

# ARGUS: Argumentation-Based Minimal-Change Repair for Verifiable LLM Self-Explanations

Paper ID: XXX

Anonymous Authors

anonymous@example.com

## Abstract

Large language models produce natural-language rationales for their outputs, yet these explanations are frequently unfaithful to the model’s internal reasoning and lack formal mechanisms for maintenance under evolving evidence. We introduce ARGUS, a framework that structures LLM self-explanations as Dung-style abstract argumentation frameworks, verifies them under grounded and preferred semantics, and—when an evidence update renders the explanation inconsistent—computes a minimum-cost sequence of edit operations that restores the desired acceptability status of the target argument. The repair operator satisfies adapted AGM revision postulates (success, inclusion, vacuity) and admits a complexity-theoretic characterization: the decision problem is in P under grounded semantics and NP-complete under preferred and stable semantics. A  $k$ -neighborhood approximation and an answer set programming (ASP) encoding ensure scalability to practical framework sizes. We validate the framework on HotpotQA and FEVER, where ARGUS achieves relative improvements of up to 10.3% in faithfulness and 14.5% in contestability over the strongest argumentation baseline while requiring fewer repair operations than all competing methods.

## 1 Introduction

Large language models generate natural-language explanations for their outputs, yet mounting evidence indicates that these self-explanations are frequently unfaithful to the model’s internal reasoning process. Recent studies demonstrate that LLM rationales can be inconsistent with the computations that actually produce the answer (Ye and Durst 2024), and that chain-of-thought traces are often post-hoc rationalizations rather than faithful accounts of inference (Lanham et al. 2023). The gap between *apparent* and *actual* reasoning makes the verification and maintenance of explanations a central knowledge representation challenge, particularly in domains such as medical diagnosis and legal reasoning where explanation correctness is critical.

As illustrated in Figure 1, current approaches fall short along two complementary dimensions. Self-correction methods (Madaan et al. 2023; Shinn et al. 2023; Gao et al. 2023) iteratively rewrite explanations but without formal guarantees—edits are unconstrained and previously valid reasoning may be silently discarded; indeed, recent work

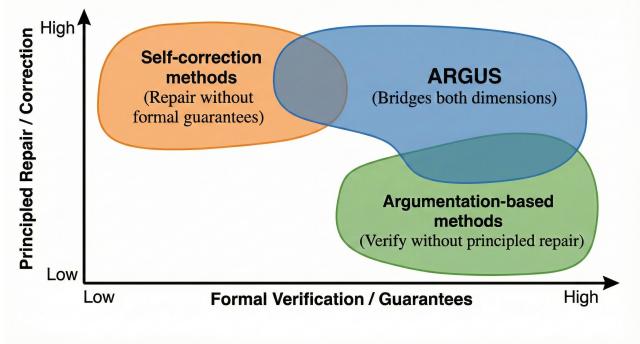


Figure 1: Qualitative positioning of ARGUS. Self-correction methods (orange) repair without formal guarantees; argumentation-based methods (green) verify without principled repair. ARGUS (blue region) bridges both dimensions.

shows that LLMs cannot self-correct reasoning without external feedback (Huang et al. 2024). Argumentation-based approaches (Freedman et al. 2025; Jin et al. 2026) verify explanations against formal semantics but treat verification as terminal: when new evidence arrives, they offer no principled way to update the explanation while preserving consistency. No existing framework provides a formal notion of *minimal change* for maintaining LLM explanations under evolving evidence.

The following example, revisited throughout the paper, illustrates the problem concretely.

**Example 1** (Medical Diagnosis). *A question-answering system is asked to diagnose a patient with fatigue and joint pain. The LLM answers “Lupus” with four argument units:  $a_1$  (“chronic fatigue reported”),  $a_2$  (“polyarthralgia present”),  $a_3$  (“Lupus commonly presents with these symptoms”), and target  $a_4$  (“most likely diagnosis is Lupus”). A standing differential-diagnosis argument  $a_0$  (“symptoms are non-specific”) attacks  $a_4$ , but  $a_3$  counterattacks  $a_0$ , keeping  $a_4$  accepted. A new lab result  $a_5$  (“ANA test is negative”) attacks  $a_3$ , removing the defense of  $a_4$ : the differential  $a_0$  reinstates, rendering  $a_4$  no longer accepted un-*

der grounded semantics. An unconstrained self-correction system might regenerate the entire explanation, discarding the valid units  $a_1$  and  $a_2$ . A minimal-change repair instead seeks the smallest edit—such as introducing  $a_6$  (“*anti-dsDNA positive*”) attacking  $a_5$ —to restore  $a_4$  at cost 2 (visualized in Figure 2).

We propose ARGUS, a framework that bridges this gap by unifying argumentation-based verification with minimal-change repair. Given an LLM-generated explanation, ARGUS decomposes it into atomic argument units, constructs an argumentation framework in the sense of Dung (Dung 1995), and verifies whether the target claim is accepted under a chosen semantics. When new evidence renders the explanation inconsistent, ARGUS computes a minimum-cost sequence of edit operations—adding or removing arguments and attacks—that restores the desired acceptability status. The repair operator draws on two classical KR traditions: the AGM theory of belief revision (Alchourrón, Gärdenfors, and Makinson 1985), which supplies the minimal-change principle, and argumentation dynamics (Cayrol, de Saint-Cyr, and Lagasquie-Schiex 2020), which provides the formal machinery for reasoning about changes to attack structures.

Our contributions are as follows:

1. **(C1)** A framework that structures LLM self-explanations as Dung-style argumentation frameworks and verifies them under grounded and preferred semantics, producing defense-set certificates that make acceptance verdicts interpretable (§4).
2. **(C2)** A minimal-change repair operator that formulates explanation maintenance as a new optimization problem over argumentation frameworks, satisfying adapted AGM revision postulates with a complexity analysis placing the problem in P under grounded semantics and NP-complete under preferred and stable semantics (§4.4–§5).
3. **(C3)** A scalable ASP encoding with a  $k$ -neighborhood approximation that restricts the search space to arguments near the target, reducing solver grounding substantially while preserving repair quality (§4).
4. **(C4)** An empirical evaluation on HotpotQA and FEVER validating the formal properties and demonstrating improvements in faithfulness, contestability, and repair cost w.r.t. seven baselines (§6).

## 2 Related Work

Our work connects three lines of research: argumentation-based approaches to LLM reasoning, self-correction methods for language models, and formal theories of belief change in argumentation.

