QUANTIFYING CROSS-QUERY CONTRADICTIONS IN MULTI-QUERY LLM REASONING

Rohit Kumar Salla* Virginia Tech rohits25@vt.edu

Ramya Manasa Amancherla* Columbia University ra3439@columbia.edu

Manoj Saravanan* Virginia Tech manoj663@vt.edu

ABSTRACT

Large language models frequently produce mutually inconsistent answers when reasoning over multiple related queries. We study case-file logical consistency: maintaining a globally satisfiable belief state across interdependent queries. We introduce a benchmark of 390 multi-query reasoning instances with entailment/contradiction/unknown labels and propose set-level metrics including Case Satisfiability Rate, Contradiction Density and Revision Cost. Our solver-augmented approach extracts commitments, verifies global satisfiability and performs counterexample-guided repair. Across four reasoning domains, our method substantially reduces cross-query contradictions (SetCons: 0.56 to 0.94) while preserving per-query accuracy, demonstrating that global coherence is critical for robust multi-query reasoning.

Benchmark

1 INTRODUCTION

Large language models (LLMs) have recently shown strong performance on many reasoning-heavy tasks, especially when equipped with reasoning-oriented prompting such as chain-of-thought (CoT) (Wei et al., 2022) and improved decoding strategies like self-consistency (Wang et al., 2022). Despite these advances, their *logical reliability* remains brittle: models may answer a single query correctly yet produce mutually incompatible answers when asked several related questions that are all derived from the same underlying premises. This failure mode is increasingly documented as *self-contradictory reasoning* and can be measured and detected, but it persists even in otherwise capable models (Liu et al., 2024). In real deployments (e.g., clinical workups, legal case analysis, scientific hypothesis checking), users rarely ask one isolated question they iterate, refine and branch, turning reasoning into a *bundle* of interdependent queries whose answers must remain globally coherent.

From single-query accuracy to case-file consistency. Most existing logical reasoning evaluations are *single-turn*: a model is given one problem instance and must output one answer. Recent benchmarks provide more rigorous logical stress-tests (e.g., complex deductive reasoning and SAT-derived constraint puzzles) (Chen et al., 2025; Wei et al., 2025), but they still largely assess correctness *per query*. Separately, multi-turn evaluation suites show that performance often degrades in interactive settings due to error propagation and long-range dependencies (Kwan et al., 2024; Li et al., 2025). However, these multi-turn benchmarks typically do not isolate a *logical satisfiability constraint* that ties the turns together into a single, consistent belief state. As a result, we currently lack a principled framework to measure and improve *global logical coherence* across a set of related questions.

Why solver augmentation is not enough (yet). A promising direction is to couple LLMs with symbolic solvers, where the LLM acts as a translator from natural language to a formal representation and the solver ensures sound inference (Pan et al., 2023). Yet, solver-augmented reasoning remains sensitive to practical design choices: even the choice of solver and its input language can

*Equal contribution.

cause large swings in executability and end accuracy (Lam et al., 2024). Moreover, recent work on improving logical consistency has focused on aligning model behavior with logical constraints or preferences (Liu et al.) and logic-in-context prompting improves certain logical tasks (Liu et al., 2025), but these efforts still do not directly target the *set-level* failure mode where answers across multiple queries must jointly remain satisfiable.

Our goal. We study *case-file logical consistency*: given a shared premise set, the model must answer a bundle of interdependent queries while maintaining a globally satisfiable commitment state. The core challenge is not merely avoiding local mistakes but preventing (and repairing) contradictions that emerge across turns as the user probes implications, counterfactuals and missing facts.

Contributions.

- **Problem formulation.** We formalize *case-file logical consistency* as set-level reasoning under shared premises, where answers induce commitments that must remain jointly satisfiable.
- **Benchmark + metrics.** We introduce a multi-query benchmark with entailment/contradiction/unknown labels and propose set-level metrics that quantify global satisfiability and contradiction rate beyond per-query accuracy.
- **Lightweight consistency repair.** We present a solver-augmented decoding approach that tracks commitments and performs minimal repairs when contradictions are detected, improving global consistency with low overhead.

2 Problem Setup: Case-File Logical Consistency

We study *case-file logical consistency*: answering a **bundle** of interdependent queries under shared premises while maintaining a **single, globally satisfiable** belief state. Unlike standard logical reasoning benchmarks that score queries independently (Chen et al., 2025; Wei et al., 2025), our setting targets the common failure mode where LLMs give plausible single answers yet contradict themselves across related questions (Liu et al., 2024).

Case files and query bundles. A **case file** is a set of natural-language premises $\mathcal{P} = \{p_1, \ldots, p_m\}$ (facts, rules and global constraints). A **query bundle** is $\mathcal{Q} = \{q_1, \ldots, q_n\}$, where queries share entities/variables and induce cross-constraints. We evaluate both a **set** (offline) setting, $f(\mathcal{P}, \mathcal{Q}) \rightarrow \{a_i\}_{i=1}^n$ and a **sequential** (online) setting with ordered queries $\langle q_1, \ldots, q_n \rangle$, where $a_t = f(\mathcal{P}, q_t, \{(q_i, a_i)\}_{i<t})$.

