When Facts Are Not Enough: Structural Faithfulness in Trace-Grounded Reasoning

Anonymous ACL submission

Abstract

Large language models are increasingly used to generate natural-language explanations of structured computational processes. In deterministic settings, explanations may report correct dates and quantities while misrepresenting temporal ordering or dependency relations among events. This exposes a limitation of atomic verification, which checks individual facts but overlooks relational consistency. To address this limitation, we propose a structural perspective on explanation faithfulness by representing execution traces and explanations as relational graphs and evaluating their alignment. We construct a trace-grounded dataset with adversarial perturbations for controlled evaluation. Experimental results show that many explanations that pass atomic verification fail under structural evaluation, revealing a substantial gap between factual correctness and structural faithfulness. Furthermore, trace-grounded generation with explicit relational constraints substantially reduces this gap and improves structural alignment. Code and datasets will be released upon publication.

1 Introduction

Explanations in deterministic systems must report correct facts, such as dates and quantities, while also preserving how events are ordered and related. Material Requirements Planning (MRP) is a representative deterministic system (Ptak, 2011). It computes material availability through deterministic netting over dated demand and supply events. Although MRP is deterministic, enterprise systems often expose only final outcomes rather than the intermediate execution process, providing limited visibility into how shortages arise (Fransoo et al., 2010). As a result, planners frequently reconstruct explanations manually, with limited support for ensuring that these explanations faithfully reflect the underlying process (Madathil et al., 2025). In this work, we view Large Language Models (LLMs) not as replacements for deterministic planning engines, but as trace-to-text explanation models that organize trace-derived evidence into natural language. This raises a faithfulness question: whether generated explanations preserve temporal and dependency relations beyond individual facts.

Recent work has explored LLMs as interfaces for explaining complex systems (Zhao et al., 2024). While prior research emphasizes grounding explanations in system behavior (Jacovi and Goldberg, 2020), deterministic reasoning introduces a distinct faithfulness challenge: although explanations are generated probabilistically, they must remain not only factually correct but also faithful to the underlying execution process. Existing evaluations primarily verify *atomic* correctness, focusing on individual claims such as dates or quantities without assessing whether temporal and dependency relations among events are faithfully preserved. We refer to this problem as Trace-to-Text (TTX) explanation under structural faithfulness constraints. The challenge is not to recompute the deterministic outcome, but to generate and evaluate explanations that preserve the temporal and dependency structure of the underlying execution process.

To address the TTX problem, we propose a structural perspective on explanation faithfulness. We represent deterministic execution traces and generated explanations as relational graphs. This allows us to evaluate whether temporal and dependency relations are consistently preserved between deterministic traces and generated explanations. Building on this formulation, we construct a trace-grounded reasoning dataset derived from anonymized real-world MRP planning data. Because structurally inconsistent yet factually correct explanations are difficult to obtain naturally, we further construct adversarial examples that preserve atomic correctness while altering structural relations. Our contributions are summarized as follows:

- We show that fact-level correctness does not guarantee structurally faithful explanations in deterministic reasoning.
- We propose a graph-based formulation for evaluating whether explanations preserve temporal and dependency relations induced by deterministic execution traces.
- We introduce a trace-grounded reasoning dataset derived from anonymized real-world MRP planning data, including adversarial perturbations that decouple factual correctness from structural consistency.
- We show that structural verification identifies failures missed by atomic evaluation and improves explanation quality through verification-guided selection.

2 Related Work

Existing work on explanation faithfulness primarily focuses on factual grounding and consistency at the level of individual statements. In contrast, our work studies whether explanations preserve the relational structure induced by deterministic execution processes.

2.1 Faithfulness in NLP and LLM Explanations

Explainability has become a critical requirement in NLP systems, particularly in high-stakes decision-support settings (Gunning et al., 2019). Recent work has investigated whether natural-language explanations generated by LLMs faithfully reflect underlying reasoning processes. Prior studies have highlighted risks such as post hoc rationalization and hallucinated explanations, where generated rationales appear plausible but do not correspond to the actual reasoning process (Jacovi and Goldberg, 2020; Turpin et al., 2023; Atanasova et al., 2020; Paul et al., 2024). Many approaches address these issues through fact-level grounding, verifying whether individual explanation statements are supported by evidence. Chain-of-thought prompting and tool-augmented reasoning similarly aim to align explanations with intermediate reasoning steps or external tools (Wei et al., 2022; Schick et al., 2023; Gao et al., 2023). However, existing evaluation methods primarily focus on factual consistency without assessing whether explanations preserve temporal ordering or dependency relations among events in structured process settings.

2.2 Structured Explanation and Data-to-Text Generation

Research on data-to-text generation and structured reasoning focuses on producing natural-language descriptions from structured inputs such as tables, records, or intermediate computations (Reiter and Dale, 2000; Wiseman et al., 2017). A central concern in this line of work is faithfulness: whether generated text remains consistent with the underlying data representation. Prior work has therefore developed methods and metrics for reducing hallucinated or unsupported content in generated text (Liu et al., 2021; Zhao et al., 2023). For example, entity-centric and logic-form-based approaches improve factual grounding by verifying whether generated statements are supported by table records or symbolic forms.

However, existing approaches primarily evaluate independently verifiable facts rather than interactions among events. As a result, current evaluation formulations provide limited support for assessing whether generated explanations faithfully preserve temporal ordering and dependency relations induced by structured processes. In deterministic reasoning tasks, outcomes depend on sequences of interdependent events rather than isolated records (Ning et al., 2018), making relational faithfulness difficult to evaluate using existing data-to-text metrics.

2.3 Deterministic Planning Systems

Deterministic planning systems such as MRP provide a representative setting where outcomes are uniquely determined by rule-based computations (Orlicky, 1974; Jacobs et al., 2011). MRP computes projected inventory levels by combining dated demand and supply events through a netting-and-offset procedure. Despite its deterministic nature, modern systems often expose only final outcomes, providing limited visibility into intermediate computation processes (Elbahri et al., 2019). As a result, existing systems provide limited support for evaluating whether explanations faithfully preserve temporal ordering and dependency relations among events.

Our work does not aim to improve MRP algorithms, optimize planning decisions, or replace existing solvers. Instead, we study how to evaluate whether natural-language explanations faithfully preserve the relational structure induced by deterministic execution processes.

2.4 Tool-Augmented and Trace-Grounded Reasoning

Recent work incorporates external tools and intermediate computations to improve reasoning reliability and task performance (Schick et al., 2023; Gao et al., 2023). While these approaches improve correctness through executable programs or retrieved evidence, explanations may still fail to preserve how intermediate events are temporally ordered or causally related (Dziri et al., 2023). In contrast, our work evaluates whether explanations preserve the temporal and dependency structure induced by deterministic execution traces.