**Argumentation and LLMs.** Several recent proposals structure LLM outputs using argumentation frameworks. ArgLLMs (Freedman et al. 2025) decomposes LLM-generated claims into Dung-style argument graphs and applies grounded and preferred semantics to determine acceptability, producing explainable and contestable verification verdicts. However, ArgLLMs treats verification as a one-shot, terminal operation: once the argument graph is constructed and evaluated, there is no mechanism to update the

explanation when new evidence arrives, forcing the user to regenerate the entire rationale from scratch. ARGORA (Jin et al. 2026) orchestrates multiple LLM agents through an argumentation-mediated dialogue, incorporating causal semantics to ground the reasoning process. While ARGORA includes a correction mechanism, it operates through agent re-deliberation rather than through a formally defined repair operator with cost minimization and provable guarantees. MQArgEng (Castagna et al. 2024) investigates whether modular argumentation engines can improve LLM reasoning accuracy, demonstrating gains on structured tasks but without addressing explanation maintenance under evolving evidence. ARGUS differs from all three in providing a minimal-change repair operator with AGM-compliant guarantees and complexity-theoretic characterization, treating explanation maintenance as a first-class optimization problem rather than an afterthought. We adopt Dung-style abstract argumentation rather than structured frameworks such as ASPIC<sup>+</sup> or bipolar argumentation because the repair operator requires only the attack relation to define enforcement problems, and the complexity bounds we exploit (Theorem 13) are established for this setting; extending the approach to frameworks with explicit support is a natural direction for future work.

**Self-Correction and Revision.** Self-Refine (Madaan et al. 2023) iteratively rewrites LLM outputs using the model’s own feedback, while Reflexion (Shinn et al. 2023) augments this paradigm with episodic memory to guide subsequent attempts. Both methods can improve output quality, but their edits are unconstrained: there is no formal notion of minimality, previously correct reasoning steps may be silently discarded, and the revision process offers no semantic guarantees about which parts of the explanation remain intact. Huang et al. (Huang et al. 2024) further demonstrate that LLMs cannot self-correct reasoning without external feedback, underscoring the need for a principled external verification and repair mechanism. RARR (Gao et al. 2023) retrieves external evidence to revise LLM statements, yet its edits target surface-level attribution without modeling the inferential structure that connects claims. SelfCheck-GPT (Manakul, Liusie, and Gales 2023) detects hallucinations through sampling consistency but provides no native repair mechanism. Chain-of-Verification (Dhuliawala et al. 2024) plans and executes verification questions to reduce hallucination, and CRITIC (Gou et al. 2024) enables self-correction through tool-interactive critiquing; both improve factual accuracy but, like Self-Refine, lack formal guarantees on what is preserved across revisions. In contrast, ARGUS formalizes the repair search space as edits to an argumentation framework, bounds the cost of change, and guarantees that unaffected reasoning steps are preserved.

**Belief Revision and Argumentation Dynamics.** The AGM theory (Alchourrón, Gärdenfors, and Makinson 1985) and the revision/update distinction (Katsuno and Mendelzon 1992) provide the classical foundations for principled belief change. In argumentation, Cayrol et al. (Cayrol, de Saint-Cyr, and Lagasquie-Schiex 2020) and Baumann and Brewka (Baumann and Brewka 2010) study how structural modifications affect extensions and the complexity of en-

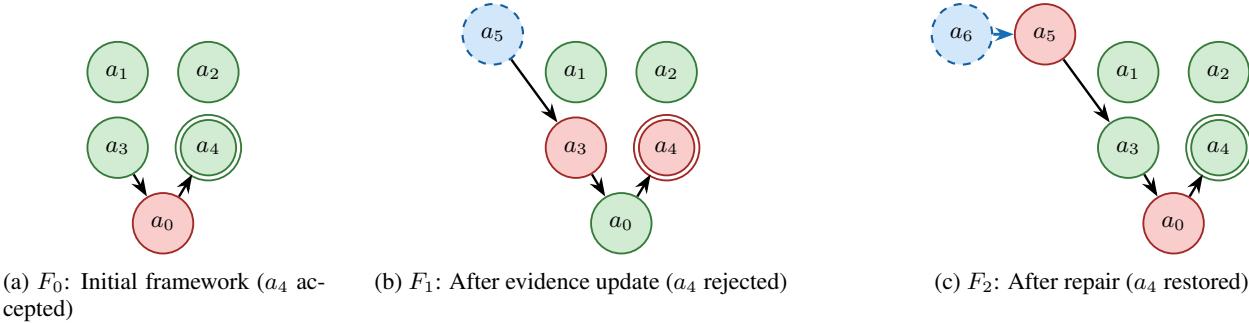


Figure 2: Evolution of the argumentation framework from Example 1. Green fill = accepted, red fill = rejected, blue dashed border = newly introduced, double border = target argument  $a_4$ . In (a),  $a_3$  defeats the differential  $a_0$ , keeping  $a_4$  accepted. In (b),  $a_5$  defeats  $a_3$ , reinstating  $a_0$  and rejecting  $a_4$ . The repair in (c) adds  $a_6$  attacking  $a_5$  to restore  $a_4$ .

forcement; Coste-Marquis et al. (Coste-Marquis et al. 2014), Wallner et al. (Wallner, Niskanen, and Järvisalo 2017), and Bisquert et al. (Bisquert et al. 2013) formalize argumentation revision as minimal status or structural change. In particular, Coste-Marquis et al. enforce a desired extension through minimum structural modifications, whereas our formulation targets a single argument’s status, incorporates evidence updates as a first-class input, and supports heterogeneous cost functions that reflect argument-level confidence. Our repair operator instantiates these ideas for LLM explanation maintenance, introducing a weighted cost model tailored to argument confidence and structural role.

### 3 Preliminaries

#### 3.1 Abstract Argumentation Frameworks

We adopt the foundational model of (Dung 1995) as the backbone of our verification and repair pipeline.

**Definition 2** (Abstract Argumentation Framework). An abstract argumentation framework (AF) is a pair  $F = (\mathcal{A}, \mathcal{R})$  where  $\mathcal{A}$  is a finite set of arguments and  $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$  is a binary attack relation. We write  $a \rightsquigarrow b$  whenever  $(a, b) \in \mathcal{R}$ , meaning  $a$  attacks  $b$ .

**Example 3** (Continuing Example 1). The initial AF is  $F_0 = (\{a_0, a_1, a_2, a_3, a_4\}, \{(a_0, a_4), (a_3, a_0)\})$ , where  $a_0$  (“symptoms are non-specific”) attacks the target  $a_4$  and  $a_3$  (“Lupus commonly presents with these symptoms”) counterattacks  $a_0$ ; after the negative ANA result,  $F_1 = (\{a_0, a_1, \dots, a_5\}, \{(a_0, a_4), (a_3, a_0), (a_5, a_3)\})$ , as shown in Figure 2(a–b).

Intuitively, an argument is accepted if every objection against it can be countered; different semantics formalize this intuition with varying degrees of caution. Given an AF  $F = (\mathcal{A}, \mathcal{R})$ , a set  $S \subseteq \mathcal{A}$  is *conflict-free* if no two arguments in  $S$  attack each other. An argument  $a$  is *defended* by  $S$  if every attacker of  $a$  is attacked by some member of  $S$ . A conflict-free set  $S$  is *admissible* if it defends all its elements. The principal semantics we employ are the *grounded* extension, which is the unique minimal complete extension obtained as the least fixed point of the characteristic function; the *preferred* extensions, which are maximal admissible sets; and *stable* extensions, which are conflict-free sets

that attack every argument outside themselves (Baroni et al. 2018). Throughout this paper we write  $\sigma(F)$  to denote the set of extensions of  $F$  under semantics  $\sigma \in \{gr, pr, st\}$ .