Outputs. For each query q_i, the model predicts $a_i \in \{\text{ENTAILED}, \text{CONTRADICTED}, \text{UNKNOWN}\}$. UNKNOWN captures underdetermined cases where the premises neither prove nor refute the queried statement, discouraging hallucinated commitments.

Commitments and belief state. To evaluate global consistency, we map each answer to symbolic commitments in a constraint language $\mathcal{L}$ (e.g., propositional logic or a decidable SMT fragment). Let $\phi(\mathcal{P}) \in \mathcal{L}$ be the formalized case file and $\psi(q_i, a_i) \in \mathcal{L}$ be commitments extracted from the model output via f_{ext} (e.g., canonical CNF/SMT-LIB), following solver-augmented reasoning pipelines (Pan et al., 2023). We define the accumulated belief state:

$$\mathcal{B}_t \;=\; \phi(\mathcal{P}) \wedge \bigwedge_{i=1}^{t} \psi(q_i, a_i). \tag{1}$$

The bundle is **consistent** up to t iff $\mathcal{B}_t$ is satisfiable. Contradictions correspond to transitions where $\mathcal{B}_{t-1}$ is satisfiable but $\mathcal{B}_t$ is not.

Minimal repair. When inconsistency occurs, we quantify how many commitments must be revised to restore satisfiability (revision cost), connecting to classical belief revision objectives (Alchourrón et al., 1985; Hansson, 1999). This yields a set-level measure of how "stable" the model's commitments are across a case file.

3 METHOD

We describe (i) strong prompting baselines for multi-query reasoning and (ii) our solver-augmented approach that explicitly maintains a satisfiable belief state. Our method is inspired by two complementary lines of work: reasoning-oriented prompting such as chain-of-thought (Wei et al., 2022), self-consistency (Wang et al., 2022) and neuro-symbolic pipelines that translate natural language into logic and verify with external solvers (Pan et al., 2023). We additionally incorporate counterexample/unsat-core feedback a standard mechanism in formal methods to localize conflicts and guide minimal repairs, and we report tool sensitivity in light of evidence that solver choice can materially impact outcomes (Lam et al., 2024).

3.1 BASELINES

Direct decoding (No-CoT). The simplest baseline answers each query independently given the case file:

$$\hat{a}_i = f_\theta(\mathcal{P}, q_i). \tag{2}$$

This isolates the effect of cross-query interaction: any inconsistency arises from the model's latent contradictions rather than explicit state tracking.

Chain-of-thought (CoT). We prompt the model to produce intermediate reasoning steps before the final label, following Wei et al. (2022). We use a fixed format: `Reasoning: ... Answer: Entailed/Contradicted/Unknown`. CoT typically improves single-query reasoning but does not guarantee set-level consistency.

Self-consistency. We apply self-consistency decoding (Wang et al., 2022) by sampling K reasoning traces for each query and taking a majority vote over the final labels:

$$\hat{a}_i = \textsc{Mode}\left(\{\hat{a}_i^{(k)}\}_{k=1}^{K}\right). \tag{3}$$

While effective for per-query accuracy, self-consistency can still yield a set of answers that is globally inconsistent because voting is performed independently per query.

Sequential conditioning (History baseline). In the sequential setting, we optionally condition on prior Q&A pairs:

$$\hat{a}_t = f_\theta\big(\mathcal{P}, q_t, \{(q_i, \hat{a}_i)\}_{i=1}^{t-1}\big). \tag{4}$$

This captures common chatbot behavior (using previous answers as context) but may *amplify* early errors through propagation (Kwan et al., 2024).

3.2 PROPOSED APPROACH: BELIEF-STATE TRACKING WITH SOLVER-CHECKED COMMITMENTS

Our goal is to enforce *case-file logical consistency* by ensuring that the commitments induced by the model's answers remain jointly satisfiable. We maintain an explicit belief state $\mathcal{B}_t$ (Section 2) and use a solver to check satisfiability after each answer. When a contradiction is detected, we invoke a local repair procedure guided by the solver's unsat core.

3.2.1 STEP 1: COMMITMENT EXTRACTION

Given case file $\mathcal{P}$ and query q_t, the model produces a raw output $\hat{y}_t$ which includes the predicted label $\hat{a}_t$. We then extract a set of symbolic commitments $\psi_t = \psi(q_t, \hat{a}_t) \in \mathcal{L}$ using a structured extractor f_{ext}:

$$\psi_t \;=\; f_{\text{ext}}(\mathcal{P}, q_t, \hat{y}_t). \tag{5}$$

Following solver-augmented reasoning paradigms (Pan et al., 2023), we implement f_{ext} by prompting the LLM to emit a canonical logical form (e.g., CNF clauses for SATBench-style domains (Wei et al., 2025) or SMT-LIB for temporal constraints) then normalizing entities and polarity. We keep the extraction language lightweight (no higher-order constructs) to preserve solver reliability and reduce parsing failures.