3 Deterministic Trace Formulation

This section formalizes deterministic planning execution as a trace-induced relational process. The resulting formulation provides the foundation for structural verification by exposing temporal and dependency relations among planning events.

To analyze structural faithfulness, we represent MRP execution as a deterministic trace. In deterministic planning systems, inventory outcomes are produced through temporally ordered demand and supply events. MRP computes projected inventory levels through a deterministic netting-and-offset procedure over a discrete planning horizon $\mathcal{H}$, where each $t \in \mathcal{H}$ represents a planning time step. Let $\mathcal{M}$ denote the set of materials. For each material $m \in \mathcal{M}$, let $\mathcal{D}^{(m)} = \{(t_i, d_i)\}_{i=1}^{N_D^{(m)}}$ denote the set of dated demand events and $\mathcal{P}^{(m)} = \{(t_j, p_j)\}_{j=1}^{N_P^{(m)}}$ denote the set of scheduled supply events. Here, d_i is the demand quantity occurring at time t_i, p_j is the supply quantity arriving at time t_j, and $N_D^{(m)} = |\mathcal{D}^{(m)}|$ and $N_P^{(m)} = |\mathcal{P}^{(m)}|$ represent the numbers of demand and supply events, respectively.

Given $\mathcal{D}^{(m)}$, $\mathcal{P}^{(m)}$, and initial stock $S_0^{(m)}$, projected inventory at time t is computed as:

$$S^{(m)}(t) = S_0^{(m)} + \sum_{t_j \le t} p_j - \sum_{t_i \le t} d_i. \quad (1)$$

The first shortage date is defined as:

$$t^{*(m)} = \min\{t \in \mathcal{H} \mid S^{(m)}(t) < 0\}. \quad (2)$$

To expose the intermediate planning process, we define a deterministic execution trace as:

$$\mathcal{T} := \{(t, S^{(m)}(t), D_t^{(m)}, P_t^{(m)}) \mid t \in \mathcal{H}\}, \quad (3)$$

where $D_t^{(m)} = \sum_{t_i \le t} d_i$ and $P_t^{(m)} = \sum_{t_j \le t} p_j$ denote cumulative demand and cumulative supply up to time t, respectively. The trace records the full inventory evolution process and uniquely determines the resulting planning outcome.

3.1 Opacity of Deterministic Netting

Without access to intermediate states, planners cannot identify which events caused inventory depletion or how inventory evolved over time, limiting diagnosis to generic corrective actions. Although access to the deterministic trace $\mathcal{T}$ improves visibility into inventory evolution, it does not guarantee that generated explanations faithfully preserve temporal ordering and dependency relations among events.

3.2 Formal Problem Definition

Let X denote the structured planning input, including demand, supply, inventory, and planning records. Given X and deterministic trace $\mathcal{T}$, the goal is to generate a natural-language explanation y grounded in trace-derived evidence. However, access to $\mathcal{T}$ does not ensure structural correctness: explanations may use factually correct information while misrepresenting temporal ordering or dependency relations between events. This motivates evaluation beyond isolated factual correctness toward relational consistency with the deterministic trace.

4 Structural Faithfulness

This section formalizes structural faithfulness as relational consistency between deterministic traces and generated explanations. The resulting framework enables graph-based verification of whether explanations preserve temporal and dependency relations induced by the execution process.

As discussed in Section 3, access to the deterministic trace $\mathcal{T}$ does not guarantee structural faithfulness. We therefore evaluate explanations based on relational consistency with the execution trace.

Atomic Faithfulness Existing approaches primarily evaluate atomic faithfulness by verifying whether explanations correspond to individual elements in the trace without assessing whether relational structure is preserved. We define *structural*

faithfulness as the degree to which an explanation preserves the relational structure induced by the trace.

Why Graphs? We use a graph representation because the target errors are relational rather than purely factual. While list or tuple representations encode individual events, they do not directly represent temporal or dependency relations between events. We therefore represent each event as a node and each temporal or dependency relation as an edge, enabling direct relation-level comparison between traces and explanations.

Relation Graph We model the deterministic trace $\mathcal{T}^{(m)}$ as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each node $e_i \in \mathcal{V}$ represents an event described by its type, timestamp, and quantity. The edge set $\mathcal{E}$ consists of temporal and dependency relations between events.

- **Temporal edges** $(e_i \prec e_j)$, representing chronological ordering where event e_i occurs before event e_j $(i < j)$.
- **Dependency edges** $(e_i \rightarrow e_j)$, representing that event e_i causally contributes to event e_j.

Explanation Graph We define an explanation graph $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}})$ as a relational graph extracted from a generated explanation y. Nodes in $\hat{\mathcal{V}}$ correspond to events referenced in y, and edges in $\hat{\mathcal{E}}$ encode temporal and dependency relations expressed in the explanation.

Structural Faithfulness Metric We evaluate structural faithfulness by comparing the explanation graph $\hat{\mathcal{G}}$ with the trace graph $\mathcal{G}$. Let E_r denote the set of relational edges in the trace graph $\mathcal{G}$, and let $\hat{E}_r$ denote the set of relational edges extracted from the explanation graph $\hat{\mathcal{G}}$ for a given explanation y. Each metric is computed at the explanation level and take a value in $[0, 1]$. We define structural precision and recall as:

$$\text{Precision} = \frac{|\hat{E}_r \cap E_r|}{|\hat{E}_r|}, \qquad \text{Recall} = \frac{|\hat{E}_r \cap E_r|}{|E_r|}. \tag{4}$$

We then compute the structural F1 score as:

$$\text{F1}_{\text{struct}} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{5}$$

Relations are considered matched when the corresponding relations in the two graphs connect the same events and share the same relation category (i.e., temporal or dependency). High Precision (e.g., Precision $= 1$) indicates that the explanation introduces no unsupported relations, while high Recall (e.g., Recall $= 1$) indicates that all trace relations are preserved in the explanation. If no relational edges are extracted, we set the corresponding metric value to 0.

Structural Violations We categorize structural violations into three types: temporal inversion, spurious causality, and aggregation distortion. These capture cases where explanations misorder events, introduce unsupported dependencies, or over-aggregate event chains in ways that cannot be detected through fact-level verification alone. Additional examples are provided in Appendix H.2.