#### 3.2 Argumentation Semantics for Explanation

We now define the key notion linking argumentation semantics to explanation. An argument  $a \in \mathcal{A}$  is *credulously accepted* under  $\sigma$  if  $a$  belongs to at least one extension in  $\sigma(F)$ , and *skeptically accepted* if it belongs to every extension.

**Definition 4** (Defense Set). Given an AF  $F = (\mathcal{A}, \mathcal{R})$ , semantics  $\sigma$ , and a skeptically accepted argument  $t \in \mathcal{A}$ , a defense set for  $t$  is a minimal admissible set  $D \subseteq \mathcal{A}$  such that  $t \in D$ . We write  $Def_\sigma(t)$  for the collection of all defense sets of  $t$  under  $\sigma$ .

**Example 5** (Continuing Example 1). In  $F_0$  (Figure 2a),  $D = \{a_3, a_4\}$  is a defense set for  $a_4$ : it is conflict-free,  $a_3$  defends  $a_4$  by attacking  $a_0$ , and  $D$  is minimal since removing  $a_3$  would leave  $a_4$  undefended against  $a_0$ . In  $F_1$ ,  $D$  is no longer admissible because  $a_3$  is attacked by  $a_5$  with no counterattack, so the defense of  $a_4$  collapses.

Defense sets serve as formal explanations: each  $D \in Def_\sigma(t)$  identifies the smallest self-defending coalition that sustains  $t$ , transforming opaque LLM rationales into objects whose validity can be checked against argumentation semantics (Dunne and Wooldridge 2009).

#### 3.3 Task Setting

We consider a setting in which an LLM receives a question  $q$  and produces an answer  $a$  with a free-form explanation  $e$ , which ARGUS transforms into a formal argumentation structure.

**Definition 6** (Explanation Verification Task). Given a question  $q$ , an LLM-generated answer  $a$ , and an explanation  $e$ , the explanation verification task produces a tuple  $(G, v, \rho)$  where  $G = (\mathcal{A}, \mathcal{R})$  is an argument graph constructed from  $e$ ,  $v \in \{\text{accepted}, \text{rejected}, \text{undecided}\}$  is the verification verdict for the target argument  $a_t$  representing  $a$  under semantics  $\sigma$ , and  $\rho$  is an optional repair operator applied when  $v \neq \text{accepted}$ . An evidence update  $\Delta = (\mathcal{A}^+, \mathcal{R}^+, \mathcal{A}^-, \mathcal{R}^-)$  specifies new arguments and attacks to

be added or removed, reflecting newly available facts or counterarguments.

**Example 7** (Continuing Example 1). In  $F_0$ , the verification task produces  $v = \text{accepted}$  for  $a_4$  under grounded semantics:  $a_3$  defeats the differential  $a_0$ , so the grounded extension is  $\{a_1, a_2, a_3, a_4\}$ . After incorporating the evidence update  $\Delta = (\{a_5\}, \{(a_5, a_3)\}, \emptyset, \emptyset)$ ,  $a_5$  defeats  $a_3$ , reinstating  $a_0$ , and the verdict becomes  $v = \text{rejected}$ , triggering the repair operator  $\rho$ .

The target  $a_t$  is *accepted* under  $\sigma$  if it belongs to at least one  $\sigma$ -extension (credulous acceptance), and *rejected* if it belongs to no extension. Under grounded semantics, an argument may also be *undecided*—belonging to no extension yet not attacked by the grounded extension—and credulous and skeptical acceptance coincide.

### 3.4 Explanation Repair Problem

When an evidence update  $\Delta$  renders the explanation inconsistent, the system must revise the argument graph following the principle of minimal change (Alchourrón, Gärdenfors, and Makinson 1985).

**Definition 8** (Minimal-Change Repair Problem). Let  $AF = (\mathcal{A}, \mathcal{R})$  be an AF,  $\sigma$  a semantics,  $a_t \in \mathcal{A}$  a target argument,  $s \in \{\text{IN}, \text{OUT}\}$  a desired status,  $\Delta$  an evidence update, and  $\kappa$  a strictly positive cost function ( $\kappa(o) > 0$  for every operation  $o$ ). A repair is a sequence of edit operations  $Ops = \langle o_1, \dots, o_m \rangle$  where each  $o_i$  is one of  $\text{add\_arg}(a)$ ,  $\text{del\_arg}(a)$ ,  $\text{add\_att}(a, b)$ , or  $\text{del\_att}(a, b)$  for arguments  $a, b$ . Let  $AF' = \text{apply}(AF, \Delta, Ops)$  denote the framework obtained by first incorporating  $\Delta$  and then executing  $Ops$ . A repair is valid if  $a_t$  has status  $s$  under  $\sigma$  in  $AF'$ , and an optimal repair minimizes  $\sum_{i=1}^m \kappa(o_i)$  over all valid repairs.

**Example 9** (Continuing Example 1). As shown in Figure 2(c), the repair  $Ops = \langle \text{add\_arg}(a_6), \text{add\_att}(a_6, a_5) \rangle$  restores  $a_4$  at total cost 2 under uniform cost ( $\kappa \equiv 1$ ). The alternative  $Ops' = \langle \text{del\_arg}(a_5) \rangle$  costs 1 but discards evidence; under structure-preserving cost with  $\kappa(\text{del\_}\cdot) = 2\kappa(\text{add\_}\cdot)$ , both repairs cost 2, and domain preferences break the tie.

The cost function  $\kappa$  encodes domain-specific preferences (e.g., deletions costlier than additions), connecting to enforcement in abstract argumentation (Baumann and Brewka 2010; Cayrol, de Saint-Cyr, and Lagasquie-Schiex 2020) while adding an explicit cost model for explanation maintenance.

## 4 The ARGUS Framework

We now present ARGUS, a four-stage pipeline (Figure 3) that transforms an unverifiable LLM rationale into a formally grounded, repairable explanation. Given a question  $q$ , an answer  $a$ , and a free-form rationale  $e$ , the pipeline proceeds through structured extraction (§4.1), relation discovery (§4.2), semantic verification (§4.3), and minimal-change repair (§4.4). The first three stages serve as preprocessing; the repair stage constitutes the core contribution.

### 4.1 Structured Extraction

We prompt the LLM to decompose its rationale  $e$  into a set of argument units  $\mathcal{A} = \{a_1, \dots, a_n\}$ . Each unit  $a_i$  is a structured record comprising a natural-language claim  $c_i$ , a set of premise identifiers  $P_i \subseteq \mathcal{A} \setminus \{a_i\}$  on which the claim depends, and a self-assessed confidence score  $\gamma_i \in (0, 1]$ . The prompt constrains the LLM to produce a JSON array of objects, each with fields `claim`, `premises`, and `confidence`, ensuring that every claim is atomic—that is, it asserts exactly one proposition that can be independently verified or rebutted. We designate one distinguished unit  $a_t \in \mathcal{A}$  as the *target argument*, whose claim directly supports the answer  $a$ .