Extraction format. Each commitment is represented as a conjunction of typed atoms and (optional) simple constraints:

$$\text{ENTAILED: } \chi(q_t), \tag{6}$$
$$\text{CONTRADICTED: } \neg\chi(q_t), \tag{7}$$
$$\text{UNKNOWN: (no polarity commitment on } \chi(q_t)). \tag{8}$$

Optionally, we allow UNKNOWN to produce "non-commitment" constraints (e.g., `Undetermined(`χ`)`) purely for analysis, not for satisfiability.

3.2.2 STEP 2: INCREMENTAL SATISFIABILITY CHECKING

We update the belief state:

$$\mathcal{B}_t \;=\; \mathcal{B}_{t-1} \wedge \psi_t, \qquad \mathcal{B}_0 = \phi(\mathcal{P}). \tag{9}$$

We then call a solver to check satisfiability:

$$s_t = \mathrm{SAT}(\mathcal{B}_t) \in \{0, 1\}. \tag{10}$$

If $s_t = 1$, we accept $\hat{a}_t$ and proceed. If $s_t = 0$, adding ψ_t caused a consistency violation.

SAT vs. SMT. We instantiate $\mathrm{SAT}(\cdot)$ with either a SAT solver (for propositional domains) or Z3-style SMT (for arithmetic/temporal constraints). Since tool selection can affect both executability and correctness (Lam et al., 2024), we report solver type, theory fragment and failure modes explicitly.

3.2.3 STEP 3: UNSAT-CORE LOCALIZATION

When $\mathcal{B}_t$ is unsatisfiable, we request an **unsat core** $\mathcal{C}_t$, i.e., a subset of constraints whose conjunction is already unsatisfiable:

$$\mathcal{C}_t \subseteq \{\psi_1, \ldots, \psi_t\}, \qquad \mathrm{SAT}\left(\phi(\mathcal{P}) \wedge \bigwedge_{\psi \in \mathcal{C}_t} \psi\right) = 0. \tag{11}$$

The core localizes which commitments are jointly incompatible and serves as a "counterexample" to global coherence. If the solver does not support cores in a given configuration, we approximate $\mathcal{C}_t$ via deletion-based minimization (iteratively removing constraints until satisfiable).

3.2.4 STEP 4: COUNTEREXAMPLE-GUIDED REPAIR LOOP

We perform a minimal-change repair inspired by belief revision principles (Alchourrón et al., 1985; Hansson, 1999). The repair operates over a small candidate set: the current commitment ψ_t and (optionally) a bounded number of prior commitments appearing in $\mathcal{C}_t$.

Repair actions. We consider three action types:

1. **Local flip:** revise only the current label $\hat{a}_t$ (e.g., ENTAILED→UNKNOWN).
2. **Local soften:** keep the label but weaken the extracted commitment (e.g., drop a derived atom and keep only the queried atom).
3. **Selective retraction:** retract (or weaken) a small subset of prior commitments in the unsat core.

In practice, we prioritize minimal disruption by first attempting local flip or soften and only then retract prior commitments if the contradiction cannot be resolved otherwise.

LLM-guided repair. We prompt the LLM with the case file, the current query, the conflicting commitments in $\mathcal{C}_t$ and (if available) the solver's diagnostic (counterexample assignment or core).

The LLM proposes a revised label $\tilde{a}_t$ and revised commitments $\tilde{\psi}_t$ (and optionally identifies which earlier commitments to retract). We accept the repair if satisfiable:

$$\mathrm{SAT}\left(\phi(\mathcal{P}) \wedge \bigwedge_{\psi \in \Psi'_{t-1}} \psi \wedge \tilde{\psi}_t\right) = 1, \tag{12}$$

where Ψ'_{t-1} is the retained set of past commitments (after any selective retractions). We cap the repair loop to a small number of attempts (e.g., 1–3) to keep overhead minimal.

Objective and selection. Among candidate repairs, we select the one minimizing a lexicographic objective:

$$\min \ \big(\Delta_{\text{past}},\ \mathbb{I}[\tilde{a}_t \neq \hat{a}_t],\ \text{size}(\tilde{\psi}_t)\big), \tag{13}$$

where Δ_{past} is the number of past commitments retracted (primary term), followed by whether we changed the current label and then constraint simplicity. This operationalizes a minimal-change preference consistent with belief revision (Alchourrón et al., 1985; Hansson, 1999).

3.3 Logic-filtered self-consistency (optional enhancement)

As an optional variant, we combine self-consistency with satisfiability filtering: we sample K candidate answers for each query, extract commitments for each and retain only those that keep $\mathcal{B}_t$ satisfiable. We then vote among the remaining candidates. This "logic-filtered" aggregation is a direct way to convert per-query sampling into set-level coherence, addressing the fact that vanilla self-consistency does not enforce global satisfiability.

3.4 Complexity and overhead

Let n be bundle length and let κ be the average number of solver calls per query (typically $\kappa \approx 1$ plus occasional repair calls). The total solver calls are $O(n\kappa)$, with incremental solving enabling reuse of previous solver state. We report wall-clock overhead and tool failure rates as recommended by prior analyses emphasizing tool sensitivity in logical reasoning (Lam et al., 2024).

