Evaluating Large Reasoning Models on Belief Revision

Nurul Fajrin Ariyani[1,2], Zied Bouraoui[3], Richard Booth[2], Steven Schockaert[2]
[1]Dept. of Informatics, Institut Teknologi Sepuluh Nopember, Indonesia
[2]Cardiff NLP, Cardiff University, UK [3]CRIL-CNRS & University of Artois, France
nf.ariyani@its.ac.id {boothr2,schockaerts1}@cardiff.ac.uk bouraoui@cril.fr

Abstract

Logic offers a natural framework for benchmarking LLM-based reasoning models. While frontier models perform well on standard logical reasoning tasks, it remains unclear to what extent these capabilities generalize to other forms of reasoning. In this paper, we use belief revision problems to stress-test the reasoning abilities of these models on a less familiar task, which builds on satisfiability and entailment checking but requires a careful strategy to manage complexity. We show that belief revision is indeed challenging for existing models, even when only small and relatively simple problem instances are considered.

1 Introduction

Large Reasoning Models (LRMs), such as DeepSeek-R1 (Guo et al., 2025) and the models from OpenAI's o1 family[1], improve on the reasoning abilities of LLMs by generating long chain-of-thought (CoT) derivations, having been optimized for this form of "thinking" through reinforcement learning. While they have led to impressive advances in mathematics and coding, the extent to which their reasoning abilities generalize to other forms of problem solving remains unclear.

Recent work, among others, has evaluated LRMs on graph search (Rameshkumar et al., 2025), planning (Valmeekam et al., 2025), puzzles (Shojaee et al., 2025), satisfiability checking in propositional logic (Hazra et al., 2025) and graph coloring (Heyman and Zylberberg, 2025). Broadly speaking, LRMs have been found to outperform LLMs on such problems (albeit with limitations once problem instances reach a certain complexity level). A key reason for their success stems from their ability to simulate systematic search: while CoTs are linear sequences of derivation steps, inspection of LRMs' reasoning traces suggests that they are capable of simulating operations such as backtracking within such sequences, allowing them to simulate classical strategies for satisfiability checking (Hazra et al., 2025) and search methods such as A^* (Lehnert et al., 2024; Stechly et al., 2025).

However, concerns remain about the generalization abilities of LRMs. For instance, Das et al. (2025) recently found o3 to fail on a particular type of relational reasoning problems, even for instances that can be straightforwardly solved in two or three reasoning steps, while Stechly et al. (2025) found that models that were trained to solve one type of mazes were unable to solve a much simpler class of mazes. The observation that LRMs reason about propositional satisfiability by simulating classical SAT solvers may thus merely reflect the presence of ground truth reasoning traces from such solvers in their training data. Indeed, Li et al. (2025) found that LRM performance depends on the presence, in the training data, of reasoning traces with the right structure, where the validity of the reasoning traces themselves, surprisingly, appears to be less important (Stechly et al., 2025; Li et al., 2025). Yue et al. (2025) found that, compared to LLMs, LRMs improve sample efficiency but do not actually discover new reasoning traces, casting further doubts on their general purpose reasoning abilities.

To further our understanding of the reasoning abilities of LRMs, in this paper we focus on the problem of belief revision (Gärdenfors, 1988). Given a knowledge base K, expressed as a set of propositional logic formulas, and an observation α, expressed as another formula, the task we consider is to remove a minimal set of formulas from K such that α is logically consistent with the remaining formulas in K. As a special case, when $K \cup \{\alpha\}$ is already consistent, then no formulas should be removed. Otherwise, restoring consistency involves removing one or more formulas. While belief revision has been extensively studied within philosophy and knowledge representation (Hansson, 1999;

[1]https://openai.com/index/openai-o1-system-card/

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 717–730
November 6-10, 2026 ©2026 Association for Computational Linguistics

Peppas, 2008), to the best of our knowledge, the considered reasoning task has not previously been considered for evaluating LRMs (or LLMs more generally). Within a broader setting, the question of how LLMs revise their beliefs has been studied from various angles (De Cao et al., 2021; Hase et al., 2023; Wilie et al., 2024), but these works do not study formal reasoning problems. More closely related, Helwe et al. (2025) introduce a reasoning benchmark which requires LLMs to update their beliefs as facts are added to and removed from an initial knowledge base. However, their challenge is about keeping track of knowledge edits, rather than about detecting and repairing inconsistencies.