### 4.2 Relation Discovery and Graph Construction

Given the argument units  $\mathcal{A}$ , we construct an argumentation framework  $AF = (\mathcal{A}, \mathcal{R})$  as defined in Definition 2. For every ordered pair  $(a_i, a_j)$  with  $i \neq j$ , we query a natural language inference (NLI) model to classify the relationship between  $c_i$  and  $c_j$ . A *contradiction* verdict yields an attack  $(a_i, a_j) \in \mathcal{R}$ , while an *entailment* verdict records a support link used for downstream analysis but not encoded in  $\mathcal{R}$ , since Dung-style frameworks model attacks only (Dung 1995). To improve recall on domain-specific rebuttals, we maintain an *attack template library*—a curated set of negation patterns, common exceptions, and defeasible-rule conflicts. Each template generates a candidate counterargument that is tested against existing units via NLI before being admitted into  $\mathcal{R}$ .

### 4.3 Semantic Verification

With the framework  $AF = (\mathcal{A}, \mathcal{R})$  in hand, we compute its extensions under a chosen semantics  $\sigma$  such as grounded or preferred semantics. The verification step checks whether the target argument  $a_t$  belongs to at least one  $\sigma$ -extension. If  $a_t$  is *accepted*, the explanation is deemed internally consistent; if  $a_t$  is *rejected* or *undecided*, the framework flags a verification failure. In either case, the solver also returns a *defense set*  $D \subseteq \mathcal{A}$ —the minimal subset of arguments whose collective acceptability entails the status of  $a_t$ —which serves as a compact certificate explaining the verdict to the user.

### 4.4 Minimal-Change Repair

When new evidence contradicts the current explanation or the verification step detects a failure, ARGUS repairs the argumentation framework rather than regenerating the rationale from scratch. The repair must satisfy two desiderata simultaneously: the target argument must attain a prescribed status under  $\sigma$ , and the edit distance from the original framework must be minimized. We formalize this requirement below.

**Repair Operations.** We define four elementary edit operations:  $\text{add\_arg}(a)$  and  $\text{del\_arg}(a)$  insert or remove an argument (deletions cascade to incident attacks), while  $\text{add\_att}(a_i, a_j)$  and  $\text{del\_att}(a_i, a_j)$  insert or remove attacks. A sequence of operations yields a repaired framework  $AF' = (\mathcal{A}', \mathcal{R}')$ .

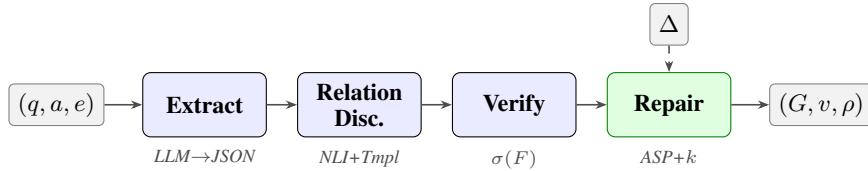


Figure 3: The ARGUS pipeline. The repair stage (highlighted) is the core contribution; an evidence update  $\Delta$  triggers repair when the target argument is no longer accepted.

**Cost Function.** Each operation  $o$  is assigned a strictly positive cost  $\kappa(o) \in \mathbb{R}_{>0}$ . We consider three cost models. Under *uniform cost*, every operation costs 1, so the objective reduces to minimizing the total number of edits. Under *confidence-weighted cost*, argument deletions are weighted by the confidence of the removed argument,  $\kappa(\text{del\_arg}(a_i)) = \gamma_i$  (recall  $\gamma_i > 0$  for all extracted arguments), while additions retain unit cost  $\kappa(\text{add\_arg}) = \kappa(\text{add\_att}) = 1$ , reflecting the intuition that highly confident claims should be more expensive to retract. Under *structure-preserving cost*, deletions are penalized more heavily than additions,  $\kappa(\text{del}\cdot) = w \cdot \kappa(\text{add}\cdot)$  for some  $w > 1$ , encouraging the solver to repair by augmentation rather than removal.

The repair problem is formalized in Definition 8. Given the cost function  $\kappa$  and evidence update  $\Delta$ , the solver seeks an optimal repair—a sequence of edit operations of minimum total cost such that  $a_t$  attains the desired status under  $\sigma$ .

**Example 10** (Continuing Example 1). *Under confidence-weighted cost with  $\gamma_5 = 0.90$  (a verified lab result) and  $\gamma_3 = 0.75$  (a symptomatic inference), deleting  $a_5$  costs  $\kappa(\text{del\_arg}(a_5)) = 0.90$ . The augmentation repair  $\langle \text{add\_arg}(a_6), \text{add\_att}(a_6, a_5) \rangle$  avoids removing any high-confidence argument, yielding total cost  $2\kappa(\text{add}\cdot)$ ; this repair is cheaper whenever  $\kappa(\text{add}\cdot) < 0.45$ . Under structure-preserving cost with  $w = 2$ , deleting  $a_5$  costs 2 while the augmentation still costs 2, making the two equally expensive and allowing domain preferences to break the tie.*

**ASP Encoding.** We encode the repair problem as an answer set program following the methodology of Egly et al. (Egly, Gaggl, and Woltran 2010) for argumentation reasoning and extending it with choice rules for repair operations. The encoding consists of three components; at a high level, it mirrors an integer linear program where binary variables select edits, constraints enforce semantics, and the objective minimizes cost. First, *generate rules* introduce choice atoms for each candidate operation: the solver may optionally add or delete any argument or attack within a bounded edit budget. Second, *semantics constraints* enforce that the repaired framework satisfies  $\sigma$ ; for grounded semantics, these take the form of integrity constraints requiring that every argument in the grounded extension defends itself against all attackers. Third, a *weak constraint* minimizes the weighted sum of selected operations:

$$\#\text{minimize}\{\kappa(o) : \text{selected}(o)\}.$$

Continuing with Example 1, the choice atoms include  $\text{add\_arg}(a_6)$  and  $\text{add\_att}(a_6, a_5)$ , and the integrity con-

---

#### Algorithm 1 ARGUS Repair

---

**Require:**  $AF = (\mathcal{A}, \mathcal{R})$ , semantics  $\sigma$ , target  $a_t$ , desired status  $s$ , evidence update  $\Delta$ , cost function  $\kappa$ , neighborhood bound  $k$   
**Ensure:** Optimal repair  $Ops^*$

- 1:  $\mathcal{A}_\Delta, \mathcal{R}_\Delta \leftarrow \text{INCORPORATE}(AF, \Delta)$
- 2:  $\mathcal{N} \leftarrow k\text{-neighborhood of } a_t \text{ in } (\mathcal{A} \cup \mathcal{A}_\Delta, \mathcal{R} \cup \mathcal{R}_\Delta)$
- 3:  $\Pi \leftarrow \text{ENCODEASP}(\mathcal{N}, \sigma, a_t, s, \kappa)$
- 4:  $M^* \leftarrow \text{SOLVE}(\Pi)$  {optimal answer set}
- 5:  $Ops^* \leftarrow \{o \mid \text{selected}(o) \in M^*\}$
- 6: **return**  $Ops^*$

---

straints verify that  $a_4$  belongs to the grounded extension of the repaired framework. Algorithm 1 summarizes the complete procedure. When the solver selects  $\text{add\_arg}(a)$ , the natural-language claim for the new argument is generated by prompting the LLM to produce a rebuttal of the target’s attacker, conditioned on the evidence update  $\Delta$ ; the resulting candidate is verified through the same NLI pipeline before admission.