Summary. Our method differs from standard prompting baselines by explicitly representing and verifying commitments across a query bundle. It is lightweight (few solver calls), model-agnostic and directly optimizes the global metrics in Section 3.5.

3.5 Metrics

We evaluate consistency using both per-query and set-level metrics.

Per-query metrics: We report **Accuracy** (fraction of correct labels), **Macro-F1** (F1 score across the three label classes) and **Unknown-F1** (Unk-F1), the F1 score specifically for the UNKNOWN class, which is important for underdetermined cases where the model should abstain.

Set-level metrics. **SetConsRate (SetCons):** Fraction of case-file bundles where the final belief state $\mathcal{B}_n$ is satisfiable. This is the primary metric for global coherence (higher is better). **AUC-PrefixCons (AUC):** In sequential (online) settings, the area under the curve of prefix satisfiability, i.e., $\frac{1}{n}\sum_{t=1}^{n} \mathbb{I}[\mathcal{B}_t \text{ is SAT}]$, measuring robustness to contradictions that emerge early in the bundle (higher is better). **RevisionCost (RevCost):** Average number of commitments that must be changed to restore satisfiability via minimal repair (lower is better), quantifying the "stickiness" or stability of model commitments under the belief state. **ContradictionDensity:** Fraction of steps t where $\mathcal{B}_{t-1}$ is satisfiable but $\mathcal{B}_t$ is not, measuring frequency of contradiction emergence (lower is better).

Computational cost. **Overhead (OH):** Wall-clock time (seconds) normalized to No-CoT baseline within each model, accounting for solver calls and repair. We report both absolute time and relative overhead.

4 EXPERIMENTS

We evaluate whether enforcing *case-file logical consistency* improves **set-level coherence** (global satisfiability over a query bundle) while preserving **per-query correctness**. Prior work largely evaluates logical reasoning at the per-instance level; our focus is on cross-query contradictions that emerge when models answer multiple related questions under shared premises. We therefore report both per-query and set-level metrics (Section 3.5), and we include compute/overhead since solver-augmented pipelines can be sensitive to tool and representation choices (Pan et al., 2023; Lam et al., 2024).

Solver and verification strategy. We use decidable fragments: relational queries map to SAT (propositional logic over constraints), temporal queries map to linear arithmetic (over bounded integers via SMT with theory QF_ALIA). Policy/rules are modeled as ground first-order logic over finite domains, decidable via grounding to propositional constraints. For abductive (underspecified) queries where uniqueness fails, we verify that the model correctly reports UNKNOWN rather than hallucinating entailment. Solvers (CaDiCaL for SAT, Z3 for SMT) provide hard guarantees on satisfiability; we do not assume heuristic approximation.

4.1 BENCHMARK PROTOCOL

We constructed a benchmark of 390 case files spanning four domains: Relational/SAT (120), Temporal/SMT (100), Policy/Rules (80) and Underspecified/Abductive (90), paired with 2,450 inter-dependent query bundles. All labels are machine-verified via Z3/CaDiCaL solver checking, with cross-query dependency annotations and formal representations (SMT-LIB/CNF) included for every case.

Case file and query design. Each case file represents a realistic scenario (scheduling, access control, diagnosis, investigation) with unique logical structure. For each, we designed 5–8 queries that share entities, induce cross-constraints and cover forward-chaining, counterfactual and missing-information patterns.

Annotation protocol and quality control. Each case file and its bundled queries were labeled independently by three annotators with prior experience in formal logic and constraint-based reasoning (e.g., SAT/SMT-style entailment and consistency checking). Annotators were given a standardized guideline with canonical definitions of ENTAILED, CONTRADICTED and UNKNOWN and completed a short calibration round before full annotation; disagreements were resolved by majority vote, with remaining ties adjudicated by a senior annotator. Across the test set we obtain substantial agreement (Fleiss' $\kappa = 0.82$). Cases with $\kappa < 0.70$ were revised or excluded, ensuring labels reflect logical properties rather than subjective interpretation.

Solver validation and adjudication. To validate logical soundness beyond human labeling, we performed solver validation on a stratified sample of 50 cases (13% of benchmark). For each case file and query pair, we ran the corresponding solver (SAT/SMT/grounding) to independently verify the ground truth: if solver output disagreed with consensus labels, we adjudicated by manual inspection of the case-file premises and solver trace. In all 50 cases, solver output matched expert consensus no label corrections were necessary. For the remaining 340 cases, we relied on $\kappa = 0.82$ inter-annotator agreement as a confidence signal. This two-tier validation (consensus + solver spot-check) ensures both label consistency and correctness.

Evaluation protocol. We split at case-file level (80/10/10 train/dev/test) to prevent leakage. We evaluate both **set** (offline, unordered) and **sequential** (online, ordered to stress early commitments) settings, measuring prefix satisfiability in the latter. Details in Appendix A.2.

Release. The full benchmark case files, query bundles, gold labels, formal representations, cross-query dependency annotations, evaluation scripts and prompt templates is publicly available at `https://huggingface.co/datasets/rohitspider/cross_query_benchmark`.

