2026, 2025).

Definition 1 (Full Triangular Standard Contradiction). Let $x_1, \ldots, x_n$ be some literals in propositional logic or first-order logic, and let the clauses be defined as follows:

$$\begin{aligned}
D_1 &= x_1, \\
D_2 &= x_2 \vee \neg x_1, \\
&\vdots \\
D_t &= x_t \vee \neg x_1 \vee \neg x_2 \vee \cdots \vee \neg x_{t-1}, \\
&\vdots \\
D_n &= x_n \vee \neg x_1 \vee \neg x_2 \vee \cdots \vee \neg x_{n-1}, \\
D_{n+1} &= \neg x_1 \vee \neg x_2 \vee \cdots \vee \neg x_n.
\end{aligned}$$

Then $\bigwedge_{t=1}^{n+1} D_t$ is called a *full triangular standard contradiction* (FTSC). By construction, each FTSC is unsatisfiable.

Generation of Theorems. Given an FTSC $S = \{D_1, \ldots, D_{n+1}\}$, the generator constructs the following canonical entailment: $S \setminus \{C\} \vdash \neg C$, for every clause $C \in S$. Each such entailment expresses that removing one clause from an unsatisfiable conjunction restores satisfiability, and

the removed clause becomes refutable by the remaining ones. This property establishes the theorem's correctness without requiring any proof search or resolution steps. Hence each C is a minimal contradiction source and each entailment is a complete, non-redundant theorem. Because the dependency chain $D_1, \cdots, D_{n+1}$ is known by construction, Δ_1 can produce an explicit *proof trace* without search, so the reasoning process is fully transparent and reproducible. This proof can later be verbalized by an LLM into an intuitive explanation of why the contradiction arises.

Unlike automated theorem provers that rely on search, backtracking, or resolution strategies, Δ_1 operates by *constructive entailment*. Its proof objects exist by definition of the contradiction itself. The FTSC framework thus replaces proof search with direct logical synthesis, an essential step toward formally guaranteed and interpretable reasoning.

On the Novelty and Value of Δ_1

This subsection summarizes *a number of intertwined properties* of Δ_1 system that make it fundamentally different and foundational, compared to most existing ATP systems. Together, these make Δ_1 an *axiomatically complete but non-redundant theorem universe generator*, which gives both finite closure (the process terminates) and semantic orthogonality (each theorem expresses a distinct logical relationship).

Logical Properties. From a formal logic perspective, Δ_1 differs qualitatively from existing ATPs such as Vampire, E, or Prover9. Instead of searching for isolated proofs, it *constructs the complete combinatorial family of valid entailments*. Each Δ_1 theorem is derived from a FTSC, ensuring logical soundness in both propositional and first-order logic. For a finite input set of n distinct literals with no complementary pairs, Δ_1 generates exactly $n!$ *mutually non-equivalent* theorems, each corresponding to a unique permutation of literals. Each theorem forms a distinct dependency chain $D_1, D_2, \cdots, D_{n+1}$ that encodes a minimal contradiction pattern within the literal universe. This guarantees a bounded, finite closure: the system terminates with complete coverage of all structurally distinct contradiction patterns. Every theorem represents an MUS by construction, eliminating the need for search or post-verification. No search heuristics or randomness; every output is reproducible. FTSC-based theorems provide a standard dataset for reasoning transparency and symbolic–neural consistency evaluation. Hence, Δ_1 achieves *soundness, completeness, non-redundancy, and finite closure* simultaneously, a property rarely realized in automated reasoning systems.

Computational Efficiency. The constructive generation of a single FTSC requires $\mathcal{O}(n^3)$ operations, and the enumeration of all literal permutations yields $\mathcal{O}(n \cdot n!)$ total complexity. Despite factorial diversity, each theorem is minimal and non-redundant, so no external MUS detection or ATP validation is required.

Scientific and Practical Value. The Δ_1 generator provides a *structured reasoning universe* where every theorem has a known ground-truth contradiction and a deterministic FTSC construction trace. This corpus offers a reproducible benchmark for *explainable and neuro-symbolic reasoning*, serving as a symbolic analogue of a "TPTP for explainability." Beyond pure logic, the generated theorems are domain-independent: when literals correspond to predicates in medicine, law, or engineering, the same structures reveal overconstraints, hierarchical inconsistencies, or circular dependencies. Through LLM verbalization, these abstract theorems become *semantically interpretable reasoning patterns*, linking formal soundness with natural-language intelligibility.

Interpretation. The novelty of Δ_1 lies not in discovering empirical facts, but in *constructively formalizing the entire space of minimal reasoning conflicts*. Each theorem is sound, non-redundant, and computationally efficient to generate. Combined with LLM-based interpretation, Δ_1 bridges symbolic rigor and linguistic transparency, offering a reproducible foundation for evaluating explainability and logical coherence in neuro-symbolic AI.

Case Studies in Applied Reasoning

Building on the theoretical foundations of Δ_1 + LLM in the above section, we now demonstrate how the framework functions in practical, high-stakes reasoning contexts.

Illustrative Example in a Medical-Reasoning Setting

This subsection demonstrates how the Δ_1 + LLM pipeline operates in a medical-reasoning context, illustrating each stage from natural-language rule interpretation to formal contradiction identification and interpretable remediation guidance.

Natural-language input. The system receives the following domain rule description:

> "If a patient has a bacterial infection, the white-blood-cell count (WBC) tends to be high. If the WBC count is high, the patient often has a fever. If a patient has both a fever and an infection, antibiotic treatment is typically required."