5 Method

Figure 1 illustrates the overall framework. Given the structured planning input X, we first construct a deterministic trace $\mathcal{T}$, from which both trace-grounded evidence $\mathcal{E}$ and a relational graph $\mathcal{G}$ are derived (see Appendix C–H for implementation details). A language model then generates candidate explanations conditioned on $\mathcal{E}$, performing constrained explanatory organization: selecting salient trace events, ordering them coherently, and verbalizing their temporal and dependency relations without altering the underlying deterministic computation. Each explanation is converted into an explanation graph $\hat{\mathcal{G}}$, which is compared against $\mathcal{G}$ for structural verification. Detailed procedures are provided in Algorithm 1 and Algorithm 2 in Appendix D. The final explanation is selected based on verification signals. Formally, the overall process is:

$$X \rightarrow \mathcal{T} \rightarrow (\mathcal{E}, \mathcal{G}) \rightarrow y \rightarrow \hat{\mathcal{G}}$$

Trace and Relational Graph Construction Given structured inputs, we construct a deterministic execution trace that records inventory evolution over the planning horizon. From this trace, we derive dated events represented as $(\text{type}, \text{date}, \text{quantity})$, including demand, supply, and shortage events. We then automatically construct a relational graph whose nodes correspond to events and whose edges encode temporal ordering and dependency relations between events. Temporal edges are derived from event timestamps, while dependency edges are derived from deterministic

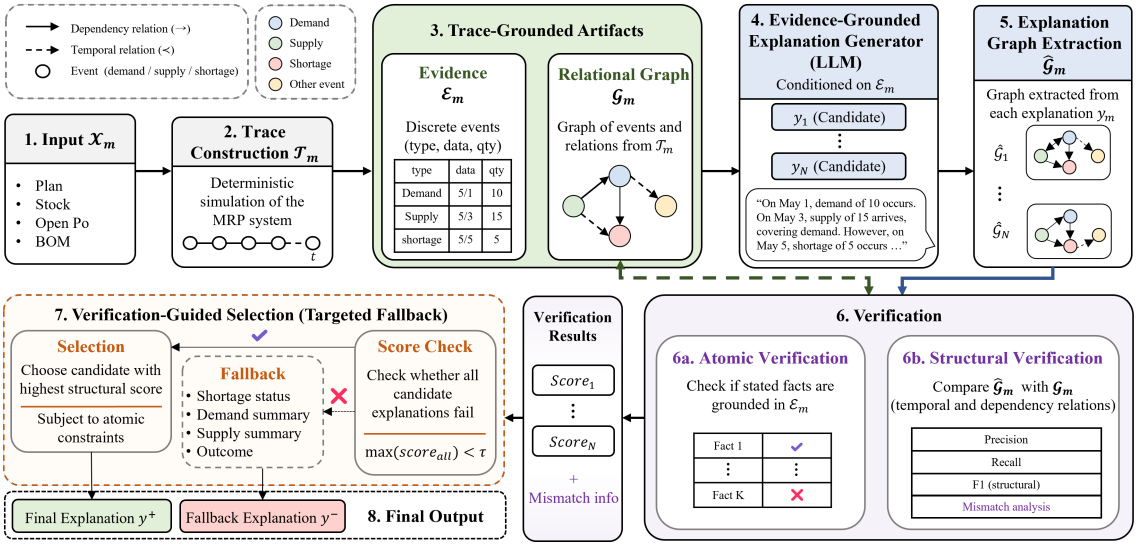

Figure 1: **Overview of the trace-grounded explanation framework.** Input data is transformed into a deterministic trace, from which evidence and a relational graph are derived. A language model generates N candidate explanations, which are verified, and the system selects the highest-scoring candidate or falls back to a trace-derived summary. Here, $m \in \mathcal{M}$ denotes a material instance, and N denotes the number of generated explanation candidates.

netting rules used by the execution engine. Thus, the trace graph is generated from the execution trace rather than manually annotated.

Trace-Grounded Explanation Generation Given trace-derived evidence $\mathcal{E}$, a language model generates candidate explanations conditioned on verified trace evidence rather than raw planning tables.

Explanation Graph Extraction For each explanation y, we extract an explanation graph $\hat{\mathcal{G}}$ from the generated natural-language explanation. Unlike the relational graph $\mathcal{G}$, which is deterministically constructed from the execution trace, the explanation graph represents the events and relations explicitly expressed in the explanation itself. Events are normalized into structured tuples $(\text{type}, \text{date}, \text{quantity})$, from which temporal and dependency edges are constructed. To reduce extraction noise, relation construction prioritizes explicit and verifiable relations. Further implementation details are provided in Appendix D.

Verification We evaluate explanations at two complementary levels. Atomic verification checks whether individual facts are grounded in $\mathcal{E}$. Structural verification compares $\hat{\mathcal{G}}$ with $\mathcal{G}$. Structural Precision, Recall, and $F1_{\text{struct}}$ are computed using Eqs. 4–5.

Verification-Guided Selection and Fallback Rather than selecting a single explanation, we use verification as a selection signal for faithful explanations. Multiple candidate explanations are converted into explanation graphs and verified against $\mathcal{G}$. Candidates failing atomic verification are discarded, and the remaining candidate with the highest structural F1 is selected. If all candidates fall below the structural threshold, a minimal trace-derived summary is returned (see Appendix E).

6 Trace-Grounded Reasoning Dataset

To enable controlled evaluation of structural faithfulness, we construct a trace-grounded reasoning dataset for deterministic planning. Unlike conventional explanation datasets that contain only input-output pairs, our dataset includes traces, evidence, relational graphs, explanations, and verification labels derived from the same deterministic process.

Original (Real-World) Dataset. The original dataset is derived from anonymized real-world MRP planning data. Each instance contains structured planning inputs, including Bill of Materials (BOM), stock, Open Purchase Orders (Open PO), and production plans, together with a deterministic trace, trace-grounded evidence, a relational graph, a natural-language explanation, and verification labels for atomic correctness and structural consistency. Because all components originate from the same deterministic process, the dataset enables controlled evaluation of whether explanations preserve relational structure.

Adversarial Perturbations. We augment the dataset with adversarial examples that alter relational structure while preserving atomic correctness. These perturbations include temporal inversion, spurious causality, and aggregation distortion. Table 1 shows representative examples.

Type	Example
Temporal inversion	The explanation states that the shortage emerged before the major demand event on 2026-01-10 reduced projected inventory.
Spurious causality	The delayed replenishment arrival is described as causing the shortage, despite the shortage being triggered by accumulated demand.
Aggregation distortion	Multiple dated demand events are summarized as a generic supply-demand imbalance without explicit dependencies.

Table 1: Examples of structural perturbations.

Dataset Statistics. The dataset contains 840 instances across multiple planning scenarios, including 120 original samples and 720 adversarial samples. Verification outcomes are distributed as PASS (13.2%), PARTIAL (43.9%), and FAIL (42.9%). The large proportion of PARTIAL and FAIL cases indicates that many explanations remain atomically correct while failing to preserve temporal and dependency relations, highlighting the gap between atomic correctness and structural faithfulness.