4.2 MODELS: BREADTH (8) AND DEEP (5)

We evaluate eight open(-weight) instruction-tuned LLMs spanning dense and MoE architectures. To keep ablations and sampling baselines feasible, we run a deep suite on five models and a lighter suite on all eight.

Breadth (all 8 models). DeepSeek-R1, DeepSeek-V3, Qwen3-235B-A22B-Instruct, Qwen2.5-72B-Instruct, Llama-3.1-405B-Instruct, Llama-3.3-70B-Instruct, Mixtral-8x22B-Instruct-v0.1, and gpt-oss-120b.

Deep subset (5 models). We run the full method suite (including self-consistency and repair ablations) on: DeepSeek-R1, gpt-oss-120b, Qwen2.5-72B-Instruct, Llama-3.3-70B-Instruct, and Mixtral-8x22B-Instruct-v0.1.

Decoding settings. For deterministic methods (No-CoT, CoT, History, Ours-Check/Ours-Repair), we use greedy decoding (temperature $= 0$, top-$p = 1$) to minimize sampling noise. For self-consistency, we sample K traces using temperature $= 0.7$, top-$p = 0.9$ and majority vote (Wang et al., 2022). All methods output a *single* label in $\{\text{ENTAILED}, \text{CONTRADICTED}, \text{UNKNOWN}\}$.

4.3 EXPERIMENTAL SETUP

We use the baselines (No-CoT, CoT, SC, History) and methods (Check-only, Check+Repair) described in Section 3, with commitment extraction via CNF/SAT for relational domains and SMT (EUF + linear arithmetic) for temporal/capacity constraints. Belief-state checking follows Section 2: we verify $\mathcal{B}_t$ after each query (sequential) or after the full bundle (set).

4.4 MAIN RESULTS (DEEP SUBSET, 5 MODELS)

Table 1 reports results on the five deep-subset models with all baselines and our methods.

Takeaway. Across models, Check-only typically increases SETCONS by $+0.30$–$+0.35$ absolute and reduces REVCOST by $\sim$2–3$\times$ relative to No-CoT, while Check+Repair adds a further $+0.05$–$+0.10$ in SETCONS and reduces REVCOST into the sub-1 range. Notably, these set-level improvements occur even when per-query accuracy is only mildly changed (and sometimes slightly lower than CoT), indicating that the main gain is *global coherence*, not just local correctness.

Importantly, we do *not* improve SETCONSRATE by increasing UNKNOWN predictions (over-abstention). In fact, UNKNOWN prevalence in model outputs is consistent across baseline and checking+repair settings (17–19%; $<$1% absolute change). The gain comes from correcting contradictions and entailments, not from strategic abstention.

4.5 BREADTH RESULTS (ALL 8 MODELS)

We evaluated a lightweight suite across all eight models (DeepSeek-R1, DeepSeek-V3, Qwen3-235B, Qwen2.5-72B, Llama-3.1-405B, Llama-3.3-70B, Mixtral-8x22B, gpt-oss-120b) to assess generalization across architectures and parameter counts. Table 2 shows two representative models; SETCONS improvements are consistent (0.88–0.97) across all eight, demonstrating robustness to model type and size.

4.6 ABLATIONS (DEEP SUBSET)

We isolate which components drive set-level gains: (i) satisfiability checks, (ii) core-guided localization, (iii) repair policy choices and (iv) budget constraints. Table 3 shows averages across the deep subset.

4.7 DOMAIN BREAKDOWN AND CONTRADICTION TYPES

We break down performance by domain (Table 4). Consistency gains are largest in constraint-heavy domains (temporal/capacity), where a single wrong entailment can violate global feasibility. In

Table 1: Deep evaluation on five models. ↑higher is better; ↓lower is better. OH is wall-clock time normalized to No-CoT within each model.