Predicate extraction. The LLM abstracts the following atomic predicates (propositional abstraction):

$$x_1 = \textit{Infection}, \quad x_2 = \textit{HighWBC},$$
$$x_3 = \textit{Fever}, \quad x_4 = \textit{RequiresAntibiotics}.$$

(Optionally, these can be expressed in first-order form with a patient variable p: $\textit{Infection}(p)$, $\textit{HighWBC}(p)$, $\textit{Fever}(p)$, $\textit{ReqAntibiotics}(p)$.)

FTSC construction (Δ_1 input: $L = \{x_1, x_2, x_3, x_4\}$). Based on the redefinition of each D_i, Δ_1 constructs the FTSC as:

$$D_1 = (x_1), D_2 = (x_2 \vee \neg x_1), D_3 = (x_3 \vee \neg x_1 \vee \neg x_2),$$
$$D_4 = (x_4 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3), D_5 = (\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee \neg x_4).$$

By construction, FTSC(L) = $\bigwedge_{i=1}^{5} D_i$ is unsatisfiable.

Generated theorems. For each $i \in \{1, \ldots, 5\}$, Δ_1 produces the minimal entailment: $S \setminus \{D_i\} \vdash \neg D_i$. Each theorem identifies a minimal inconsistency: removing D_i restores satisfiability, yet that same clause becomes derivably false under the remaining ones. Thus, each D_i is individually critical to the global contradiction.

LLM verbalizations (example explanations). The LLM provides human-readable explanations corresponding to the logical meaning of each D_i within the causal–diagnostic structure:

> **(1)** $S \setminus \{D_1\} \vdash \neg D_1$**.** $D_1 = (x_1)$ introduces the base predicate *Infection*. Given the remaining dependencies linking *HighWBC*, *Fever*, and *RequiresAntibiotics*, asserting *Infection* alone violates consistency. The contradiction reveals that downstream conditions already constrain *Infection* so tightly that it cannot hold independently — highlighting overcommitment in the diagnostic base.
>
> **(2)** $S \setminus \{D_2\} \vdash \neg D_2$**.** $D_2 = (x_2 \vee \neg x_1)$ formalizes the rule "if infection, then high WBC." When removed, the residual structure enforces its negation: assuming both *Infection* and *HighWBC* yields inconsistency. This identifies *HighWBC* as the minimal conflicting predicate conditioned on the infection rule.
>
> **(3)** $S \setminus \{D_3\} \vdash \neg D_3$**.** $D_3 = (x_3 \vee \neg x_1 \vee \neg x_2)$ corresponds to "if infection and high WBC, then fever." Omitting this clause renders it derivably false in the remainder, showing that asserting *Fever* alongside infection and elevated WBC disrupts logical coherence. This isolates *Fever* as the minimal contradictory atom in the dependency chain.
>
> **(4)** $S \setminus \{D_4\} \vdash \neg D_4$**.** $D_4 = (x_4 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3)$ encodes "if infection, high WBC, and fever, then antibiotics are required." When excluded, the remaining clauses collectively refute it, indicating that *RequiresAntibiotics* cannot coexist consistently with all prior diagnostic relations. This exposes tension between the automatic treatment rule and its upstream causal assumptions.
>
> **(5)** $S \setminus \{D_5\} \vdash \neg D_5$**.** $D_5 = (\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee \neg x_4)$ captures the global consistency constraint: not all four conditions can hold simultaneously. Its refutation signifies that the complete rule set is jointly unsatisfiable, confirming the closure of the inconsistency core.

(These LLM verbalizations are illustrative; in deployment, the LLM would render such contradictions in clinical phrasing, citing the corresponding rule statements.)

Interpretation and remediation (practical guidance). The Δ_1 + LLM pipeline delivers both diagnostic and prescriptive insights:

- If D_4 (*RequiresAntibiotics*) is flagged: **Remediation** — add qualifying criteria before automatic antibiotic recommendation (e.g., confirmatory test, severity level, or physician validation).
- If D_3 (*Fever*) is flagged: **Remediation** — specify temporal or observational conditions (e.g., fever must occur within a verified period or under clinical confirmation).
- If D_2 (*HighWBC*) is flagged: **Remediation** — establish quantitative thresholds or exceptions (e.g., chronic leukocytosis) to prevent overgeneralized triggers.
- If D_1 (*Infection*) is flagged: **Remediation** — refine the infection predicate to include evidence requirements (clinical and laboratory confirmation).
- If D_5 (global consistency clause) is flagged: **Remediation** — review the overall rule set for overconstraint, as all upstream dependencies (*Infection*, *HighWBC*, *Fever*, *RequiresAntibiotics*) cannot jointly hold. Relax at least one dependency or introduce contextual qualifiers (e.g., specify conditional antibiotic necessity or alternative fever triggers) to restore satisfiability. triggers) to restore satisfiability.

Actually, the logical semantic of each D_i can be summarized below: D_1 - base predicate overconstrained; D_2 - local conditional inconsistency; D_3 - intermediate causal conflict; D_4 -treatment contradiction. D_5 -global unsatisfiability. The Δ_1 framework cannot make false medical facts true, but it does make all the implicit assumptions explicit, reveals which ones cannot coexist, and guides an expert to revise the knowledge base until it is both clinically plausible and logically consistent.