7 Experiments

7.1 Experimental Setup

We evaluate explanation quality in deterministic MRP planning scenarios using the full trace-grounded reasoning dataset. Our experiments evaluate relational preservation and verification-guided selection. All explanations are generated using GPT-4.1-mini and evaluated using atomic correctness, structural faithfulness, and human judgments. Atomic correctness is measured by verifying whether explanation facts are supported by trace-grounded evidence, while structural faithfulness is measured by edge-level matching between trace and explanation graphs.

7.2 Baselines

We compare methods with increasing levels of structural constraint, using controlled variants designed to isolate the effects of factual grounding, retrieval, chain-of-thought prompting, and explicit relational constraints; further baseline rationale is provided in Appendix G. (i) **Rule-Summary**, a deterministic trace-based summary without LLM generation; (ii) **No-Constraint**, which relies on unconstrained LLM generation; (iii) **Evidence-CoT**, which applies chain-of-thought reasoning over structured evidence; (iv) **Snippet-RAG**, which uses retrieval-based generation over trace snippets; (v) **Trace-Grounded**, which generates explanations from verified trace-derived evidence; and (vi) **Trace-Grounded+Rel**, which adds explicit temporal and dependency constraints.

7.3 Evaluation Metrics

Atomic Correctness. Atomic correctness measures whether individual factual elements, such as dates, quantities, and event mentions, are consistent with trace-derived evidence. This corresponds to fact-level verification, where each claim is independently matched against the execution trace.

Structural Faithfulness. Structural faithfulness evaluates whether explanations preserve the relational structure induced by the deterministic trace. Low Recall indicates missing relations present in the trace, while low Precision reflects spurious relations unsupported by the trace. Unlike atomic correctness, structural evaluation penalizes relational inconsistencies even when individual facts are correct.

Atomic–Structural Gap. To quantify the mismatch between factual correctness and structural consistency, we define the *atomic–structural gap* as the conditional probability that an explanation remains structurally inconsistent despite passing atomic verification:

$$\text{Gap} = P\left(F_{\text{struct}}(y) < \tau \mid A(y) = 1\right).$$

Here, $A(y) = 1$ indicates that explanation y belongs to the set of explanations that pass atomic verification, and $F_{\text{struct}}(y)$ denotes the structural F1 score defined in Eq. 5. The threshold $\tau \in [0, 1]$ represents the minimum structural consistency requirement; in our experiments, we set $\tau = 0.5$.

Human Evaluation. We conduct human evaluation on a subset of samples to assess perceived explanation quality. Annotators rate *causal coherence*, *trustworthiness*, and *readability* using a 5-point Likert scale; protocol details and dimension definitions are provided in Appendix H.1. Human evaluation provides a complementary perspective on explanation quality, which may not reflect structural faithfulness.

Method	RQ1: Atomic vs Structural Gap			RQ2: Structural Analysis		
	Atomic ↑	Structural F1 ↑	Gap ↓	Structural Precision ↑	Structural Recall ↑	Structural F1 ↑
Rule-Summary	0.000	0.409	0.000	**1.000**	0.263	0.409
No-Constraint	0.225	0.016	1.000	0.050	0.010	0.016
Evidence-CoT	0.392	0.040	0.936	0.133	0.024	0.040
Snippet-RAG	0.650	0.101	0.974	0.298	0.065	0.101
Trace-Grounded	**0.917**	0.486	0.246	**1.000**	0.340	0.486
Trace-Grounded+Rel	0.808	**0.633**	**0.031**	0.942	**0.503**	**0.633**

Table 2: **Unified evaluation of atomic correctness and structural faithfulness.** RQ1 shows that atomic correctness does not guarantee structural faithfulness, as evidenced by large gaps. RQ2 reveals that structural inconsistencies primarily arise from missing relations, reflected in low recall. Trace-Grounded improves structural alignment, and explicit relational constraints further enhance completeness.

7.4 RQ1: Atomic correctness does not imply structural faithfulness

We examine whether atomic correctness is sufficient to ensure structurally faithful explanations. As shown in Table 2, retrieval-based and unconstrained methods (e.g., Snippet-RAG) achieve relatively high atomic correctness (0.650), yet exhibit extremely low structural F1 (0.101), resulting in large atomic–structural gaps (0.974). A similar pattern is observed for Evidence-CoT, which improves atomic correctness (0.392) but fails to recover relational structure (F1 = 0.040, gap = 0.936).

These results indicate that preserving isolated factual elements from the deterministic trace does not guarantee preservation of the execution structure. Although explanations may correctly mention dates, quantities, and events, they often fail to preserve the temporal ordering and dependency relations induced by the deterministic netting process. As a result, explanations can remain atomically correct while structurally misrepresenting how the outcome was produced. This gap is further amplified in the adversarial evaluation setting, where perturbations are intentionally designed to preserve atomic correctness while altering relations. In contrast, Trace-Grounded+Rel substantially reduces the gap (0.031), showing that explicit relational constraints improve structural preservation.

7.5 RQ2: Where do structural inconsistencies arise?

While RQ1 establishes the atomic–structural gap, RQ2 analyzes where structural inconsistencies arise. Table 2 shows that retrieval-based and unconstrained methods exhibit very low structural recall, indicating that many trace relations are omitted. For example, Rule-Summary method achieves perfect Precision but low Recall, capturing only partial relations. This indicates that conservative summaries can be structurally safe but explanatorily incomplete: they avoid unsupported relations but often omit temporal and dependency links needed to explain the deterministic outcome. In other words, high precision alone does not imply that an explanation contains enough relational information to support faithful diagnosis. This result clarifies the role of LLM generation in our setting. The LLM does not recompute the planning outcome, but organizes trace-derived evidence into a more complete relational explanation, while structural verification constrains unsupported or misordered relations.

Overall, structural failures are driven largely by missing dependencies rather than only incorrect relations. Trace-Grounded improves structural alignment by constraining generation to trace-derived evidence, and relational constraints further improve completeness (Rec. = 0.503). These failures are consistent with the violations defined in Section 4, including temporal inversion, missing dependency relations, and aggregation distortion. Retrieval-based explanations often exhibit aggregation distortion by summarizing events without preserving dependency structure. This suggests that failures arise from omitted relations as well as hallucinated ones. Additional qualitative examples are provided in Appendix H.2.

7.6 RQ3: Can verification improve explanation quality?

Following the framework in Figure 1, we generate multiple candidate explanations for the same trace-derived evidence $\mathcal{E}$, convert each candidate into an explanation graph $\hat{\mathcal{G}}$, and rerank the candidates using structural verification scores against $\mathcal{G}$. We investigate whether structural verification can serve as a selection signal for faithful explanations.