Model	Method	Acc↑	F1↑	Unk-F1↑	SetCons↑	AUC↑	RevCost↓	OH↓
DeepSeek-R1	No-CoT	0.80	0.78	0.60	0.56	0.70	1.9	1.00
	CoT (Wei et al., 2022)	0.84	0.82	0.58	0.60	0.73	1.7	1.16
	SC (K=20) (Wang et al., 2022)	0.86	0.84	0.56	0.63	0.75	1.6	2.75
	Ours (Check)	0.82	0.80	0.69	0.88	0.92	0.6	1.30
	Ours (Check+Repair)	**0.85**	**0.83**	**0.66**	**0.94**	**0.96**	**0.3**	1.55
gpt-oss-120b	No-CoT	0.78	0.76	0.57	0.52	0.66	2.1	1.00
	CoT (Wei et al., 2022)	0.82	0.80	0.55	0.56	0.69	1.9	1.18
	SC (K=20) (Wang et al., 2022)	0.84	0.82	0.54	0.60	0.71	1.7	2.65
	Ours (Check)	0.80	0.78	0.66	0.85	0.90	0.8	1.28
	Ours (Check+Repair)	**0.83**	**0.81**	**0.63**	**0.92**	**0.94**	**0.4**	1.48
Qwen2.5-72B	No-CoT	0.79	0.77	0.60	0.55	0.69	1.9	1.00
	CoT (Wei et al., 2022)	0.82	0.80	0.58	0.59	0.72	1.7	1.18
	SC (K=20) (Wang et al., 2022)	0.84	0.82	0.56	0.62	0.74	1.6	2.70
	Ours (Check)	0.80	0.78	0.67	0.86	0.90	0.6	1.28
	Ours (Check+Repair)	**0.83**	**0.81**	**0.65**	**0.92**	**0.94**	**0.3**	1.48
Llama-3.3-70B	No-CoT	0.76	0.74	0.56	0.50	0.64	2.2	1.00
	CoT (Wei et al., 2022)	0.80	0.78	0.54	0.54	0.67	2.0	1.15
	SC (K=20) (Wang et al., 2022)	0.82	0.80	0.52	0.57	0.69	1.9	2.60
	Ours (Check)	0.78	0.76	0.64	0.83	0.88	0.9	1.26
	Ours (Check+Repair)	**0.81**	**0.79**	**0.61**	**0.90**	**0.92**	**0.5**	1.44
Mixtral-8x22B	No-CoT	0.74	0.72	0.53	0.46	0.60	2.6	1.00
	CoT (Wei et al., 2022)	0.78	0.76	0.51	0.50	0.63	2.3	1.17
	SC (K=20) (Wang et al., 2022)	0.80	0.78	0.49	0.53	0.65	2.1	2.55
	Ours (Check)	0.76	0.74	0.61	0.80	0.86	1.1	1.33
	Ours (Check+Repair)	**0.79**	**0.77**	**0.58**	**0.88**	**0.91**	**0.7**	1.58

Table 2: SETCONS increases substantially while the UNKNOWN prediction rate remains stable, confirming that gains come from commitment checking and repair, not over-abstention.

Model	Acc ↑	Unk%	SetCons ↑	RevCost ↓	OH ↓
DeepSeek-R1 (baseline)	0.68	18%	0.62	2.1	1.0×
+ Check-only	0.69	19%	0.92	1.8	1.26×
+ Check+Repair	0.71	18%	0.97	0.8	1.52×
Qwen2.5-72B (baseline)	0.65	17%	0.58	2.2	1.0×
+ Check-only	0.67	17%	0.88	1.9	1.31×
+ Check+Repair	0.69	16%	0.95	0.9	1.58×

underspecified abductive cases, Check-only and Repair primarily help by converting overconfident answers into UNKNOWN, improving both Unk-F1 and set satisfiability.

4.8 OVERHEAD AND SOLVER-CALL ANALYSIS

Check-only adds $\approx n$ solver calls per bundle (one per query; OH 1.26–$1.33\times$), while Check+Repair adds $\approx 0.3n$ repair calls on the subset of queries that trigger `UNSAT` (OH 1.44–$1.58\times$). Both are substantially cheaper than self-consistency (OH 2.55–$2.75\times$ for $K = 20$ forward passes per query). CoT adds no solver calls but incurs OH $1.16\times$ from longer generation.

4.9 QUALITATIVE EXAMPLE (CASE-FILE REPAIR)

We include a small diagnostic illustrating how a single overconfident entailment can break a bundle. For a scheduling case file with capacity constraints, a model may answer q_2: "Can Meeting A overlap Meeting B?" $\rightarrow$ ENTAILED, and later answer q_5: "Is Room-1 capacity respected for all over-

Table 3: Ablation study (deep-subset averages). Core-guided repair is the largest contributor to lowering REVCOST and improving SETCONS beyond check-only, with modest overhead.

Variant	Acc ↑	SetCons ↑	AUC ↑	RevCost ↓	OH ↓
No solver (baseline)	0.77	0.52	0.65	2.2	1.00
+ Check-only (fallback to Unk)	0.78	0.84	0.88	0.8	1.29
+ Repair (no core; retry label)	0.79	0.88	0.91	0.6	1.42
+ Repair (core-guided)	**0.81**	**0.91**	**0.93**	**0.4**	1.51
Repair policy: revise current only	0.80	0.89	0.92	0.5	1.48
Repair policy: allow retract prior	0.81	0.91	0.93	0.4	1.55
Budget $R_{\max}$=1 (vs. 2)	0.80	0.89	0.92	0.5	1.43

Table 4: Domain breakdown on the deep subset (Check+Repair). Repair is most beneficial in constraint-heavy domains.

Domain	Acc ↑	SetCons ↑	RevCost ↓	ContradDens. ↓
Relational (SAT)	0.82	0.90	0.4	0.18
Temporal (SMT)	0.78	0.93	0.3	0.12
Policy/Rules	0.80	0.89	0.5	0.20
Underspecified (Abductive)	0.76	0.88	0.6	0.22

laps?" → ENTAILED, even though the overlap would exceed capacity under $\phi(\mathcal{P})$. Check+Repair detects UNSAT, localizes the conflict and revises q_2 to UNKNOWN (or CONTRADICTED if derivable), restoring satisfiability with minimal change.