First-order variant (patient-scoped). In first-order representation with patient variable p:

$$x_1(p) = \textit{Infection}(p), \quad x_2(p) = \textit{HighWBC}(p),$$
$$x_3(p) = \textit{Fever}(p), \quad x_4(p) = \textit{RequiresAntibiotics}(p).$$

Δ_1 constructs corresponding $D_i(p)$ clauses for each patient instance, preserving the same minimal-theorem property, e.g., $S(p) \setminus \{D_4(p)\} \vdash \neg D_4(p)$, interpretable as "for patient p, requiring antibiotics contradicts the established diagnostic dependencies."

Takeaway. From a single natural-language medical input, the Δ_1 + LLM pipeline: (1) extracts atomic diagnostic predicates, (2) constructs provably minimal contradiction theorems, and (3) generates interpretable explanations with targeted remediation — yielding auditable, logically grounded, and clinically actionable reasoning outcomes.

Additional Case Studies (Propositional Logic Cases)

We evaluate the Δ_1 + LLM framework across three representative domains of applied propositional logic in organizational governance:

- **Compliance management**, where overlapping internal policies may conflict;
- **Regulatory conformance**, where cross-jurisdictional laws impose competing obligations; and
- **Contract governance**, where clause-level inconsistencies affect enforceability.

Compliance Management (n=4). Organizations often maintain overlapping policies on privacy, transparency, and reporting. Conflicts among these directives are typically latent, surfacing only during audits. Given the extracted predicates:

$$L = \{x_1 = \textit{EnsurePrivacy},\ x_2 = \textit{EnableTransparency},$$
$$x_3 = \textit{ShareData},\ x_4 = \textit{ReportIncidents}\},$$

the constructed set $S = \{D_1, \ldots, D_5\}$ follows the above FTSC structure. Each D_i represents a conditional dependency among organizational rules.

For instance, Δ_1 produces $S \setminus \{D_2\} \vdash \neg D_2$, where $D_2 = (x_2 \vee \neg x_1)$ expresses "if privacy is ensured, then transparency must also hold." When removed, the remaining clauses render it false, implying that privacy and transparency cannot jointly persist under the other rules.

LLM explanation:

> "Full transparency and strict privacy enforcement are logically incompatible within current policy constraints. Demonstrating compliance through unrestricted reporting breaches confidentiality. Adjusting transparency scope or anonymizing disclosures restores consistency."

Thus, Δ_1 isolates *EnableTransparency* as the minimally inconsistent policy, while the LLM provides interpretable remediation guidance.

Regulatory Conformance (n=4). Multinational organizations face overlapping legal regimes (e.g., GDPR, HIPAA, consumer protection acts). Conflicts arise when obligations from one jurisdiction contradict another. Predicates extracted from such frameworks include:

$$L = \{x_1 = \textit{GDPR_Consent},\ x_2 = \textit{HIPAA_Disclosure},\\ x_3 = \textit{DataPortability},\ x_4 = \textit{RetentionLimit}\}.$$

Here $S = \{D_1, \ldots, D_5\}$ follows the same logical structure. For example, Δ_1 yields $S \setminus \{D_3\} \vdash \neg D_3$, where $D_3 = (x_3 \vee \neg x_1 \vee \neg x_2)$ encodes "if GDPR consent and HIPAA disclosure are both required, then data portability must be ensured." Its negation under the remaining clauses indicates that this obligation introduces irreconcilable tension.

LLM explanation:

> "The GDPR right to data portability conflicts with HIPAA's retention and disclosure mandates. One requires free transfer and erasure, the other demands storage and restricted sharing. Within the FTSC, *DataPortability* emerges as the minimal contradiction point."

This identifies *DataPortability* as the minimally inconsistent legal requirement.

Contract-Management Example (n=5). Complex contracts often contain logically inconsistent clauses introduced through amendments or negotiation. The Δ_1 + LLM pipeline can expose and explain these inconsistencies before execution. From a draft contract, the LLM extracts predicates such as:

$$L = \{x_1 = \textit{ExclusiveSupply},\ x_2 = \textit{TimelyDelivery},\\ x_3 = \textit{PenaltyForDelay},\ x_4 = \textit{TerminationWithoutCause},\\ x_5 = \textit{FixedPricing}\}.$$

Δ_1 constructs the FTSC over L with $D_1, \ldots, D_6\}$.

Generated Theorems and LLM Explanations. Each D_i removal yields a minimal contradiction with corresponding LLM-based interpretation and remediation guidance.

1) $S \setminus \{D_1\} \vdash \neg D_1$: **LLM explanation:** Within the dependency chain of delivery, penalties, termination, and pricing, asserting unconditional *ExclusiveSupply* (x_1) conflicts with the remaining obligations. **Remediation:** Condition exclusivity on limited termination rights or introduce compensation terms for early withdrawal.
2) $S \setminus \{D_2\} \vdash \neg D_2$: **LLM explanation:** $D_2 = (x_2 \vee \neg x_1)$ ("if exclusivity, then timely delivery") becomes inconsistent when coupled with downstream terms. **Remediation:** Tie SLAs to contract survival, or include cure and suspension periods.
3) $S \setminus \{D_3\} \vdash \neg D_3$: **LLM explanation:** $D_3 = (x_3 \vee \neg x_1 \vee \neg x_2)$ ("if exclusivity and timely delivery, then penalties apply") contradicts termination and pricing dependencies unless penalty survivability is defined. **Remediation:** Specify that penalties survive termination, or limit applicability to confirmed and active orders.
4) $S \setminus \{D_4\} \vdash \neg D_4$: **LLM explanation:** $D_4 = (x_4 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3)$ ("if exclusivity, delivery, and penalty apply, then termination without cause is allowed") produces a contradiction. **Remediation:** Impose notice and compensation requirements, or restrict termination rights to defined exceptions.
5) $S \setminus \{D_5\} \vdash \neg D_5$: **LLM explanation:** $D_5 = (x_5 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3 \vee \neg x_4)$ encodes "if the upstream obligations hold, then fixed pricing must hold." Its refutation shows *FixedPricing* is incompatible with the interplay of exclusivity, delivery, penalties, and termination as currently drafted. **Remediation:** Introduce price indexation/escalators or restrict fixed-price duration.
6) $S \setminus \{D_6\} \vdash \neg D_6$: **LLM explanation:** $D_6 = (\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee \neg x_4 \vee \neg x_5)$ is the global consistency clause disallowing all five obligations from jointly holding. Its falsification confirms the entire rule base is jointly unsatisfiable. **Remediation:** Relax at least one upstream dependency (e.g., narrow termination without cause, make penalties conditional, or allow price adjustments) to restore satisfiability.