Belief revision poses unique challenges, which make this task particularly interesting for evaluating LRMs. Problems such as satisfiability checking or graph colouring can essentially be solved by guessing and verifying a solution. Doing well on these tasks is then a matter of (i) being able to systematically search for possible solutions, (ii) being able to recognize a valid solution once it has been found, and (iii) relying on suitable heuristics for deciding which partial solutions are most promising to explore. Belief revision cannot be cast into this framework: we can guess which formulas need to be removed but verifying the validity of such a guess still requires solving some NP-complete problems, in particular for proving the minimality of the set of formulas.[2] Solving large problem instances is thus computationally hard, even for symbolic solvers. However, smaller instances can often be solved relatively easily, by reasoning about the causes of any inconsistencies. As such, our focus is on evaluating whether LRMs are capable of solving *simple* belief revision instances. Our hypothesis is that solving such instances requires more creativity than previously considered reasoning problems (for evaluating LRMs), and that this will pose a challenge for LRMs.

2 Problem Formulation

We recall some basic notions from propositional logic and formally define the belief revision task that we consider in this paper. Let $\mathcal{A}$ be a set of atomic propositions, called *atoms*. A literal l is either an atom from $\mathcal{A}$ or the negation of such an atom. A clause is a disjunction of literals. An interpretation ω is a truth assignment $\mathcal{A} \rightarrow \{\mathit{true}, \mathit{false}\}$. This truth assignment can be straightforwardly extended to clauses. For instance, ω makes $\neg a \vee b$ true if $\omega(a) = \mathit{false}$ or $\omega(b) = \mathit{true}$. For a clause α, we write $\omega \models \alpha$ to denote that ω makes α true. In such a case, we say that ω is a model of α. For a set of clauses K, we also write $\omega \models K$ to denote that ω makes every clause in K true, in which case we call ω a model of K. We say that K is satisfiable if K has a model ω. Otherwise, we say that K is unsatisfiable or inconsistent. If K is a set of clauses and α is a clause, we say that K entails α, written $K \models \alpha$, if every model of K is a model of α.

Assume we are given a set of clauses K, reflecting our *initial beliefs* about some domain, and another clause α, which reflects some kind of *observation*. If $K \cup \{\alpha\}$ is unsatisfiable, then our initial beliefs must be incorrect. The problem of belief revision is to modify K to some set $\widehat{K}$ such that $\widehat{K} \cup \{\alpha\}$ is satisfiable. Different possible choices for this modified set $\widehat{K}$ can generally be made. As such, various principles have been studied in the belief revision literature to guide the revision process. The most central principle is that of minimal change, i.e. we want $\widehat{K}$ to be as similar to K as possible. This notion of minimality can itself be defined in different ways. In this paper, we focus on a simple but important special case, where $\widehat{K}$ is a maximal subset of K. In other words, $\widehat{K} \subseteq \mathcal{K}$ is considered a valid revision of K if (i) $\widehat{K} \cup \{\alpha\}$ is satisfiable and (ii) for every $\widetilde{K}$ such that $\widehat{K} \subset \widetilde{K} \subseteq K$ it holds that $\widetilde{K} \cup \{\alpha\}$ is unsatisfiable. There can be multiple valid revisions $\widehat{K}$ for a given problem instance (K, α). In such case, we simply require that the model selects any of these.

3 Dataset Generation

Since belief revision is computationally hard, we cannot expect LRMs to perform well on large or complex problem instances. Our focus is therefore on analyzing whether LRMs can solve *simple* belief revision problems. We use synthetically generated instances, as they allow us to control the difficulty and avoid dataset contamination.[3] To generate a problem instance, we first sample a random set of clauses. Each sampled clause is a disjunction of three literals, where every literal is an atom or negated atom with equal probability. The atoms are sampled from a set $\mathcal{A}$ of cardinality 6, and each problem instance consists of 26 clauses. This

[2] Formally, decision problems related to belief revision tend to be located on the second, or even third, level of the polynomial hierarchy (Liberatore and Schaerf, 1996).