Method	Struct. F1 ↑	Atomic ↑
TG	0.4662	0.9000
TG + Rerank	**0.5267**	**0.9500**
TG + Rel	0.6602	0.7917
TG + Rel + Rerank	**0.6835**	**0.9000**

Table 3: Effect of verification-based reranking. TG denotes Trace-Grounded generation, and Rel denotes explicit relational constraints.

As shown in Table 3, reranking improves structural F1 for both Trace-Grounded and Trace-Grounded+Rel settings. Structural F1 increases from 0.4662 to 0.5267 for the base setting and from 0.6602 to 0.6835 for the relationally constrained setting. Atomic correctness also improves, suggesting that structural verification helps select explanations that preserve both atomic correctness and relational consistency. These results indicate that structural verification is not only an evaluation metric, but also a practical selection signal for structurally faithful explanations.

7.7 Human Evaluation vs. Structural Faithfulness

Method	Causal ↑	Trust. ↑	Read. ↑
Rule-Summary	2.22	2.03	3.45
No-Constraint	4.20	4.07	4.12
Snippet-RAG	**4.40**	**4.17**	**4.58**
Trace-Grounded	4.07	3.65	4.35

Table 4: Human evaluation results averaged over 20 cases and three annotators.

We conduct human evaluation to examine whether human-perceived explanation quality aligns with structural faithfulness. Table 4 reports human scores for causal coherence, trustworthiness, and readability. LLM-based methods outperform rule-based summaries, indicating that natural-language explanations improve perceived interpretability. However, human preference does not necessarily align with structural faithfulness: Snippet-RAG receives the highest human scores despite weak structural alignment, as fluent explanations may contain unsupported or underspecified relations. In contrast, Trace-Grounded yields slightly lower human scores but stronger structural alignment. Thus, human evaluation should be complemented with structural verification.

8 Discussion

Our results establish a gap between atomic correctness and structural faithfulness: explanations may satisfy fact-level constraints while failing to preserve event relations. This shows that evaluation in deterministic settings should move beyond isolated facts and assess whether explanations preserve how events are ordered and connected. Structural verification also provides an actionable signal for improving generation. By selecting among multiple candidates, the verifier identifies explanations that better preserve relational structure, suggesting that structural faithfulness can be integrated into generation through verification-guided selection. Our evaluation relies on extracting relational structures from natural language explanations using a hybrid approach combining rule-based parsing and limited LLM-assisted inference. To reduce noise, explicit temporal relations are identified using deterministic rules, while LLM inference is restricted to implicit dependencies. Many violations considered in this work, such as temporal inversion and aggregation distortion, can be identified using rule-based patterns, although improving the robustness of relation extraction remains an important direction. Finally, while MRP serves as a representative case, the framework may extend to other deterministic processes where outcomes arise from structured interactions among events, such as algorithm execution, scheduling, and procedural workflows.

9 Conclusion

This paper studies explanation faithfulness in deterministic reasoning from a structural perspective. We show that atomic correctness does not guarantee structural faithfulness: explanations can report correct facts while misrepresenting temporal and dependency relations. By representing execution traces and explanations as relational graphs, we identify failures missed by fact-level evaluation and show that structural verification can also serve as a selection mechanism through reranking. These findings suggest that faithful explanations in deterministic settings require preserving relational structure, not only factual correctness. This perspective provides a more precise way to evaluate explanations in settings where outcomes depend on structured interactions among events. Future work includes improving relation extraction and extending structural faithfulness toward state-aware explanations.

Limitations

Our work has several limitations. First, structural faithfulness relies on the accuracy of extracting relational structures from natural-language explanations. Although we mitigate this issue using a hybrid approach combining rule-based parsing and limited LLM-assisted inference, extraction errors may still affect the reliability of structural evaluation. Second, our study focuses on explanation faithfulness rather than planning optimality. We assume that the underlying deterministic planning computation is correct, and do not evaluate the optimality or feasibility of the planning decisions themselves. Our focus is whether natural-language explanations faithfully preserve the structure of that computation. Third, our framework assumes access to deterministic execution traces. While this is appropriate for domains such as MRP and other structured processes, it may not directly generalize to settings where the underlying process is implicit, stochastic, or only partially observable. Fourth, our evaluation focuses on a specific set of relational properties, namely temporal ordering and dependency relations. Other aspects of explanation quality, such as abstraction, granularity, or pragmatic usefulness for decision-making, are not explicitly captured. Finally, while structural verification can improve explanation quality through reranking, our approach currently depends on generating multiple candidate explanations. This may introduce additional computational overhead, and integrating structural constraints more directly into generation remains an important direction for future work. Beyond event-level structural faithfulness, another promising direction is to interpret deterministic traces as state trajectories and extend verification toward state-aware explanations that preserve how system states evolve over time.

Ethics, Risks, and AI Assistance Disclosure

This work investigates the use of large language models as explanation interfaces for deterministic material planning systems, rather than as autonomous decision-making agents. The proposed framework does not modify MRP decisions, forecasts, or planning logic, but instead generates natural-language explanations grounded in explicit and verifiable execution traces. Potential risks may arise if generated explanations are interpreted as normative recommendations, or if the underlying trace information is incomplete or incorrectly specified. To mitigate these risks, the framework employs strict schema normalization, deterministic trace exposure, and evidence-constrained generation, restricting explanations to trace-grounded information. Generative AI assistants were used for language refinement and editing support during manuscript preparation. All technical designs, formal definitions, experimental settings, analyses, and interpretations were developed, verified, and validated by the authors. The authors take full responsibility for the content of this paper and its claims.

References

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2020. A diagnostic study of explainability techniques for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3256–3274, Online. Association for Computational Linguistics.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena Hwang, Soumya Sanyal, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. 2023. Faith and fate: Limits of transformers on compositionality. In *Advances in Neural Information Processing Systems*, volume 36.

Faisel Mohamed Elbahri, Omar Ismael Al-Sanjary, Musab A. M. Ali, Zakiya Ali Naif, Omar Ahmed Ibrahim, and M. N. Mohammed. 2019. Difference comparison of SAP, Oracle, and Microsoft solutions based on cloud ERP systems: A review. In *2019 IEEE 15th International Colloquium on Signal Processing & Its Applications (CSPA)*. IEEE.

Jan C. Fransoo, Toni Waefler, and John R. Wilson, editors. 2010. *Behavioral Operations in Planning and Scheduling*. Springer Science & Business Media, New York, NY.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pages 10764–10799. PMLR.

David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. 2019. XAI—explainable artificial intelligence. *Science Robotics*, 4(37):eaay7120.

F. Robert Jacobs, Richard B. Chase, and Nicholas J. Aquilano. 2011. *Manufacturing Planning and Control for Supply Chain Management*. McGraw-Hill, New York, NY.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205. Association for Computational Linguistics.