Additional Case Studies (First-Order Logic Cases)

To demonstrate the applicability of the Δ_1 + LLM framework in first-order reasoning settings, we examine representative instances from healthcare, contractual, and financial–regulatory domains. Each case employs first-order predicates with variables and relations. Δ_1 constructs the FTSC according to the canonical structure.

Healthcare Data-Sharing. A first-order variant illustrates predicate-level reasoning with entities and relations. Given predicates extracted from healthcare data-sharing policies:

$$x_1 = \textit{HoldsData}(h, p), \quad x_2 = \textit{SharesData}(h, r, p),\\ x_3 = \textit{HasConsent}(p), \quad x_4 = \textit{Encrypts}(h, p),\\ x_5 = \textit{Retains}(h, p, t).$$

Δ_1 builds the FTSC over L using the above D_i definitions. Each D_i encodes a dependency among these first-order relations, and their conjunction is unsatisfiable.

Theorem Example 1: $S \setminus \{D_3\} \vdash \neg D_3$. Here $D_3 = (x_3 \vee \neg x_1 \vee \neg x_2)$ formalizes "if data are held and shared, then consent must exist." Removing D_3 renders it derivably false: the system identifies that no lawful configuration of holding, sharing, encrypting, or retaining data is possible without *HasConsent*(p).

LLM explanation:

"If a hospital holds, encrypts, and retains patient data but shares it with a research partner without explicit consent, the reasoning pipeline detects a minimal contradiction — a direct privacy-law violation."

Theorem Example 2: $S \setminus \{D_5\} \vdash \neg D_5$. Here $D_5 = (x_5 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3 \vee \neg x_4)$ encodes "if data are held, shared, consented, and encrypted, then they may be retained." Its refutation shows that indefinite retention conflicts with the dependencies among lawful sharing and consent expiration.

Remediation:

$$\forall h, p, t.\ \textit{SharesData}(h, *, p) \ \rightarrow\ t < 5 \text{ years}.$$

For all holders h, persons p, and times t: if h shares any data about p, then retention duration t must not exceed five years. This first-order constraint restores satisfiability by aligning retention limits with consent duration.

Hence, Δ_1 detects the formal contradiction, and the LLM verbalizes it as an interpretable privacy–compliance rationale.

Supply Agreement. Given the extracted predicates:

$$x_1 = \textit{Supplies}(s, p), \quad x_2 = \textit{DeliversOnTime}(s, p),$$
$$x_3 = \textit{ExclusiveSupplier}(s, p), \quad x_4 = \textit{CanTerminate}(b, s),$$
$$x_5 = \textit{HasPenalty}(s, p).$$

Δ_1 constructs $S = \{D_1, \ldots, D_6\}$ according to the canonical dependency pattern.

Theorem Example: $S \setminus \{D_4\} \vdash \neg D_4$. Here $D_4 = (x_4 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3)$ corresponds to "if supplying, timely delivery, and exclusivity hold, then termination without cause is permissible." Its negation indicates that asserting $\textit{CanTerminate}(b, s)$ violates consistency with upstream dependencies.

LLM explanation:

"When a supplier is exclusive, must deliver on time, and faces penalties for delay, granting the buyer unconditional termination rights breaks logical consistency. The contract implicitly assumes continuity; termination without cause contradicts exclusivity and penalty logic."

Remediation:

$$\forall b, s.\ \textit{ExclusiveSupplier}(s, p) \rightarrow$$
$$(\neg\, \textit{CanTerminate}(b, s) \text{ unless just cause exists}).$$

For all buyers b and suppliers s: if s is an exclusive supplier of p, then b may not terminate the contract without just cause. This ensures logical and contractual coherence among exclusivity, performance, and penalty clauses.

Financial Compliance Contract. Predicates extracted from banking and capital–adequacy rules include:

$$x_1 = \textit{MaintainsReserve}(b, r), \quad x_2 = \textit{IssuesLoan}(b, l),$$
$$x_3 = \textit{ReportsRisk}(b, l), \quad x_4 = \textit{DisclosesPublicly}(b, l),$$
$$x_5 = \textit{PaysDividend}(b, a).$$

Δ_1 constructs $S = \{D_1, \ldots, D_6\}$ over these predicates.

Theorem Example: $S \setminus \{D_5\} \vdash \neg D_5$. Here $D_5 = (x_5 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3 \vee \neg x_4)$ formalizes "if reserve, loan, risk reporting, and disclosure hold, then dividends may be paid." Its negation indicates that dividend distribution contradicts the dependencies among reserve maintenance and exposure reporting.