[3] Our problem generator and the generated datasets themselves are available at https://github.com/ariyaninf/belief_revision_sat.

ensures that the ratio of clauses to atoms approximates the critical value of 4.267, for which randomly generated 3-SAT instances are least likely to be trivial (Mitchell et al., 1992). From these randomly sampled sets of clauses X, we select those that satisfy the following two criteria: (i) X is unsatisfiable, but there are distinct clauses α and β in X such that $X \setminus \{\alpha\}$ and $X \setminus \{\beta\}$ are satisfiable; (ii) there are at least two distinct minimal subsets of X which are unsatisfiable. The second condition ensures that instances cannot simply be solved by identifying a single conflict (i.e. the task is to find a clause whose removal fixes *all* conflicts). Once a set X satisfying both criteria has been found, with corresponding clauses α and β, we use $X \setminus \{\alpha\}$ as the (satisfiable) set of beliefs K and α as the observation. The existence of the clause β ensures that the belief revision problem can be solved by removing a single clause from the set of beliefs.

We sample a total of 100 such problem instances. Each of the clauses are converted to natural language by using the verbalization strategy from Ariyani et al. (2025), which uses templates to convert a clause such as $a \vee \neg b \vee c$ to a phrase such as *Alice is smart or Alice is not tall or Alice is helpful*. The prompts used to feed the problem instances to LRMs are listed in Appendix C. We also sample 100 problem instances (K, α), where $K \cup \{\alpha\}$ is satisfiable, i.e. $\widehat{K} = K$, to analyze whether models can identify whether any clauses have to be removed at all. As for the former instances, we have $|K \cup \{\alpha\}| = 26$, where all clauses consist of three literals and involve a total of 6 atoms.

4 Experimental Results

Table 1 summarizes the results for a variety of models. We show results separately for the 100 problem instances (K, α) for which $K \cup \{\alpha\}$ is unsatisfiable (i.e. where the model is expected to remove a clause from K) and for the 100 problem instances where $K \cup \{\alpha\}$ is already satisfiable. The table shows the number of instances that were answered correctly, as well as a breakdown of the error type for the remaining instances. In particular, we distinguish three types of errors: (i) cases where the model returns a satisfiable set of clauses, but where more clauses than needed have been removed (shown as NM); (ii) cases where the observation α was not successfully added to the final set of clauses (shown as MO); and (iii) cases where the set of clauses that was returned is (still) inconsistent (shown as Inc).[4] The results confirm the challenging nature of the task, especially for the unsatisfiable problem instances. Interestingly, Grok-4 correctly solved all but one of these cases, but all the other models failed in a significant number of cases. Due to the small set of the dataset[5], minor performance differences have to be interpreted with caution.

Analysis of Performance Variability The results in Table 1 are based on a single run. To analyze performance variability, Table 2 shows the number of incorrectly answered problem instances across three runs. The aggregate performance of DeepSeek-V3.2 and Qwen-3 is particularly stable, with only small differences in the number of errors across runs. In contrast, models such as Gemini-2.5-Flash and DeepSeek-R1 appear to be somewhat more variable. Stable aggregate performance does not necessarily imply that the same problem instances are answered incorrectly. Table 2 therefore also reports the number of instances that were incorrectly answered in all three runs. For example, DeepSeek-V3.2 and Qwen-3 do not have any overlapping incorrectly answered problem instances, despite having very similar error counts in each run. More generally, for several models, the overlap is substantially smaller than the number of errors in each individual run. This suggests that models primarily fail because they act as noisy reasoners, rather than being unable to deal with certain problems (Khalid et al., 2025). Note that some of the models from Table 1 could not be included in this analysis, as they have been deprecated since we carried out the main experiments.

Analysis on Variants of the Benchmark To stress-test the best-performing models, in Table 3 we show results for a set of problem instances that require two clauses to be removed for restoring consistency (while otherwise being generated as explained in Section 3. Surprisingly, the models perform similarly to Table 1, showing that this setting does not pose significant additional challenges.

Figure 1 shows the impact of changing the number of clauses and the number of atoms. Performance generally decreases when more clauses are

[4] Note that in some cases, the observation is missing from the result and the resulting set of clauses is moreover inconsistent. Such cases are counted under both MO and Inc, hence the values reported do not always add up to 100.