Tianyu Liu, Xin Zheng, Baobao Chang, and Zhifang Sui. 2021. Towards faithfulness in open domain table-to-text generation from an entity-centric view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13415–13423.

Abhilash Puthanveettil Madathil, Xichun Luo, Qi Liu, Charles Walker, Rajeshkumar Madarkar, and Yi Qin. 2025. A review of explainable artificial intelligence in smart manufacturing. *International Journal of Production Research*, pages 1–44.

Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. 2018. Joint reasoning for temporal and causal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2278–2288, Melbourne, Australia. Association for Computational Linguistics.

Joseph A. Orlicky. 1974. *Material Requirements Planning: The New Way of Life in Production and Inventory Management*. McGraw-Hill, New York, NY.

Debjit Paul, Robert West, Antoine Bosselut, and Boi Faltings. 2024. Making reasoning matter: Measuring and improving faithfulness of chain-of-thought reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15012–15032, Miami, Florida, USA. Association for Computational Linguistics.

Carol A. Ptak. 2011. *Orlicky's Material Requirements Planning*, 3 edition. McGraw-Hill, New York, NY.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Cambridge University Press, Cambridge.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, pages 68539–68551.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, volume 36.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 15(2):1–38.

Yilun Zhao, Zhenting Qi, Linyong Nan, Lorenzo Jaime Flores, and Dragomir Radev. 2023. LoFT: Enhancing faithfulness and diversity for table-to-text generation via logic form control. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 554–561, Dubrovnik, Croatia. Association for Computational Linguistics.

A Formal Definitions and Notation

This appendix defines the core objects used to formalize structural faithfulness in deterministic reasoning.

MRP Inputs. For a material m, the input instance is defined as $X_m = \{S_0, \mathcal{D}, \mathcal{P}, \mathcal{H}\}$, where S_0 is the initial on-hand stock, $\mathcal{D} = \{(t_i, d_i)\}$ is the set of dated demand events, $\mathcal{P} = \{(t_j, p_j)\}$ is the set of dated supply events, and $\mathcal{H}$ is a discrete planning horizon.

Deterministic Netting and Shortage. Projected inventory at time t is computed deterministically as:

$$S(t) = S_0 + \sum_{t_j \le t} p_j - \sum_{t_i \le t} d_i. \tag{6}$$

The first shortage date is defined as:

$$t^* = \min\{\, t \in \mathcal{H} : S(t) < 0 \,\}. \tag{7}$$

Deterministic Trace. We define a deterministic execution trace:

$$\mathcal{T}_m = \{(t, S(t), D_t, P_t) \mid t \in \mathcal{H}\}, \tag{8}$$

where $D_t = \sum_{t_i \le t} d_i$ and $P_t = \sum_{t_j \le t} p_j$. The trace uniquely determines the planning outcome.

Relational Structure. From $\mathcal{T}_m$, we derive a set of dated events $e = (\texttt{type}, \texttt{date}, \texttt{quantity})$, where $\texttt{type} \in \{\texttt{demand}, \texttt{supply}, \texttt{shortage}\}$. These events induce two types of relations: (i) temporal relations, which encode event ordering, and (ii) dependency relations, which encode contributory links between events. Together, these relations define the relational structure of the deterministic process.

Atomic Consistency. Let y_m be a natural-language explanation for material m, and let $I(y_m)$ denote the set of explicit informational atoms extracted from y_m (e.g., dates or quantities). We define atomic consistency as the requirement that these atoms be supported by trace-derived evidence:

$$I(y_m) \subseteq E_m, \tag{9}$$

where E_m is the set of admissible facts deterministically extracted from $\mathcal{T}_m$.

Structural Faithfulness. Beyond atomic consistency, an explanation must preserve the relational structure induced by $\mathcal{T}_m$. We therefore evaluate structural faithfulness by comparing the relations expressed in the explanation with those derived from the deterministic trace.

B Example of a Deterministic MRP Trace

Table 5 shows an excerpt of the deterministic MRP trace for *Material* 1. The trace records cumulative demand D_t, cumulative applied PO P_t, and the resulting projected inventory $S(t)$ over the planning horizon. In our implementation, when the inventory first becomes negative, the system registers the shortage date and immediately applies all remaining open PO quantity in the same step (see the SimulationAgent logic).

This example illustrates two key properties. First, the shortage event is uniquely determined by the cumulative effect of dated demand and supply events (Eq. 6–7). Second, the trace explicitly reveals the relational structure of the process, enabling verification of whether explanations correctly preserve event ordering and dependency relations.

C Deterministic Trace Construction

We construct a deterministic execution trace $\mathcal{T}$ from dated demand and supply events using the netting rule defined in Eq. 6.

At each time step $t \in \mathcal{H}$, cumulative demand and supply are updated, and projected inventory $S(t)$ is computed deterministically. The trace records $(t, S(t), D_t, P_t)$ for all $t \in \mathcal{H}$.

The first shortage date is defined as the earliest time t such that $S(t) < 0$, which is uniquely determined by the trace.

D Graph Construction and Structural Evaluation

This appendix provides additional implementation details for relational graph construction, explanation graph extraction, and structural evaluation.

D.1 Trace Graph Construction

From the deterministic trace $\mathcal{T}$, we derive a set of events represented as

$$e = (\texttt{type}, \texttt{date}, \texttt{quantity}),$$

where $\texttt{type} \in \{\texttt{demand}, \texttt{supply}, \texttt{shortage}\}$.

These events form nodes in a relational graph

$$\mathcal{G} = (V, E_r),$$

where edges encode temporal and dependency relations between events.

Temporal edges represent chronological ordering between events. A temporal edge $(v_i \prec v_j)$ is established when $t_i < t_j$.

Date	Demand$_t$	PO$_t$	S_{t-1}	$S(t)$ before PO	$S(t)$ after PO
2026-03-20	139.191	0	324.900	185.709	185.709
2026-03-21	14.143	0	185.709	171.566	171.566
2026-03-22	125.060	0	171.566	46.506	46.506
2026-03-23	139.203	276	46.506	-92.697	183.303
2026-03-24	14.143	0	183.303	169.160	169.160
2026-03-25	125.060	0	169.160	44.100	44.100
2026-03-26	139.263	0	44.100	-95.163	-95.163

Table 5: Excerpt of the deterministic netting trace for Material 1 around the first shortage event. Demand$_t$ denotes the daily demand increment; PO$_t$ denotes the PO quantity applied at that date. The first shortage is detected on 2026-03-23 when $S(t)$ becomes negative before PO application, after which remaining PO is applied immediately.