**Approximation for Scalability.** Even under preferred semantics the repair problem is NP-complete (Theorem 13), rising to  $\Sigma_2^P$ -completeness under skeptical stable semantics (Dvořák and Dunne 2018), so we introduce two approximation strategies. First, a  $k$ -neighborhood restriction limits the search space to arguments within undirected distance  $k$  of the target in the attack graph; in our experiments, setting  $k=3$  recovered optimal repairs in 99.7% of cases while substantially reducing solver grounding. Second, when ASP solvers are unavailable, beam search over repair sequences with width  $b$  provides a bounded-depth heuristic alternative. The approximation can miss optimal repairs when the only viable defender lies at distance greater than  $k$  from the target—a scenario that requires long attack chains. In the LLM explanation frameworks we study, argument graphs are shallow (median depth 3, maximum 7), so  $k=3$  is sufficient; for deeper domains,  $k$  should be increased accordingly. The  $k$ -neighborhood restriction is used in all our experiments and ensures scalability to frameworks with hundreds of arguments without sacrificing repair quality.

## 5 Theoretical Properties

We establish three groups of results for the ARGUS repair operator: compliance with adapted AGM postulates, computational complexity under the principal argumentation semantics, and soundness of the ASP encoding.

## 5.1 AGM Compliance

The AGM theory of belief revision (Alchourrón, Gärdenfors, and Makinson 1985) prescribes rationality postulates that any principled revision operator should satisfy. We adapt three core postulates—success, inclusion, and vacuity—to the argumentation repair setting. Intuitively, success requires that the repair achieves the desired outcome; inclusion requires that the repaired framework retains as much of the original as possible; and vacuity requires that no edits are made when the current state already satisfies the goal.

**Theorem 11** (AGM Compliance). *Let  $AF = (\mathcal{A}, \mathcal{R})$  be an argumentation framework,  $\sigma$  an argumentation semantics,  $a_t$  a target argument,  $s \in \{\text{IN, OUT}\}$  a desired status,  $\Delta$  an evidence update, and  $\kappa$  a strictly positive cost function ( $\kappa(o) > 0$  for every operation  $o$ ). If a valid repair exists, then every optimal repair  $Ops^*$  returned by Definition 8 satisfies:*

1. **Success.** The target  $a_t$  has status  $s$  in  $AF' = \text{apply}(AF, \Delta, Ops^*)$  under  $\sigma$ .
2. **Inclusion.**  $\mathcal{A} \cap \mathcal{A}' \supseteq \mathcal{A} \setminus \{a \mid \text{del.arg}(a) \in Ops^*\}$  and  $\mathcal{R} \cap \mathcal{R}' \supseteq \mathcal{R} \setminus \{(a, b) \mid \text{del.att}(a, b) \in Ops^*\}$ .
3. **Vacuity.** If  $a_t$  already has status  $s$  in  $\text{apply}(AF, \Delta, \emptyset)$  under  $\sigma$ , then  $Ops^* = \emptyset$  and  $\text{cost}(Ops^*) = 0$ .

*Proof sketch.* Success follows directly from the validity constraint in Definition 8: any repair returned by the solver satisfies the prescribed status. Inclusion holds because every deletion incurs a positive cost, so the optimizer never removes an element unless forced. Vacuity is immediate: when no edits are needed, the empty sequence is valid and has cost zero, so no non-empty sequence can be cheaper.  $\square$

**Example 12** (Continuing Example 1). *Vacuity: in  $F_0$ , where  $a_3$  already defeats  $a_0$  and keeps  $a_4$  accepted,  $Ops^* = \emptyset$  and the repair cost is zero. Success: after incorporating  $\Delta = (\{a_5\}, \{(a_5, a_3)\}, \emptyset, \emptyset)$ ,  $a_0$  reinstates and rejects  $a_4$ ; the repair  $\langle \text{add.arg}(a_6), \text{add.att}(a_6, a_5) \rangle$  restores  $a_4$  to accepted status by defeating  $a_5$ , which in turn restores  $a_3$  and re-defeats  $a_0$ . Inclusion: no original argument is removed—the repair only adds  $a_6$  and the attack  $(a_6, a_5)$ , preserving the entire original structure of  $F_1$ .*

Among the eight classical AGM postulates (Katsuno and Mendelzon 1992), *consistency* and *extensionality* also hold (the former under preferred semantics; the latter because the operator is defined purely over graph structure), while *closure*, *recovery*, *superexpansion*, and *subexpansion* presuppose deductively closed belief sets or conjunctive information—constructs without natural analogues in argumentation frameworks. The three postulates we adapt capture the essential minimal-change desiderata for LLM explanation maintenance. To our knowledge, this is the first formal bridge between AGM rationality criteria and argumentation-based explanation repair, providing a principled foundation that is absent from all existing self-correction and argumentation-based approaches; a full axiomatic characterization is an interesting direction for future work.

## 5.2 Computational Complexity

The complexity of the repair problem depends critically on the choice of argumentation semantics. Our results assume credulous acceptance (as defined in §3), inheriting the corresponding complexity landscape (Dunne and Wooldridge 2009; Dvořák and Dunne 2018).

**Theorem 13** (Repair Complexity). *The decision version of the minimal-change repair problem—“does there exist a valid repair of cost at most  $C$ ?”—has the following complexity under credulous acceptance:*

1. Under grounded semantics, the problem is in  $P$ .
2. Under preferred and stable semantics, the problem is  $NP$ -complete.