[5] This was needed to manage the high cost of some of the tested models; see Appendix D for an analysis on cost.

	Unsatisfiable					Satisfiable				
	Acc. (%)	95% CI	NM	MO	Inc	Acc. (%)	95% CI	NM	MO	Inc
o3	81	[73, 88]	2	1	16	**100**	[100, 100]	0	0	0
o3-mini	30	[21, 39]	10	31	33	98	[95, 100]	0	2	0
o4-mini	30	[21, 39]	12	6	54	94	[89, 98]	0	6	0
DeepSeek-R1	51	[41, 61]	5	1	43	79	[71, 87]	6	18	0
DeepSeek-V3.2	91	[85, 96]	4	0	5	89	[83, 95]	0	11	0
Gemini-2.5-Flash	33	[24, 43]	20	0	47	71	[62, 80]	26	4	0
Gemini-2.5-Pro	51	[41, 61]	7	1	41	91	[85, 96]	9	0	0
Claude-3.7	40	[31, 50]	28	0	32	93	[88, 98]	6	1	0
Qwen-3	78	[70, 86]	13	1	8	98	[95, 100]	0	2	0
QWQ	2	[0, 5]	14	36	53	88	[81, 94]	2	11	1
Olmo-3-7B	4	[1, 8]	10	81	6	14	[8, 21]	7	86	0
Olmo-3-32B	16	[9, 23]	15	26	45	71	[62, 80]	1	28	0
Phi-4	3	[0, 7]	21	53	35	7	[2, 12]	88	58	2
Grok-4	**99**	[97, 100]	0	0	1	98	[95, 100]	2	0	0
Grok-4.3	83	[75, 90]	1	0	16	96	[92, 99]	4	0	0

Table 1: Main results on 100 unsatisfiable and 100 satisfiable belief-revision instances. We report the results in terms of accuracy (*Acc.*) and also provide the corresponding 95% bootstrap confidence intervals (CI). The remaining columns show a breakdown of the errors: the number of instances where the set of removed clauses is not minimal (NM), the number of instances where the observation is missing from the final knowledge base (MO) and the number of instances where the inconsistency was not resolved (Inc).

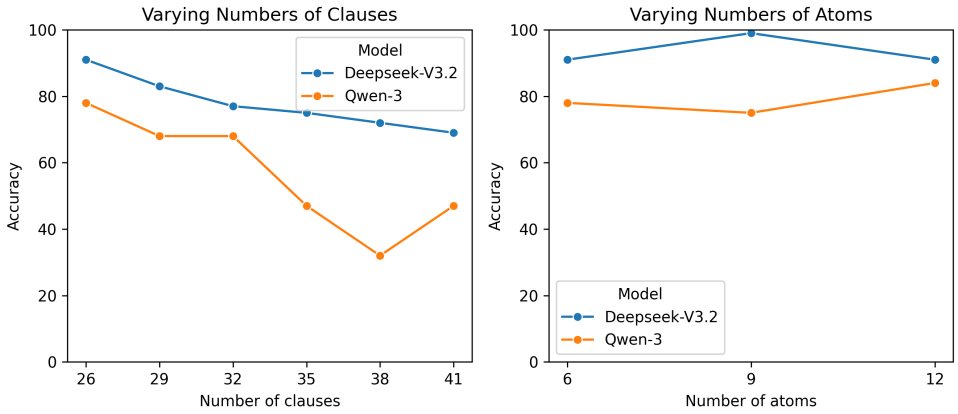


Figure 1: Left: problems with 6 atoms and a varying number of clauses. Right: problems with 26 clauses and a varying number of atoms. Each data point shows the number of correctly solved problem instances, out of 100. All instances require one clause to be removed.

	r1	r2	r3	Overlap
o3	17	21	25	1
o3-mini	60	66	66	25
o4-mini	58	66	56	24
DeepSeek-R1	44	31	38	9
DeepSeek-V3.2	5	4	4	0
Gemini-2.5-Flash	47	66	72	30
Gemini-2.5-Pro	42	42	42	12
Qwen-3	9	10	9	0
Phi-4	85	80	76	54
Grok-4.3	19	16	27	8

Table 2: Total number of incorrectly answered problem instances across 100 unsatisfiable belief revision problems. The table shows results across three different runs (r1–r3), as well as the number of problem instances that were incorrectly answered in all three runs (overlap).