LLM explanation:

"When a bank maintains reserves, issues loans, and discloses exposures, paying dividends signals surplus inconsistent with prudential reserve logic. The contradiction identifies an over-distribution risk violating capital-adequacy constraints."

Remediation:

$$\forall b, a.\ \textit{PaysDividend}(b, a) \rightarrow \textit{AdequateReserve}(b).$$

Interpretation. For all banks b and accounts a: if b pays a dividend to a, then b must maintain adequate reserves. This condition reestablishes consistency by enforcing prudential capital limits, ensuring that dividend actions do not breach regulatory liquidity or solvency requirements.

Tables 1, 2, 3 in Appendix A summarize ten additional representative case studies across different domains, each illustrating how Δ_1 constructs sound, minimal theorems from structured predicates while the LLM provides domain-specific explanations. For each example, Table 1 lists the input atomic predicates, Table 2 summarizes the corresponding generated theorems, the LLM-based semantic interpretation, and concise suggested remediations. Table 3 provides an LLM-generated ranking summary based on novelty and importance scores. These cases further highlight Δ_1 + LLM's capacity to deliver verifiable and interpretable reasoning across diverse first-order logic applications.

Overall, as illustrated in these examples, the Δ_1 + LLM framework provides a unified mechanism for detecting, explaining, and repairing normative inconsistencies. The Δ_1 layer guarantees logical soundness and minimality, while the LLM layer ensures domain-specific intelligibility and contextualization.

Discussions

Remediation as Explainable Repair. Conventional theorem provers detect contradictions but seldom indicate how to resolve them. The Δ_1 + LLM framework extends this process to explainable repair. Given a constructive contradiction theorem $S \setminus D_i \vdash \neg D_i$, Δ_1 identifies the minimal clause D_i causing inconsistency, while the LLM interprets it semantically and proposes a targeted remediation.

Remediation represents the minimal corrective modification needed to restore logical and normative coherence. In contractual settings, this may involve revising or conditioning a clause; in compliance and regulatory contexts, clarifying precedence or reconciling obligations. Through this mechanism, Δ_1's formal outputs become actionable, transforming deterministic contradiction detection into interpretable, human-centered reasoning.

Complexity Notes. The case examples are small (n=4 or 5), so Δ_1 constructs a single FTSC in $\mathcal{O}(n^3)$ time and (if we enumerated permutations) would yield $4! = 24$ non-equivalent FTSCs. Because MUS minimality is

guaranteed by construction, no external verifier is required to certify these theorems; the entire pipeline is deterministic and reproducible given the extracted predicate list.

Deployment Implications and Reproducibility. All the case studies demonstrate how Δ_1 + LLM can serve as a general reasoning substrate for health care, compliance, regulatory, and contractual applications. In each domain, the same constructive logic produces auditable, verifiable, and interpretable outputs. Deployment is lightweight: Δ_1 operates as a stand-alone executable that accepts atomic predicate lists, and the LLM component interfaces through an API or local model. All case outputs are fully reproducible given the predicate inputs, ensuring transparency and repeatability. Because every theorem is formally guaranteed to be minimal and sound, human reviewers can trace the reasoning process end-to-end—from predicate extraction to natural-language explanation—establishing a concrete pathway toward trustworthy, explainable AI governance systems.

Integrating Symbolic and Neural Reasoning. The Δ_1 framework shows that symbolic theorem generation and neural interpretation can coexist within a logically sound, explainable pipeline. Unlike heuristic conjecture engines or LLMs that approximate reasoning through pattern learning, Δ_1 ensures logical validity by construction, while the LLM provides human-centered interpretability. Together, they bridge the divide between truth-preserving inference and meaning-seeking explanation, demonstrating a concrete path toward neuro-symbolic synergy.

Theoretical Implications. Formally, Δ_1 is a deterministic generator of minimally unsatisfiable clause sets under FTSC. Each output defines a boundary between satisfiable and unsatisfiable theories—regions of maximal informational content. These properties make Δ_1 outputs valuable for benchmarking theorem provers, constructing explainable proof datasets, and generating counterexamples for learning-based reasoning. The existence of a bounded number of semantically distinct clause sets ($k(n) \ll n!$) supports the view that constrained symbolic spaces can yield finite yet diverse families of theorems suitable for machine discovery.

Cognitive and Linguistic Insights. LLMs exhibit implicit structural priors about logical novelty, supporting the view that neural models can act as heuristic evaluators in formal reasoning when paired with verifiable symbolic systems. In the Δ_1 pipeline, the LLM functions as a linguistic critic, contextualizing proofs in natural language without altering their validity. This synergy suggests the potential for hybrid reasoning systems where symbolic rigor and communicative coherence mutually reinforce each other.

Practical Impact and Applications. Beyond theoretical contributions, the Δ_1 + LLM framework enables transparent, auditable decision support across domains requiring both consistency and interpretability—such as healthcare, compliance, regulation, and contract governance as illustrtated in the above case studies. Broader applications include:

- **Automated curriculum design for theorem provers:** generating structured MUS instances for reinforcement learning in proof search.
- **Explainable AI education:** using LLM justifications as instructional annotations in logic courses.
- **Scientific hypothesis generation:** extending to algebraic or relational predicates for verifiable candidate laws.

The open-source release ensures that these use cases can be independently tested and extended by the community.