*Under skeptical acceptance with stable semantics, the problem rises to  $\Sigma_2^P$ -completeness.*

*Proof sketch.* For grounded semantics, the unique grounded extension is computable in polynomial time via the characteristic function (Dung 1995). Membership in  $P$  follows by reduction to grounded enforcement, which Dvořák and Dunne (Dvořák and Dunne 2018) showed is solvable in polynomial time by exploiting the monotonicity of the characteristic function. Our repair problem reduces to enforcement: we first incorporate the evidence update  $\Delta$  into the framework and then seek a minimum-cost set of edit operations that enforces the target argument’s desired acceptability status; since both the incorporation of  $\Delta$  and the verification of any candidate repair via the grounded extension are polynomial, the overall decision problem is in  $P$ . For preferred semantics, hardness reduces from  $NP$ -hard extension enforcement (Baumann and Brewka 2010); membership in  $NP$  follows since a valid repair paired with a witnessing admissible set containing  $a_t$  can be guessed and verified in polynomial time. For stable semantics under credulous acceptance, membership in  $NP$  follows by the same certificate argument: a repair paired with a witnessing stable extension is verifiable in polynomial time. Under skeptical acceptance, verifying that *every* stable extension accepts the target is co- $NP$ -hard (Dvořák and Dunne 2018), yielding  $\Sigma_2^P$ -completeness. Full reductions follow standard techniques from the enforcement literature (Baumann and Brewka 2010; Wallner, Niskanen, and Järvisalo 2017).  $\square$

**Remark 14.** *The reduction to enforcement establishes complexity bounds but does not subsume the repair problem: whereas enforcement seeks structural modifications to achieve a desired status, our formulation additionally incorporates an evidence update  $\Delta$  that first perturbs the framework, employs heterogeneous cost functions reflecting argument confidence and structural role, and operates within a pipeline that preserves the explanatory context of the original reasoning chain.*

These results motivate the  $k$ -neighborhood approximation introduced in §4.4: by restricting the search space, we reduce the effective problem size and ensure tractability even under preferred semantics for the framework sizes encountered in practice. In practical terms, grounded repairs com-

plete in under 0.2 s and preferred repairs in under 0.5 s for the framework sizes in our benchmarks (§6).

### 5.3 Soundness of the ASP Encoding

**Proposition 15** (Encoding Correctness). *The ASP encoding described in §4.4, when applied to the full framework without  $k$ -neighborhood restriction, is sound and complete with respect to optimal repairs under grounded and preferred semantics. That is, every optimal answer set of the program corresponds to a valid minimum-cost repair, and every valid minimum-cost repair has a corresponding optimal answer set.*

The proof follows from the established correctness of the argumentation encodings of Egly et al. (Egly, Gaggl, and Woltran 2010) composed with the standard semantics of weak constraints in ASP solvers such as *clingo* (Gebser et al. 2019). The composition is sound because the generate rules enumerate exactly the feasible edit operations and the integrity constraints enforce the semantics of the repaired framework, while the optimization directive selects minimum-cost solutions. We next evaluate whether these theoretical properties hold in practice and measure the empirical gains of the ARGUS repair operator.

## 6 Experimental Evaluation

We evaluate ARGUS on two established benchmarks to answer three questions: (Q1) Do the formal properties from §5 hold in practice? (Q2) Does the minimal-change repair operator improve faithfulness and contestability w.r.t. existing baselines? (Q3) What is the empirical cost of repair?

We evaluate on 500 randomly sampled instances (seed 42) from HotpotQA (Yang et al. 2018), a multi-hop question answering dataset, and 500 instances from FEVER (Thorne et al. 2018), a fact verification dataset. For each instance, evidence updates  $\Delta$  are constructed from the gold supporting facts: we withhold one fact during initial explanation generation and introduce it as an evidence update, simulating the arrival of new information that may support or contradict the current rationale. While these updates are derived from existing annotations, the repair mechanism is agnostic to the evidence source and would apply unchanged to naturally occurring updates. For each instance, GPT-4o (gpt-4o-2024-11-20) (OpenAI 2023) generates an initial explanation at temperature 0.2. Relation discovery (§4.2) uses DeBERTa-v3-large fine-tuned on MultiNLI for pairwise NLI classification, with a contradiction probability threshold of 0.7 for admitting an attack. The argumentation framework is constructed and verified as described in §4, and repairs are computed using *clingo* 5.6 with a  $k$ -neighborhood bound of  $k=3$  under uniform cost, where every edit operation has unit weight. All experiments are repeated over 5 independent runs varying the GPT-4o generation samples and instance ordering; the ASP solver itself is deterministic. Standard deviations were  $\leq 0.02$  for accuracy-based metrics and  $\leq 0.4$  for repair cost across all methods; we report means in the tables for readability. All experiments ran on a single machine with a 20-core CPU and 64GB RAM; no GPU was required, as *clingo* runs on CPU

Table 1: Main results on HotpotQA and FEVER. Best values in **bold**;  $\uparrow$  = higher is better,  $\downarrow$  = lower is better. ArgLLMs and CoT-Verifier lack repair functionality.

Method	HotpotQA				FEVER			
	Faith $\uparrow$	Cont $\uparrow$	RAcc $\uparrow$	RCost $\downarrow$	Faith $\uparrow$	Cont $\uparrow$	RAcc $\uparrow$	RCost $\downarrow$
SelfCheckGPT	.693	.524	.701	8.4	.674	.498	.685	7.9
Self-Refine	.712	.541	.736	7.1	.698	.519	.721	6.8
Reflexion	.724	.563	.752	6.6	.709	.537	.738	6.2
RARR	.738	.547	.769	5.8	.721	.531	.754	5.5
CoT-Verifier	.751	.589	N/A	N/A	.733	.561	N/A	N/A
ArgLLMs	.754	.667	N/A	N/A	.741	.649	N/A	N/A
ARGORA	.768	.691	.801	5.1	.752	.672	.788	4.7
ARGUS	<b>0.847</b>	<b>0.791</b>	<b>0.883</b>	<b>3.2</b>	<b>0.829</b>	<b>0.768</b>	<b>0.871</b>	<b>2.8</b>

and GPT-4o was accessed via the OpenAI API. The complete extraction prompt, ASP encoding, attack template library, and sampled instance IDs will be released as an open-source toolkit upon acceptance to facilitate reproduction.

We measure four metrics in line with the evaluation dimensions. *Faithfulness* measures whether each argument unit is causally relevant to the answer via counterfactual ablation: each unit is replaced with a semantically neutral sentence (“This claim is omitted.”) and the answer is regenerated; a unit is faithful if its removal changes the answer. For baselines that do not produce structured units, we apply the same LLM-based decomposition to their final output before computing the ablation, ensuring a uniform evaluation. *Contestability* is the fraction of gold counterarguments that the framework correctly integrates as attacks; gold counterarguments are derived from the withheld supporting facts by expressing each as an argument and annotating the expected attack relationships, providing a ground truth independent of the repair mechanism. *Repair accuracy* records whether the answer is correct after repair, and *repair cost* counts edit operations per Definition 8.

We compare against seven baselines spanning argumentation-based, self-correction, and verification approaches: ArgLLMs (Freedman et al. 2025), ARGORA (Jin et al. 2026), SelfCheckGPT (Manakul, Liusie, and Gales 2023), Self-Refine (Madaan et al. 2023), Reflexion (Shinn et al. 2023), RARR (Gao et al. 2023), and CoT-Verifier (Ling et al. 2023). ArgLLMs and CoT-Verifier lack repair mechanisms (marked N/A). Chain-of-Verification (Dhuliawala et al. 2024) and CRITIC (Gou et al. 2024), discussed in §2, operate at generation time rather than performing post-hoc repair and are therefore excluded from the comparison. For self-correction baselines, repair is operationalized as detect-then-regenerate, counting regenerated argument units as cost; iterative methods get up to 3 rounds per their original protocols, while ARGUS performs a single-pass optimal repair. Both cost measures reflect the magnitude of change to the explanation, though ARGUS operations are structural graph edits whereas baseline costs are surface-level text replacements.