	Correct	NM	MO	Inc
o3	73	2	2	25
DeepSeek-V3.2	91	6	0	9
Qwen-3	70	6	2	28
Grok-4	97	0	0	3

Table 3: Evaluation on 100 unsatisfiable belief revision instances, where two clauses have to be removed.

considered. However, increasing the number of atoms sometimes improves performance, presumably because the reasoning problems become easier when moving away from the critical ratio of 4.267. We evaluated Grok-4 on the problems with 6 atoms and 38 clauses (not shown in the figure). It solved 90 instances correctly, suggesting that its perfor-

mance is also sensitive to the number of clauses. Additional analysis can be found in Appendix D.

5 Qualitative Analysis

We present a qualitative analysis of the reasoning traces of some models to better understand the overall strategies that are employed. While concerns have been raised about the faithfulness of reasoning traces (Chen et al., 2025), their structure may still give insights into why some models fail.

Reasoning Strategies Several strategies can be observed in the reasoning traces. First, DeepSeek-V3.2 attempts to find a satisfying assignment for all atoms by initializing truth values (often starting from the observation) and repeatedly performing consistency checks. While reasoning traces for Grok-4 are not disclosed, we observed that this model consistently provides satisfying assignments to support its justifications, suggesting that it may use a similar approach. In contrast, Qwen-3 starts by examining whether the negation of the observation ($\neg\alpha$) can be derived from the existing beliefs (K). The model then identifies subsets of premises responsible for this derivation as candidates for removal and performs resolution-based checks to restore consistency. Other models, such as Claude-3.7 and Gemini-2.5-Flash, often terminate their reasoning once a set of clauses is found whose removal restores consistency, without checking minimality.

Failure Modes Belief revision can be solved in a principled way either by systematically constructing an interpretation which satisfies the observation together with a maximal set of clauses (DeepSeek-V3.2) or through syntactic satisfiability checking using resolution (Qwen-3). However, the number of possible truth assignments increases exponentially in the number of atoms while resolution might lead to an exponential number of clauses. Both strategies thus require good heuristics for prioritizing truth assignments or resolution steps. In the presence of long derivations, some models tend to skip inference steps and jump to conclusions in an (apparently) heuristic fashion. The DeepSeek-R1 reasoning traces, in particular, often show this behaviour.[6] While all models attempt to identify a consistent set of clauses, ensuring that only a minimal set of clauses is removed requires a systematic process that appears to be missing in the strategy followed by Claude-3.7 and Gemini-2.5-Flash, and is only noisily implemented in other models. For most models, we see cases where the observation has not actually been added to the set of beliefs, which may partially reflect problems with instruction following. For the smaller models (QWQ, Olmo-3, and Phi-4) we observed a tendency to make trivial mistakes that can invalidate the entire reasoning chain, such as flipping the polarity of literals or simply omitting some clauses.

[6] Examples can be found in Appendix D.

6 Conclusions

We have evaluated LRMs on belief revision in propositional logic. We found that this task is particularly challenging for LLM-based reasoning models. Apart from Grok-4, all models made a significant number of mistakes, despite the fact that only simple problem instances were considered. The experiments clearly show the progress that has recently been made (e.g. with significant improvements between DeepSeek-R1 and DeepSeek-V3.2), while also showing that key obstacles remain for tasks where exhaustive search is not feasible.

Limitations

Our focus in this paper has been to demonstrate the usefulness of belief revision for benchmarking LRMs. While the results clearly show that this task is challenging for current models, we have not attempted to exhaustively analyze the underlying factors, given the prohibitive cost of large-scale experiments with LRMs. For instance, the difficulty of the problems can be further increased by sampling problem instances with a larger number of clauses (as shown in our analysis in Figure 1), or by specifically controlling for reasoning depth (e.g. the number of inference steps needed to identify inconsistencies). More fundamentally, the problems can also be made (considerably) harder by focusing on different variants of belief revision. For instance, rather than asking for any repair, we could ask the model to identify those clauses which belong to *all* minimal repairs (i.e. to identify the intersection of all valid repairs), or to determine whether a given hypothesis β can be inferred from all such repairs. Some variants of such problems have a complexity at the third level of the polynomial hierarchy and would thus require strong heuristics to be solved effectively.