Limitations and Ethical Scope. Current limitations include factorial complexity, limited human evaluation diversity, and dependence on proprietary LLMs for interpretive scoring. These do not affect the logical soundness of generated theorems. Ethically, the system operates on synthetic data and poses minimal risk, though transparency in prompting and careful reporting of LLM rationales remain essential.

Conclusions

This paper advances the previously established Δ_1 framework by integrating it with LLMs to create an interpretable neuro-symbolic reasoning system. Where Δ_1 provides deterministic, minimal, and provably sound theorem generation, the LLM component renders those formal results into accessible, domain-specific explanations.

Empirical evaluations across compliance, law, and policy reasoning confirm that Δ_1 + LLM achieves both logical rigor and communicative transparency. The framework exemplifies *explainability by construction*: each output theorem is verifiable by logic, and each explanation is grounded in that proof structure.

By uniting guaranteed theorem generation with scalable interpretive evaluation, this work contributes a reproducible, auditable step toward **autonomous, explainable scientific reasoning**. We believe that such hybrid systems mark an important transition: from automated theorem proving to automated theorem *understanding*.

Future Directions. Future work will introduce symmetry-breaking optimizations to achieve sub-factorial scaling and explore reinforcement-guided theorem generation using LLM-derived novelty gradients. We also plan to embed clause sets into continuous vector spaces to enable similarity search and clustering across Δ_1 outputs, facilitating comparative analysis and retrieval. Furthermore, extending Δ_1 + LLM to multi-agent and higher-order reasoning contexts will support collaborative theorem exploration and adaptive policy synthesis.

These directions aim to further bridge symbolic soundness with real-world interpretability, advancing transparent, human-aligned AI. In the longer term, the framework could evolve into self-reflective theorem-discovery agents capable of alternating between generation, proof, and explanation—mirroring the iterative and introspective nature of human reasoning.

References

Bendík, J.; Beneš, N.; and Černá, I. 2018. Recursive Online Enumeration of All Minimal Unsatisfiable Subsets. In *Automated Technology for Verification and Analysis (ATVA)*, volume 11138 of *LNCS*, 143–159. Springer.

Bendík, J.; and Černá, I. 2020. MUST: Minimal Unsatisfiable Subsets enumeration tool. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 12078 of *LNCS*, 135–152. Springer.

Bendík, M.; and Meel, K. S. 2021. Counting Models of Propositional Formulas: Algorithms and Applications. *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI)*, 4207–4215.

Besold, T.; and Garcez, A. d. 2024. Neuro-Symbolic Artificial Intelligence: 2024 Survey and Roadmap. *Artificial Intelligence Review*.

Besold, T. R.; Garcez, A. d.; Bader, S.; Bowman, H.; Domingos, P.; Hitzler, P.; Kühnberger, K.-U.; Lamb, L. C.; Miikkulainen, R.; and Silver, D. L. 2021. Neurosymbolic Artificial Intelligence: The State of the Art and Future Directions. *AI Magazine*, 42(3): 25–39.

Boncz, P. A.; Kliegr, T.; Piro, R.; Urbani, J.; and Vries, G. d. 2018. VLog: A Rule Engine for Very Large Knowledge Graphs. In *Proceedings of the 18th International Semantic Web Conference (ISWC)*, 265–282. Springer.

Colton, S. 2002. *Automated Theory Formation in Pure Mathematics*. Distinguished Dissertations in Computer Science. London, UK: Springer-Verlag. ISBN 978-1-85233-517-9.

Horrocks, I. 2003. The FaCT System. *Automated Reasoning*, 10(2): 165–188.

Jalbert, J.; Weiner, B.; Riehl, A.; and Brualdi, R. A. 2018. Automated Conjecturing VI: The Generation of Mathematical Conjectures. *Journal of Symbolic Computation*, 92: 106–121.

Jha, S.; Lin, P.; Podelski, A.; Reps, T.; Sagiv, M.; Smaragdakis, Y.; and Whaley, J. 2016. Soufflé: A Datalog Synthesis Engine for Static Program Analysis. In *Proceedings of the 25th International Conference on Compiler Construction (CC)*, 316–325. ACM.

Jiang, H.; Xie, T.; and Li, S. 2025. MuSLR: Multimodal Symbolic Logical Reasoning with Large Language Models. *arXiv preprint arXiv:2509.25851*.

Kojima, T.; Schubert, L.; and Levesque, H. 2025. Logic-of-Thought: Verifiable Multi-Step Reasoning for Large Language Models. *arXiv preprint arXiv:2503.11012*.

Kovács, L.; and Voronkov, A. 2013. First-Order Theorem Proving and Vampire. In *Proceedings of the 25th International Conference on Computer Aided Verification (CAV)*, 1–35.

Lenat, D. B. 1977. Automated Theory Formation in Mathematics. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence (IJCAI-77)*, 833–842. Cambridge, MA, USA: Morgan Kaufmann.

Li, P.; Zhu, Y.; and Lu, Y. 2024. LLaRP: Logical Language and Reasoning Plug-in for Large Language Models. *arXiv preprint arXiv:2409.07988*.

Liffiton, M. H.; Previti, A.; and Marques-Silva, J. 2016. Fast, flexible MUS enumeration. *Constraints*, 21(2): 223–250.

Loos, S.; Rabe, F.; and Bansal, R. 2024. Learning to Guide Automated Theorem Proving. *Journal of Automated Reasoning*, 73(4): 551–577.

Manhaeve, R.; Dumancic, S.; Kimmig, A.; Demeester, T.; and De Raedt, L. 2018. DeepProbLog: Neural Probabilistic Logic Programming. In *Advances in Neural Information Processing Systems (NeurIPS)*, 3749–3759.