Algorithm 1 Relational Graph Construction

Require: Trace $\mathcal{T} = \{e_i\}_{i=1}^{n}$
Ensure: Trace graph $\mathcal{G} = (V, E_r)$
$V \leftarrow \{(type_i, date_i, qty_i) : e_i \in \mathcal{T}\}$
$E_{\text{temp}} \leftarrow \emptyset, \quad E_{\text{dep}} \leftarrow \emptyset$
for all $v_i, v_j \in V,\ i \neq j$ **do**
  if $date_i < date_j$ **then**
    $E_{\text{temp}} \leftarrow E_{\text{temp}} \cup \{v_i \prec v_j\}$
  end if
  if v_i causally contributes to v_j **then**
    $E_{\text{dep}} \leftarrow E_{\text{dep}} \cup \{v_i \rightarrow v_j\}$
  end if
end for
$E_r \leftarrow E_{\text{temp}} \cup E_{\text{dep}}$
return $\mathcal{G} = (V, E_r)$
=0

Dependency edges represent directed causal contributions under the deterministic netting process. An edge $(v_i \rightarrow v_j)$ is established when event v_i directly contributes to the occurrence of event v_j, such as demand contributing to shortage or supply mitigating inventory depletion.

Algorithm 1 summarizes the trace graph construction procedure.

D.2 Explanation Graph Extraction

Unlike the trace graph, which is deterministically constructed from the execution trace, the explanation graph is extracted from generated natural-language explanations. Given an explanation y, we identify event mentions and normalize them into structured tuples

$$(\texttt{type}, \texttt{date}, \texttt{quantity}).$$

Temporal relations are extracted from normalized date ordering, while dependency relations are identified using explicit causal expressions and lexical patterns. To reduce extraction noise, the system prioritizes explicit and verifiable relations, favoring precision over recall in ambiguous cases.

Algorithm 2 summarizes the explanation graph extraction procedure.

Algorithm 2 Explanation Graph Extraction

Require: Explanation $y = \{s_k\}_{k=1}^{m}$
Ensure: Explanation graph $\hat{\mathcal{G}} = (\hat{V}, \hat{E}_r)$
$\hat{V} \leftarrow \emptyset, \quad \hat{E}_{\text{temp}} \leftarrow \emptyset, \quad \hat{E}_{\text{dep}} \leftarrow \emptyset$
for all sentence $s_k \in y$ **do**
  $M_k \leftarrow \textsc{ExtractEventMentions}(s_k)$
  for all mention $m_i \in M_k$ **do**
    $\hat{v}_i \leftarrow \textsc{NormalizeEvent}(m_i)$
    $\hat{V} \leftarrow \hat{V} \cup \{\hat{v}_i\}$
  end for
end for
for all sentence $s_k \in y$ **do**
  $R_{\text{temp}} \leftarrow \textsc{FindTemporalCues}(s_k)$
  $R_{\text{dep}} \leftarrow \textsc{FindCausalCues}(s_k)$
  for all co-mentioned event pair $(\hat{v}_i, \hat{v}_j)$ **do**
    if R_{temp} implies $\hat{v}_i \prec \hat{v}_j$ **then**
      $\hat{E}_{\text{temp}} \leftarrow \hat{E}_{\text{temp}} \cup \{\hat{v}_i \prec \hat{v}_j\}$
    end if
    if R_{dep} implies $\hat{v}_i \rightarrow \hat{v}_j$ **then**
      $\hat{E}_{\text{dep}} \leftarrow \hat{E}_{\text{dep}} \cup \{\hat{v}_i \rightarrow \hat{v}_j\}$
    end if
  end for
end for
$\hat{E}_r \leftarrow \hat{E}_{\text{temp}} \cup \hat{E}_{\text{dep}}$
return $\hat{\mathcal{G}} = (\hat{V}, \hat{E}_r)$ =0

D.3 Structural Evaluation

We compare the explanation graph $\hat{\mathcal{G}}$ with the trace graph $\mathcal{G}$ using edge-level matching. A relation is considered matched only when:

1. the connected event tuples are consistent after normalization, and
2. the relation type (temporal or dependency) is identical.

Structural precision and recall are computed as:

$$\text{Precision} = \frac{|\hat{E}_r \cap E_r|}{|\hat{E}_r|}, \qquad \text{Recall} = \frac{|\hat{E}_r \cap E_r|}{|E_r|}.$$

The structural F1 score is:

$$F1_{\text{struct}} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

Low precision indicates spurious relations not supported by the trace, whereas low recall indicates missing relations present in the deterministic process. Aggregation distortion occurs when explanations merge multiple events into coarse summaries,

obscuring individual temporal or dependency relations and leading to structural mismatches despite factually correct statements.

E Verification-Guided Selection and Fallback Details

Fallback Procedure Given multiple candidate explanations, each is evaluated using atomic and structural verification. Candidates that satisfy atomic constraints are ranked based on structural alignment scores. The system selects the highest-scoring explanation as the final output.

If no candidate satisfies the acceptance criterion, the system returns a minimal trace-derived summary. This fallback mechanism avoids returning structurally unfaithful explanations and ensures that outputs remain grounded in the deterministic trace.

Failure Signals Verification identifies atomic and structural inconsistencies in candidate explanations. Atomic failures include ungrounded numerical values, ungrounded dates, and missing core evidence. Structural failures include temporal inversion, spurious causality, and aggregation distortion. These signals are used for analysis of failure patterns rather than for iterative regeneration.

Fallback Outcomes Table 6 summarizes the distribution of response types after applying the fallback mechanism. A substantial portion of cases fall back to minimal summaries. This indicates that many generated explanations fail to satisfy strict structural constraints, highlighting the difficulty of preserving relational structure in natural-language explanations.

Response Type	Count
selected_initial	35
selected_alternative	22
selected_alternative_2	4
fallback_summary	59

Table 6: Distribution of response types after verification-guided selection and fallback.

Example We illustrate the fallback process with a representative failure case.

1. *Initial (failed).* The generated explanation aggregates multiple events and omits explicit temporal relations, leading to aggregation distortion and structural misalignment.
2. *Alternative candidate.* The system considers an alternative explanation that explicitly references dated demand and supply events, improving temporal grounding and structural alignment. However, some dependency relations remain underspecified.
3. *Final outcome.* As no candidate satisfies the acceptance criterion, the system returns a minimal trace-derived summary containing only the key verified elements (e.g., stock depletion date, shortage date, and required quantity).

Concrete Example We illustrate the effect of the fallback mechanism with a representative example. An initial explanation may describe a shortage in generic terms (e.g., increased demand leading to shortage) without specifying explicit temporal relations between events, resulting in structural misalignment. An alternative explanation may include dated events, such as demand on 2024-01-10 preceding supply on 2024-01-12 and leading to a shortage on 2024-01-11, improving temporal grounding and structural consistency. However, if all candidate explanations fail to satisfy the acceptance criterion, the system returns a minimal trace-derived summary that includes only key verified elements, such as the stock depletion date, shortage date, and additional required quantity needed to resolve the shortage.