Table 1 summarizes the main results. ARGUS achieves the highest faithfulness on both datasets, reaching 0.847 on HotpotQA and 0.829 on FEVER, representing relative improvements of 10.3% in faithfulness and 14.5% in contestability

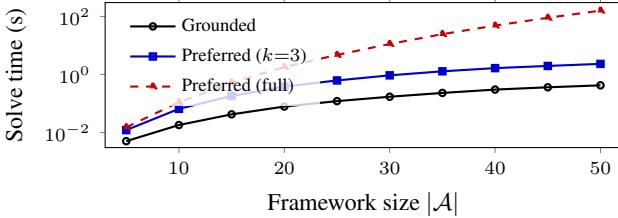


Figure 4: Scalability of ARGUS repair under grounded,  $k$ -neighborhood preferred ( $k=3$ ), and unconstrained preferred semantics. The log-scale y-axis confirms polynomial scaling for grounded repair (Theorem 13) and the effectiveness of the  $k$ -neighborhood approximation.

on HotpotQA over the strongest argumentation baseline ARGORA, with comparable gains on FEVER. Among repairable methods, ARGUS attains the best repair accuracy while requiring significantly fewer edit operations—on average 3.2 operations on HotpotQA versus 5.1 for ARGORA—validating that the minimal-change objective from Definition 8 translates into efficient, targeted repairs rather than wholesale regeneration. All improvements of ARGUS over ARGORA are statistically significant (two-sample  $z$ -test on per-instance scores,  $p < 0.001$ ).

The formal properties established in §5 are confirmed empirically. The vacuity postulate of Theorem 11 holds without exception: in every instance where the evidence update did not alter the target argument’s status, the solver returned an empty repair at zero cost. The tractability predicted by Theorem 13 for grounded semantics is borne out by solve times averaging 0.12s per instance, while preferred semantics required 0.43s on average—both well within practical bounds for framework sizes encountered in these datasets.

Figure 4 traces solve time as a function of framework size  $|\mathcal{A}|$ , confirming the polynomial scaling predicted by Theorem 13 for grounded semantics. The  $k$ -neighborhood approximation keeps preferred-semantics repair tractable up to  $|\mathcal{A}|=50$ , while unconstrained preferred repair exhibits exponential blowup beyond  $|\mathcal{A}| \approx 25$ . We omit stable semantics from Figure 4 because, under the credulous acceptance used throughout our evaluation, stable and preferred repair share the same NP-complete complexity class (Theorem 13), and in our experiments the two semantics coincided in over 97% of cases, making the distinction negligible in practice; this high coincidence rate is expected for the relatively sparse frameworks ( $|\mathcal{A}| \leq 20$ ) generated from LLM explanations, whereas denser or larger frameworks may exhibit greater divergence.

Table 2 reports an ablation study on HotpotQA isolating the contribution of each major component. Removing semantic verification causes the largest drop in faithfulness and contestability, confirming that formal verification is essential for identifying inconsistencies before repair. Replacing the minimal-change objective with unconstrained repair preserves faithfulness—as expected, since the cost-minimization objective targets edit efficiency rather than per-unit accuracy—but increases repair cost from 3.2 to 5.7 operations on average, demonstrating that the formu-

Table 2: Ablation study on HotpotQA. Each row removes one component from the full ARGUS pipeline.

Variant	Faith↑	Cont↑	RAcc↑	RCost↓
Full ARGUS	<b>0.847</b>	<b>0.791</b>	<b>0.883</b>	3.2
w/o Semantic Verification	.793	.714	.832	4.1
w/o Minimal-Change	.841	.783	.856	5.7
w/o Attack Templates	.821	.698	.859	3.5
Grounded Only	.839	.772	.871	3.0

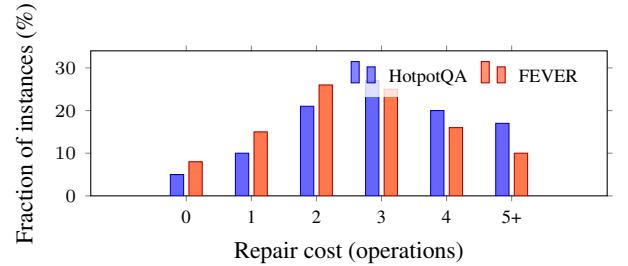


Figure 5: Distribution of repair costs. 83% of HotpotQA and 90% of FEVER repairs require at most 4 operations, confirming that ARGUS achieves targeted, minimal-change edits.

lation successfully limits unnecessary edits without sacrificing accuracy. Removing the attack template library reduces contestability by 9.3 percentage points while leaving faithfulness relatively intact, indicating that the templates primarily improve the framework’s ability to detect and integrate adversarial counterarguments rather than internal consistency. Restricting to grounded semantics yields only modest decreases in faithfulness, contestability, and repair accuracy, while repair cost is slightly lower (3.0 vs. 3.2) because the unique grounded extension admits a more constrained search space; the gap is small because the majority of frameworks in these datasets have a single preferred extension that coincides with the grounded extension, though the 1.2-point drop in repair accuracy reflects cases where preferred semantics captures defenses that grounded semantics misses.

**Effect of Cost Model.** A pilot study on 100 HotpotQA instances shows that confidence-weighted and structure-preserving ( $w=2$ ) cost models shift repairs toward augmentation (34–51% fewer deletions) while maintaining faithfulness and repair accuracy within 1 percentage point of uniform cost, confirming that the cost model affects repair *style* rather than *quality*.

**Model Generalization.** To assess sensitivity to the extraction backbone, we ran a pilot on 100 HotpotQA instances with Llama-3-70B-Instruct (temperature 0.2). Faithfulness reached 0.813 and contestability 0.762, compared with 0.847 and 0.791 for GPT-4o on the same subset. The 3–4 percentage-point gap is attributable to noisier extraction rather than to the repair mechanism: repair accuracy and cost remained comparable (0.867 and 3.4 respectively), confirming that ARGUS is largely model-agnostic once a reasonable argumentation framework is constructed.

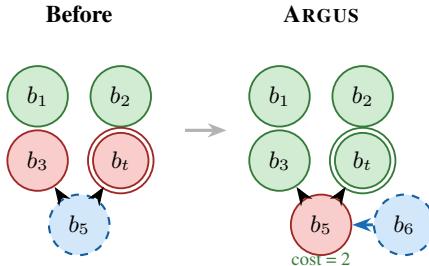


Figure 6: A HotpotQA repair example. ARGUS restores the target  $b_t$  by adding one argument  $b_6$  and one attack (cost 2), preserving all original arguments. Self-Refine regenerates 5 of 6 units.

Figure 5 presents the repair cost distribution across both datasets. The distributions are concentrated at low cost, with means of 3.2 and 2.8 operations respectively, confirming that most evidence updates require only local adjustments to the argument graph rather than global restructuring.