Acknowledgments

Nurul Ariyani was supported by Institut Teknologi Sepuluh Nopember (ITS) through the Academic Excellence Research Funding, 3830/PKS/ITS/2026. Steven Schockaert was supported by EPSRC grants EP/W003309/1 and EP/Y028805/1. Zied Bouraoui was supported by ANR-22-CE23-0002 ERIANA.

References

Marah Abdin, Sahaj Agarwal, Ahmed Awadallah, Vidhisha Balachandran, Harkirat Behl, Lingjiao Chen, Gustavo de Rosa, Suriya Gunasekar, Mojan Javaheripi, Neel Joshi, Piero Kauffmann, Yash Lara, Caio César Teodoro Mendes, Arindam Mitra, Besmira Nushi, Dimitris Papailiopoulos, Olli Saarikivi, Shital Shah, Vaishnavi Shrivastava, and 4 others. 2025. Phi-4-reasoning technical report. *Preprint*, arXiv:2504.21318.

Nurul Fajrin Ariyani, Zied Bouraoui, Richard Booth, and Steven Schockaert. 2025. There's no such thing as simple reasoning for llms. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 4503–4514. Association for Computational Linguistics.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, Vladimir Mikulik, Samuel R. Bowman, Jan Leike, Jared Kaplan, and Ethan Perez. 2025. Reasoning models don't always say what they think. *CoRR*, abs/2505.05410.

Anirban Das, Irtaza Khalid, Rafael Peñaloza, and Steven Schockaert. 2025. When no paths lead to Rome: Benchmarking systematic neural relational reasoning. In *Proceedings of The Thirty-Ninth Annual Conference on Neural Information Processing Systems*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenhao Xu, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, and 245 others. 2025. Deepseek-v3.2: Pushing the frontier of open large language models. *Preprint*, arXiv:2512.02556.

Peter Gärdenfors. 1988. *Knowledge in Flux: Modeling the Dynamics of Epistemic States.* The MIT Press.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nat.*, 645(8081):633–638.

Sven Ove Hansson. 1999. *A Textbook of Belief Dynamics - Theory Change and Database Updating*, volume 11 of *Applied logic series*. Kluwer.

Peter Hase, Mona Diab, Asli Celikyilmaz, Xian Li, Zornitsa Kozareva, Veselin Stoyanov, Mohit Bansal, and Srinivasan Iyer. 2023. Methods for measuring, updating, and visualizing factual beliefs in language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2714–2731, Dubrovnik, Croatia. Association for Computational Linguistics.

Rishi Hazra, Gabriele Venturato, Pedro Zuidberg Dos Martires, and Luc De Raedt. 2025. Have large language models learned to reason? a characterization via 3-SAT. In *Second Conference on Language Modeling*.

Chadi Helwe, Sultan AlRashed, and Francesco Orabona. 2025. ReviseQA: A benchmark for belief revision in multi-turn logical reasoning. In *ICML 2025 Workshop on Assessing World Models*.

Alex Heyman and Joel Zylberberg. 2025. Evaluating the systematic reasoning abilities of large language models through graph coloring. *CoRR*, abs/2502.07087.

Irtaza Khalid, Amir Masoud Nourollah, and Steven Schockaert. 2025. Large language and reasoning models are shallow disjunctive reasoners. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 8843–8869. Association for Computational Linguistics.

Daniel Le Berre and Anne Parrain. 2010. The sat4j library, release 2.2, system description. *Journal on Satisfiability Boolean Modeling and Computation*, 7:59–64.

Lucas Lehnert, Sainbayar Sukhbaatar, Paul McVay, Michael Rabbat, and Yuandong Tian. 2024. Beyond A*: Better planning with transformers via search dynamics bootstrapping. In *Proceedings of the First Conference on Language Modeling*.