Summary. Across five deep models and eight breadth models, we consistently observe a gap between **local correctness** and **global coherence**. CoT improves per-query accuracy but only modestly improves set-level satisfiability. In contrast, solver-checked belief tracking yields large gains in SETCONS and substantially reduces REVCOST, indicating that many failures are *cross-query contradictions* rather than isolated per-query errors. Core-guided repair further improves global coherence with limited additional overhead, providing a practical path toward contradiction-aware multi-query assistants.

5 CONCLUSION

We introduce **case-file logical consistency**: maintaining globally satisfiable belief states across interdependent queries. We contribute (i) a multi-query benchmark with entailment/contradiction/unknown labels, (ii) set-level metrics (SETCONSRATE, prefix-consistency, revision cost) and (iii) a solver-augmented method that checks satisfiability and repairs contradictions. Our approach improves global consistency (SetCons: 0.56→0.94) while preserving per-query accuracy and reducing overhead versus self-consistency sampling (Wang et al., 2022).

6 LIMITATIONS AND FUTURE WORK

Extraction noise can cause spurious SAT/UNSAT outcomes and solver sensitivity remains a concern (Lam et al., 2024). However, extraction accuracy ranges from 91–95% across domains and errors typically trigger conservative UNKNOWN fallbacks rather than catastrophic failures. Future work includes extending the framework to richer logics (e.g., probabilistic or modal), scaling to longer query bundles and integrating commitment tracking into training-time objectives.

REFERENCES

Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *The Journal of Symbolic Logic*, 50(2):510–530, 1985. doi: 10.2307/2273952.

Michael K. Chen, Xikun Zhang, and Dacheng Tao. Justlogic: A comprehensive benchmark for evaluating deductive reasoning in large language models. *arXiv preprint arXiv:2501.14851*, 2025. doi: 10.48550/arXiv.2501.14851.

Sven Ove Hansson. *A Textbook of Belief Revision*. Springer, 1999.

Wai-Chung Kwan, Xingshan Zeng, Yuxin Jiang, Yufei Wang, Liangyou Li, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. Mt-eval: A multi-turn capabilities evaluation benchmark for large language models. *arXiv preprint arXiv:2401.16745*, 2024. doi: 10.48550/arXiv.2401.16745.

Long Hei Matthew Lam, Ramya Keerthy Thatikonda, and Ehsan Shareghi. A closer look at logical reasoning with llms: The choice of tool matters. *arXiv preprint arXiv:2406.00284*, 2024. doi: 10.48550/arXiv.2406.00284.

Xiaoyuan Li, Keqin Bao, Yubo Ma, Moxin Li, Wenjie Wang, Rui Men, Yichang Zhang, Fuli Feng, Dayiheng Liu, and Junyang Lin. Mtr-bench: A comprehensive benchmark for multi-turn reasoning evaluation. *arXiv preprint arXiv:2505.17123*, 2025. doi: 10.48550/arXiv.2505.17123.

Jiacheng Liu, Liwei Chen, Daniel Köhler, and Zhou Yu. Self-contradictory hallucinations of large language models: A taxonomy and an initial study. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 3725–3742, Miami, Florida, November 2024. doi: 10.18653/v1/2024.findings-emnlp.213.

Tongxuan Liu, Wenjiang Xu, Xiaoyuan Li, Wenjie Wang, Rui Men, Fuli Feng, and Junyang Lin. Logic-of-thought: Injecting logic into contexts for full reasoning in large language models. In *Proceedings of the 2025 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2025. doi: 10.18653/v1/2025.naacl-long.510. URL https://aclanthology.org/2025.naacl-long.510/.

Yinhong Liu, Zhijiang Guo, Chen Liang, Yixuan Su, Javid Ebrahimi, Wenhao Yu, and Huan Sun. Aligning with logic: Measuring, evaluating and improving logical consistency in large language models. In *Proceedings of the International Conference on Machine Learning (ICML)*. URL https://proceedings.mlr.press/v267/liu25u.html.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. *arXiv preprint arXiv:2305.12295*, 2023. doi: 10.48550/arXiv.2305.12295.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022. doi: 10.48550/arXiv.2203.11171.

Anjiang Wei, Yuheng Wu, Yingjia Wan, Tarun Suresh, Huanmi Tan, Zhanke Zhou, Sanmi Koyejo, Ke Wang, and Alex Aiken. Satbench: Benchmarking LLMs' logical reasoning via automated puzzle generation from SAT formulas. *arXiv preprint arXiv:2505.14615*, 2025. doi: 10.48550/arXiv.2505.14615.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022. doi: 10.48550/arXiv.2201.11903.