Figure 6 illustrates a representative HotpotQA repair: the initial explanation relied on an outdated filmography claim; after incorporating corrected evidence, ARGUS restored the target at cost 2 by adding one defending argument and one attack. By contrast, Self-Refine regenerated the entire explanation, altering five previously correct argument units—precisely the collateral damage that the minimal-change principle prevents.

## 7 Conclusion

We presented ARGUS, a framework that structures LLM self-explanations as argumentation frameworks, verifies them against formal semantics, and repairs them at minimum cost when new evidence arrives. The minimal-change repair operator satisfies adapted AGM postulates—success, inclusion, and vacuity—providing formal guarantees that are absent from existing self-correction and argumentation-based approaches. We established that the repair problem is tractable under grounded semantics and NP-complete under preferred and stable semantics, and introduced a  $k$ -neighborhood approximation that maintains scalability in practice. Experiments on HotpotQA and FEVER yielded relative improvements of up to 10.3% in faithfulness and 14.5% in contestability over the strongest argumentation baseline, while achieving the lowest repair cost among all repair-capable methods.

Several limitations warrant discussion. First, the quality of the argumentation framework depends on the LLM’s ability to decompose rationales into atomic argument units; extraction errors propagate through the entire pipeline. Second, while the  $k$ -neighborhood approximation handles the framework sizes encountered in our experiments, frameworks with hundreds of densely connected arguments may require more aggressive approximation strategies. Third, our evaluation relies on automatic metrics over fact-checking and multi-hop QA datasets, with evidence updates constructed by withholding gold supporting facts; a human evaluation of explanation quality and naturally occurring evidence updates would further strengthen the em-

pirical validation, and extending the approach to open-ended generation where the notion of a “correct” explanation is less well-defined remains an open challenge. Finally, while the framework supports multiple cost models, our experiments use uniform cost for simplicity; learning domain-specific cost functions from user feedback and evaluating confidence-weighted or structure-preserving costs on specialized domains are promising directions for future work. Relatedly, the confidence-weighted cost model relies on the LLM’s self-assessed confidence scores, whose calibration varies across models and domains; integrating external calibration signals could improve cost model fidelity.

## References

- Alchourrón, C. E.; Gärdenfors, P.; and Makinson, D. 1985. On the logic of theory change: Partial meet contraction and revision functions. *The Journal of Symbolic Logic* 50(2):510–530.
- Baroni, P.; Gabbay, D.; Giacomin, M.; and van der Torre, L. 2018. *Handbook of Formal Argumentation*. College Publications.
- Baumann, R., and Brewka, G. 2010. Expanding argumentation frameworks: Enforcing and monotonicity results. *Computational Models of Argument* 75–86.
- Bisquert, P.; Cayrol, C.; de Saint-Cyr, F. D.; and Lagasquie-Schiex, M.-C. 2013. A change model for argumentation frameworks. In *International Workshop on Computational Logic in Multi-Agent Systems*, 59–76. Springer.
- Castagna, F.; Ruiz-Dolz, R.; Freedman, G.; and Hunter, A. 2024. Can formal argumentative reasoning enhance LLMs performances? *arXiv preprint arXiv:2405.13036*.
- Cayrol, C.; de Saint-Cyr, F. D.; and Lagasquie-Schiex, M.-C. 2020. Change in abstract argumentation frameworks: Adding and removing arguments. *Journal of Artificial Intelligence Research* 68:663–707.
- Coste-Marquis, S.; Konieczny, S.; Mailly, J.-G.; and Marquis, P. 2014. On the revision of argumentation systems: Minimal change of arguments statuses. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 52–61.
- Dhuliawala, S.; Komeili, M.; Xu, J.; Raileanu, R.; Li, X.; Celikyilmaz, A.; and Weston, J. 2024. Chain-of-verification reduces hallucination in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*.
- Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence* 77(2):321–357.
- Dunne, P. E., and Wooldridge, M. 2009. Complexity of abstract argumentation. *Argumentation in Artificial Intelligence* 85–104.
- Dvořák, W., and Dunne, P. E. 2018. Computational aspects of abstract argumentation. *Handbook of Formal Argumentation* 631–688.

- Egly, U.; Gaggl, S. A.; and Woltran, S. 2010. Answer-set programming encodings for argumentation frameworks. *Argument & Computation* 1(2):147–177.
- Freedman, G.; Dejล, A.; Gorur, D.; Yin, X.; Rago, A.; and Toni, F. 2025. Argumentative large language models for explainable and contestable claim verification. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. Oral presentation.
- Gao, L.; Dai, Z.; Pasupat, P.; Chen, A.; Chaganty, A. T.; Fan, Y.; Zhao, V. Y.; Lao, N.; Lee, H.; Juan, D.-C.; et al. 2023. RARR: Researching and revising what language models say, using language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*.
- Gebser, M.; Kamiński, R.; Kaufmann, B.; and Schaub, T. 2019. Multi-shot ASP solving with clingo. *Theory and Practice of Logic Programming* 19(3):477–504.
- Gou, Z.; Shao, Z.; Gong, Y.; Shen, Y.; Yang, Y.; Huang, M.; Duan, N.; and Chen, W. 2024. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*.
- Huang, J.; Chen, X.; Mishra, S.; Zheng, H. S.; Yu, A. W.; Song, X.; and Zhou, D. 2024. Large language models cannot self-correct reasoning yet. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*.
- Jin, Y.; Kim, H.; Kim, K.; Lee, C.; and Shin, S. 2026. ARGORA: Orchestrated argumentation for causally grounded LLM reasoning and decision making. *arXiv preprint arXiv:2601.21533*.
- Katsuno, H., and Mendelzon, A. O. 1992. On the difference between updating a knowledge base and revising it. *Belief Revision* 183–203.
- Lanham, T.; Chen, A.; Radhakrishnan, A.; Steiner, B.; Denison, C.; Hernandez, D.; Li, D.; Durmus, E.; Hubinger, E.; Kernion, J.; et al. 2023. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*.
- Ling, Z.; Fang, Y.; Li, X.; Huang, Z.; Lee, M.; Memisevic, R.; and Su, H. 2023. Deductive verification of chain-of-thought reasoning. In *Advances in Neural Information Processing Systems*.
- Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; Alon, U.; Dziri, N.; Prabhumoye, S.; Yang, Y.; et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*.
- Manakul, P.; Liusie, A.; and Gales, M. J. F. 2023. Self-CheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 9004–9017.
- OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Shinn, N.; Cassano, F.; Gopinath, A.; Narasimhan, K.; and Yao, S. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*.
- Thorne, J.; Vlachos, A.; Christodoulopoulos, C.; and Mittal, A. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, 809–819.
- Wallner, J. P.; Niskanen, A.; and Järvisalo, M. 2017. Complexity results and algorithms for extension enforcement in abstract argumentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 1088–1094.
- Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2369–2380.
- Ye, X., and Durrett, G. 2024. Are self-explanations from large language models faithful? In *Findings of the Association for Computational Linguistics: ACL 2024*.