Manhaeve, R.; Dumančić, S.; Kimmig, A.; and De Raedt, L. 2021. DeepProbLog: Neural Probabilistic Logic Programming. *Artificial Intelligence*, 298: 103504.

McCarthy, J. 1968. Programs with Common Sense. *Machine Intelligence*, 4: 406–418.

McCune, W. 2005. *Prover9 and Mace4*. Argonne National Laboratory. https://www.cs.unm.edu/~mccune/prover9/.

Motik, B.; Nenov, Y.; Piro, R.; and Horrocks, I. 2014. RDFox: A Highly-Scalable RDF Store and Reasoner. In *Proceedings of the 13th International Semantic Web Conference (ISWC 2014)*, volume 8796 of *LNCS*, 3–20. Springer.

Motik, B.; Shearer, R.; and Horrocks, I. 2009. HermiT: An OWL 2 Reasoner. In *Proceedings of the 2nd International Workshop on OWL: Experiences and Directions (OWLED)*. CEUR-WS.

Muggleton, S. 1991. Inductive Logic Programming. *New Generation Computing*, 8(4): 295–318.

Newell, A.; Shaw, J. C.; and Simon, H. A. 1956. The Logic Theory Machine: A Complex Information Processing System. *IRE Transactions on Information Theory*, 2(3): 61–79.

Pan, J.; Chen, Y.; and Tang, H. 2024. Logic-LM++: Enhancing Large Language Models with Explicit Symbolic Reasoning. *arXiv preprint arXiv:2407.02514*.

Pan, L.; Albalak, A.; Wang, X.; and Wang, W. Y. 2023. Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning. *arXiv preprint arXiv:2305.12295*.

Patel-Schneider, P. F.; and Parsia, B. 2007. Pellet: A Practical OWL-DL Reasoner. In *Web Semantics: Science, Services and Agents on the World Wide Web*, volume 5, 51–53. Elsevier.

Polu, M.; and Sutskever, I. 2022. Formal Mathematics Statement Curriculum Learning. *arXiv preprint arXiv:2202.01344*.

Qin, L.; Sun, Z.; and Nie, J.-Y. 2025. VERUS-LM: Verifiable Reasoning in Large Language Models. *arXiv preprint arXiv:2501.14540*.

Schulz, S. 2019. E Prover 2.5. In *Proceedings of the 27th International Conference on Automated Deduction (CADE-27)*, volume 11716 of *Lecture Notes in Computer Science*, 370–387.

Sutcliffe, G. 2017. The TPTP Problem Library and Associated Infrastructure. *Journal of Automated Reasoning*, 59(4): 483–502.

Wu, Y. T.; Austin, J.; Polu, M.; Sutskever, I.; and Leike, J. 2023. Autoformalization with Large Language Models. *arXiv preprint arXiv:2306.04723*.

Xu, Y.; Chen, S.; Zhong, X.; Liu, J.; and He, X. 2026. Towards Multi-Clause Automated Deduction and Theorem Generation: Constructing and Applying Standard Contradictions. *Knowledge Based Systems*, 334: 114985.

Xu, Y.; Guo, H.; Chen, S.; and Liu, J. 2025. Δ_1: Automated Theorm Generator. *International Journal of Computational Intelligence Systems*, 18: 309. Δ_1 can be accessed via GitHub project: https://github.com/SWJTU-math/Automated_Theorem_Generator, along with further upgraded versions via https://github.com/SWJTU-math/Automated-Theorem-Generator-2 and https://github.com/lpy-2019/Automated-Theorem-Generator---Rectangle.

Yang, S.; Xu, B.; and Zhou, J. 2025. Neuro-symbolic reasoning in large language models: A survey and outlook. *Information Fusion*, 112: 102406.

Appendix A — Additional Examples

Table 1: Atomic predicate sets (L) for each domain corresponding to the examples in Table 2. Each L serves as input to Δ_1 for FTSC construction. Given $|L| = n$, Δ_1 deterministically generates $n+1$ dependency clauses $(D_1, \ldots, D_{n+1})$, forming a canonical unsatisfiable set and minimal contradictions $S \setminus \{D_i\} \vdash \neg D_i$.

Domain	Atomic Predicates (L)
Contract	$\{\mathit{Supplies}(s,p), \mathit{DeliversOnTime}(s,p), \mathit{ExclusiveSupplier}(s,p), \mathit{CanTerminate}(b,s), \mathit{HasPenalty}(s,p)\}$
Contract	$\{\mathit{Delivers}(s,p,t), \mathit{Insures}(s,p), \mathit{PaysPenalty}(s,p), \mathit{ForceMajeure}(t), \mathit{Renews}(c)\}$
Contract	$\{\mathit{Confidential}(d), \mathit{Shares}(p,d), \mathit{HasNDA}(p), \mathit{Audits}(p), \mathit{Discloses}(p,d)\}$
Healthcare	$\{\mathit{HoldsData}(h,p), \mathit{SharesData}(h,r,p), \mathit{HasConsent}(p), \mathit{Encrypts}(h,p), \mathit{Retains}(h,p,t)\}$
Healthcare	$\{\mathit{Diagnoses}(d,p), \mathit{Prescribes}(d,p,m), \mathit{Approves}(a,m), \mathit{Reports}(d,p), \mathit{Publishes}(a,m)\}$
Healthcare	$\{\mathit{Treats}(h,p), \mathit{UsesData}(h,p), \mathit{HasConsent}(p), \mathit{Charges}(h,p), \mathit{Transfers}(h,p,r)\}$
Finance	$\{\mathit{MaintainsReserve}(b,r), \mathit{IssuesLoan}(b,l), \mathit{ReportsRisk}(b,l), \mathit{DisclosesPublicly}(b,l), \mathit{PaysDividend}(b,a)\}$
Finance	$\{\mathit{Trades}(b,i), \mathit{Hedges}(b,i), \mathit{Reports}(b,i), \mathit{Discloses}(b,i), \mathit{Audited}(b)\}$
Regulatory	$\{\mathit{Complies}(o,r_1), \mathit{Complies}(o,r_2), \mathit{Reports}(o), \mathit{Exports}(o,p), \mathit{RetainsRecord}(o)\}$
Regulatory	$\{\mathit{Collects}(a,p), \mathit{Processes}(a,p), \mathit{Deletes}(a,p,t), \mathit{RequestsConsent}(a,p), \mathit{Transfers}(a,p,r)\}$