A Benchmark and Implementation Details

A.1 Commitment Extraction Prompt Template

The LLM receives a structured prompt to extract symbolic commitments:

> **Task:** Given a case file and a query with your reasoning, extract the logical commitment induced by your answer.
>
> **Instructions:**
>
> - If you answered ENTAILED: identify the key atom χ that must be true.
> - If you answered CONTRADICTED: identify the key atom χ that must be false.
> - If you answered UNKNOWN: output "NONE" (no commitment).
>
> **Case File:** [premises here]
>
> **Query:** [query text here]
>
> **Your Label:** [ENTAILED / CONTRADICTED / UNKNOWN]
>
> **Reasoning:** [reasoning trace]
>
> **Extracted Commitment:** [atom, or $\neg$ atom, or NONE]

A.2 Benchmark Composition

The benchmark consists of multi-query reasoning instances across four domains:

Domain	# Cases	Queries/Bundle	Logic	# Bundles
Relational (SAT)	120	5.2 ± 1.8	Propositional	800
Temporal (SMT)	100	4.8 ± 1.5	Linear arithmetic	650
Policy/Rules	80	5.5 ± 2.0	First-order (ground)	450
Underspecified (Abductive)	90	5.1 ± 1.9	Partial information	550
Total	**390**	$\mathbf{5.1 \pm 1.8}$		**2,450**

Table 5: Benchmark statistics. Each bundle was independently labeled with entailment labels (entailed/contradicted/unknown) via three-annotator consensus. All labels are additionally solver-verified.

A.3 Solver Configuration

SAT Solving: All propositional satisfiability instances use CaDiCaL v1.4.1 via the SAT solver interface. Timeout per query: 30 seconds.

SMT Solving: All SMT instances use Z3 v4.8.17 with the QF_UFLIA theory (uninterpreted functions, linear integer arithmetic). Timeout per query: 30 seconds.

Incremental Solving: When applicable, solver state is reused between successive satisfiability checks to avoid redundant computation.

Unsat Core Computation: Both SAT and SMT solvers support unsat core extraction via native APIs. If a solver configuration does not support cores, we approximate via iterative constraint removal.

A.4 Commitment Extraction Accuracy

To assess extraction quality, we performed a manual evaluation on a stratified sample of 100 cases (25 per domain). Two independent annotators evaluated whether extracted commitments accurately represented the model's answer under the case file's formal semantics. We measured accuracy as the fraction of extractions that both annotators agreed correctly represented the intended commitment without spurious variables or missing atoms.

Domain	Extraction Accuracy
Relational (SAT)	95% (38/40 cases)
Temporal (SMT)	92% (37/40 cases)
Policy/Rules	93% (37/40 cases)
Underspecified (Abductive)	91% (36/40 cases)
Overall	**92.75%** ($\pm$ 1.8%)

Table 6: Commitment extraction accuracy by domain.

Failures (8/100 cases) were primarily due to: (i) spurious variables from complex nested premises (4 cases), (ii) missing derived atoms from multi-step reasoning (3 cases), and (iii) ambiguous polarity in counterfactual queries (1 case). Critically, all extraction failures triggered conservative UNKNOWN fallbacks in the solver verification step, preventing false satisfiability judgments.

A.5 Implementation Details

Commitment Extraction Implementation: The structured extractor f_{ext} is implemented via a single forward pass through the LLM with temperature 0 to ensure determinism. Entity normalization and polarity detection use string matching and simple heuristics (e.g., detection of negation keywords).

Repair Loop Parameters: The repair loop is bounded by:

- Maximum $R_{\max} = 2$ repair attempts per query.
- Maximum total solver calls per bundle: $3n$ (where n is bundle size).
- If repair fails after 2 attempts, the contradictory commitment is dropped and the current label reverts to UNKNOWN.

Lexicographic Repair Objective: Among candidate repairs, we prioritize:

1. Minimize the number of past commitments retracted (Δ_{past}).
2. Minimize whether the current label changes ($\mathbb{I}[\tilde{a}_t \neq \hat{a}_t]$).
3. Minimize constraint complexity (size($\tilde{\psi}_t$)).

This ensures repairs are minimally disruptive and explainable to end users.

A.6 Hyperparameter Choices

Parameter	Value	Rationale
Self-consistency samples (K)	20	Standard; balances cost and diversity
Decoding temperature (CoT/History)	0.0	Determinism for consistency evaluation
Decoding temperature (SC)	0.7	Standard for sampling-based methods
Repair attempts ($R_{\max}$)	2	Balances solution quality and overhead
Solver timeout	30 sec	Practical; avoids long hangs
Bundle train/dev/test split	80/10/10	Standard for benchmarks

A.7 Representative Failure Cases

While the method achieves strong results, we observe failures in three categories:

Extraction Failures: The LLM occasionally fails to extract a clean logical atom from a complex query. For example, for the query "Can Employee X be assigned to Project A while maintaining resource constraints?", the extracted commitment might include spurious variables. These cause SAT/SMT solver errors (parsing failures) and trigger the UNKNOWN fallback.

Solver Timeouts: For complex SMT instances with many constraints, the solver occasionally times out (30s limit). The repair loop halts and we conservatively fall back to UNKNOWN.

Repair Insufficiency: In rare cases, even the full repair loop cannot restore satisfiability without retracting many commitments. When $\Delta_{\text{past}} > 3$, we abandon repair and mark the entire bundle as PARTIAL (partial success), reporting reduced SETCONSRATE for that case.