F Adversarial Construction

To evaluate structural faithfulness beyond fact-level correctness, we construct adversarial explanations by applying controlled perturbations to structurally faithful explanations.

Perturbation Types. We consider three types of structural perturbations: (i) temporal inversion, which reverses the ordering of events, (ii) spurious causality, which introduces unsupported dependency relations between events, and (iii) aggregation distortion, which merges multiple events into coarse summaries that obscure temporal or dependency relations.

Properties. These perturbations preserve surface fluency while altering the underlying relational structure, enabling controlled evaluation of structural inconsistencies that are not detectable through atomic verification alone.

G Baseline Rationale

Our baselines are designed as controlled prompt variants rather than as competing MRP solvers. Existing MRP and optimization tools address planning computation, whereas our experiments focus on explanation faithfulness given an available deterministic trace. We therefore compare explanation-generation strategies that vary the type of information and constraints provided to the language model, including unrestricted generation, structured evidence, retrieved trace snippets, trace-grounded evidence, and explicit relational constraints.

All LLM-based methods are required to produce structured JSON outputs and are evaluated using the same atomic and structural verification pipeline. The prompt differences are limited to the intended experimental factors. No-Constraint allows natural explanation generation from available evidence without structural constraints. Evidence-CoT adds step-by-step reasoning over structured evidence. Snippet-RAG restricts the model to retrieved trace snippets rather than the full trace. Trace-Grounded uses verified trace-derived evidence and prohibits unsupported facts. Trace-Grounded+Rel further requires explicit dated events, temporal relations, and dependency relations. Rule-Summary serves as a non-LLM deterministic summary baseline.

This design allows us to analyze which ingredients are necessary for preserving relational structure, rather than treating the baselines as deliberately weakened systems.

H Atomic Consistency Checking

We extract informational atoms (e.g., dates, quantities, and event references) from explanations using rule-based parsing. Each atom is matched against trace-derived evidence. Atoms that cannot be grounded are flagged as inconsistencies. This evaluation operates at the fact level and does not capture relational or structural errors.

H.1 Human Evaluation Protocol

We conduct human evaluation to assess perceived explanation quality. We sample 20 shortage cases from the dataset. For each case, explanations generated by different methods are anonymized, randomly ordered, and independently evaluated by three annotators with domain experience. Annotators are not shown the method identity of each explanation.

Each explanation is rated on a 5-point Likert scale from 1 (very poor) to 5 (excellent) along three dimensions: causal coherence, trustworthiness, and readability. Causal coherence measures whether the explanation logically connects the events leading to the shortage. Trustworthiness measures whether the explanation appears grounded in concrete data such as dates, quantities, or trace-derived evidence, rather than relying on vague speculation. Readability measures whether the explanation is clear, concise, and easy to understand. Annotators are instructed to evaluate each explanation independently and to judge explanation quality rather than explanation length.

H.2 Qualitative Example

We provide a representative example illustrating aggregation distortion, one of the structural failures observed in explanation generation.

Scenario. Inventory is depleted before supply replenishment, leading to a shortage.

Snippet-RAG. *"Initial stock is 1250, but total demand reaches 35795.029, causing inventory to be depleted on 2026-01-15 and shortage to occur on 2026-01-16. The cumulative order quantity is insufficient to meet demand, and the shortage is attributed to supply-demand imbalance. However, the retrieved snippets are not fully connected, making detailed reasoning uncertain."*

Observation. Although the explanation correctly reports key factual elements such as the stock depletion date, shortage date, and required quantity, it reduces the shortage to a generic supply-demand imbalance and explicitly acknowledges incomplete connections between retrieved snippets. As a result, it fails to preserve the event-level dependency structure of the underlying trace and exhibits *aggregation distortion*.

Comparison. In contrast, Trace-Grounded explanations explicitly link dated demand and supply events, preserving temporal and dependency relations. This example highlights that fluent and factually correct explanations may still fail to faithfully represent the underlying process.

Error Patterns. We further analyze common failure modes that persist even when explanations satisfy atomic correctness. Across examples, we observe temporal inversion, spurious causality, and aggregation distortion, despite correct numerical

Method	Atomic	Struct. F1
Trace-Grounded	0.950	0.486
Trace-Grounded (No Evidence)	0.000	0.065
Trace-Grounded+Rel	0.725	0.633

Table 7: Ablation study evaluating the impact of trace-grounded evidence and relational constraints. Removing evidence collapses both atomic correctness and structural faithfulness, while relational constraints improve structural alignment.

Model	Method	Atomic	Struct. F1
GPT-4.1-mini	TG	0.950	0.486
	TG+Rel.	0.725	0.633
GPT-4.1	TG	0.917	0.490
	TG+Rel.	0.842	0.660
GPT-4o	TG	0.950	0.487
	TG+Rel.	0.817	0.658

Table 8: Robustness across model families. TG denotes Trace-Grounded generation, and TG+Rel. denotes Trace-Grounded generation with explicit relational constraints. Relational constraints consistently improve structural faithfulness across all models.

and temporal facts. These patterns are consistent with the structural violations identified in RQ2 and highlight the limitation of fact-level evaluation.

H.3 Ablation Study

We analyze the contribution of key components in our framework. Table 7 summarizes the effects of trace-grounded evidence and relational constraints, while Table 8 reports robustness across model families.

Effect of Graph-Based Evaluation. When evaluation is restricted to atomic correctness, a large portion of structurally inconsistent explanations remain undetected. In contrast, graph-based structural evaluation identifies these failures, demonstrating that relational modeling is necessary for faithful evaluation.

Effect of Trace-Grounded Evidence. We remove trace-derived evidence from our model (Trace-Grounded (No Evidence)). As shown in Table 7, without evidence, both atomic correctness and structural faithfulness collapse (atomic: 0.95 $\rightarrow$ 0.00, structural F1: 0.49 $\rightarrow$ 0.07), showing that grounding in deterministic execution traces is essential for reliable explanation generation.

Effect of Relational Constraints. We compare the base model (Trace-Grounded) with its relationally constrained variant. Table 7 shows that relational constraints reduce atomic correctness (0.95 $\rightarrow$ 0.73), but substantially improve structural alignment (0.49 $\rightarrow$ 0.63), indicating that enforcing relational structure leads to more faithful explanations.

Effect of Model Choice. We evaluate our method across multiple models, including GPT-4.1-mini, GPT-4.1, and GPT-4o. As shown in Table 8, relationally constrained generation consistently improves structural faithfulness across all models, while slightly reducing atomic correctness. This consistency indicates that the observed improvements are not model-specific, but arise from the proposed structural framework.