Table 2: Ten representative Δ_1 contradiction theorems with corresponding LLM interpretations and remediations. For each atomic predicate set L of size n, Δ_1 generates $n+1$ dependency clauses $(D_1, \ldots, D_{n+1})$ composing the FTSC. Each minimal entailment $S \setminus \{D_i\} \vdash \neg D_i$ reveals a domain-specific inconsistency, which the LLM then explains and resolves in natural language.

Domain	Δ_1 Theorem	LLM Explanation (Semantic Interpretation)	Remediation / Policy Fix
Contract	$S\setminus\{D_4\}\vdash\neg D_4$	Termination without cause contradicts exclusivity and penalty dependencies in the FTSC chain.	Restrict termination rights to "for cause" or require notice/compensation.
Contract	$S\setminus\{D_3\}\vdash\neg D_3$	Strict penalty enforcement becomes inconsistent when *ForceMajeure* negations are active.	Qualify penalties with force-majeure exceptions.
Contract	$S\setminus\{D_5\}\vdash\neg D_5$	Disclosure clause violates confidentiality unless NDA dependency ($\neg x_3$) is resolved.	Add NDA precondition to disclosure permissions.
Healthcare	$S\setminus\{D_3\}\vdash\neg D_3$	Data sharing without consent (*HasConsent*(p)) produces a minimal contradiction in the FTSC sequence.	Require explicit consent or anonymization before sharing.
Healthcare	$S\setminus\{D_5\}\vdash\neg D_5$	Publishing approvals without anonymization conflict with prior encryption and consent relations.	Anonymize or redact identifying data before publication.
Healthcare	$S\setminus\{D_6\}\vdash\neg D_6$	Transferring patient data contradicts consent, billing, and retention dependencies in the FTSC.	Add consent clause or lawful transfer basis.
Finance	$S\setminus\{D_5\}\vdash\neg D_5$	Dividend payment becomes inconsistent when reserves, loans, and disclosure predicates interact.	Restrict dividends until reserve adequacy verified.
Finance	$S\setminus\{D_6\}\vdash\neg D_6$	Global FTSC closure reveals overconstrained reserve and dividend dependencies.	Relax one upstream constraint or adjust reserve ratio threshold.
Regulatory	$S\setminus\{D_2\}\vdash\neg D_2$	Cross-jurisdictional obligations create contradictions under export conditions.	Harmonize export and reporting requirements; specify dominant regulation.
Regulatory	$S\setminus\{D_6\}\vdash\neg D_6$	Cross-border data transfer conflicts with deletion and consent dependencies in the FTSC closure.	Require renewed consent or deletion before data transfer.

Table 3: Ranked summary of Δ_1 + LLM reasoning outputs across domains. For each predicate set of size n, Δ_1 generates $n+1$ FTSC clauses $(D_1, \ldots, D_{n+1})$. Each contradiction theorem $S \setminus \{D_i\} \vdash \neg D_i$ is assigned a qualitative priority (High, Medium, Low) based on the LLM's interpretation of practical impact and remediation urgency.

Domain	Theorem	Priority	LLM-Suggested Remediation
Contract	$S\setminus\{D_4\}\vdash\neg D_4$	High	Restrict termination rights to "for cause" or require notice/compensation.
Contract	$S\setminus\{D_3\}\vdash\neg D_3$	Medium	Qualify penalty clauses with force-majeure exceptions.
Contract	$S\setminus\{D_5\}\vdash\neg D_5$	Low	Add NDA precondition to disclosure permissions.
Healthcare	$S\setminus\{D_3\}\vdash\neg D_3$	High	Require explicit patient consent or anonymization before data sharing.
Healthcare	$S\setminus\{D_5\}\vdash\neg D_5$	Medium	Anonymize or redact identifying data prior to publication.
Healthcare	$S\setminus\{D_6\}\vdash\neg D_6$	Low	Add lawful-basis clause permitting compliant data transfer.
Finance	$S\setminus\{D_5\}\vdash\neg D_5$	Medium	Delay dividend payments until capital adequacy confirmed.
Finance	$S\setminus\{D_6\}\vdash\neg D_6$	Low	Relax reserve ratio constraint or adjust dividend timing to restore consistency.
Regulatory	$S\setminus\{D_2\}\vdash\neg D_2$	Medium	Harmonize export and reporting obligations; specify primary regulation.
Regulatory	$S\setminus\{D_6\}\vdash\neg D_6$	Medium	Require renewed consent or deletion before cross-border data transfer.