Dacheng Li, Shiyi Cao, Tyler Griggs, Shu Liu, Xiangxi Mo, Eric Tang, Sumanth Hegde, Kourosh Hakhamaneshi, Shishir G. Patil, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. 2025. LLMs can easily learn to reason from demonstrations structure, not content, is what matters! *CoRR*, abs/2502.07374.

Paolo Liberatore and Marco Schaerf. 1996. The complexity of model checking for belief revision and

update. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI 96, IAAI 96, Portland, Oregon, USA, August 4-8, 1996, Volume 1*, pages 556–561. AAAI Press / The MIT Press.

David G. Mitchell, Bart Selman, and Hector J. Levesque. 1992. Hard and easy distributions of SAT problems. In *Proceedings of the 10th National Conference on Artificial Intelligence, San Jose, CA, USA, July 12-16, 1992*, pages 459–465. AAAI Press / The MIT Press.

Team Olmo, :, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, and 50 others. 2025. Olmo 3. *Preprint*, arXiv:2512.13961.

Pavlos Peppas. 2008. Belief revision. In Frank van Harmelen, Vladimir Lifschitz, and Bruce W. Porter, editors, *Handbook of Knowledge Representation*, volume 3 of *Foundations of Artificial Intelligence*, pages 317–359. Elsevier.

Revanth Rameshkumar, Jimson Huang, Yunxin Sun, Fei Xia, and Abulhair Saparov. 2025. Reasoning models reason well, until they don't. *CoRR*, abs/2510.22371.

Parshin Shojaee, Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. 2025. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. *CoRR*, abs/2506.06941.

Kaya Stechly, Karthik Valmeekam, Atharva Gundawar, Vardhan Palod, and Subbarao Kambhampati. 2025. Beyond semantics: The unreasonable effectiveness of reasonless intermediate tokens. *CoRR*, abs/2505.13775.

Karthik Valmeekam, Kaya Stechly, Atharva Gundawar, and Subbarao Kambhampati. 2025. A systematic evaluation of the planning and scheduling abilities of the reasoning model o1. *Trans. Mach. Learn. Res.*, 2025.

Bryan Wilie, Samuel Cahyawijaya, Etsuko Ishii, Junxian He, and Pascale Fung. 2024. Belief revision: The adaptability of large language models reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10480–10496, Miami, Florida, USA. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? *CoRR*, abs/2504.13837.

A Illustration of the Belief Revision Task

Figure 2 illustrates the considered belief revision task. A knowledge base K and observation p is provided. The model is asked to add p to K, and if necessary remove clauses from K to ensure that the final set of clauses is satisfiable. In the example on the left, the model indeed has to remove a statement. There are three possible statements that can be removed to ensure satisfiability (referred to as *nominees* in the figure). Removing any of these leads to a valid answer. However, to ensure minimality[7], only one of them should be removed. On the right, an example is given where the observation is already consistent with the knowledge base. In this case, the model should not remove any clauses.

B Additional Details

Models Table 4 presents an overview of the specific LRMs that have been used in our experiments: o3, o3-mini, o4-mini, DeepSeek-R1 (Guo et al., 2025), DeepSeek-V3.2 (DeepSeek-AI et al., 2025), Gemini-2.5-Flash, Gemini-2.5-Pro, Claude-3.7, Qwen-3 (Yang et al., 2025), QWQ, Olmo-3-7B (Olmo et al., 2025), Olmo-3-32B (Olmo et al., 2025), Phi-4 (Abdin et al., 2025), Grok-4, and Grok-4.3. The OpenAI models (o3, o3-mini, o4-mini) have been evaluated through the OpenAI API. For the other models, we use a unified API gateway provided by OpenRouter[8]. All models are evaluated in a zero-shot setting, receiving a structured prompt that permits intermediate reasoning before producing a final answer. To manage cost, we report results from a single run for each model. Tool calling was not enabled when evaluating the different models.

Satiafiability Checking To check for satisfiability while generating the dataset, we relied on the

[7] In this paper, minimality refers to "subset-minimality". In other words, a set of clauses is minimal if there is no strict subset of these clauses whose removal would be sufficient. This differs from the notion of "cardinality-minimality" which is sometimes also used in the belief revision literature, requiring that the number of clauses is minimal among all the possible repairs.

[8] The platform is accessible at `https://openrouter.ai/`

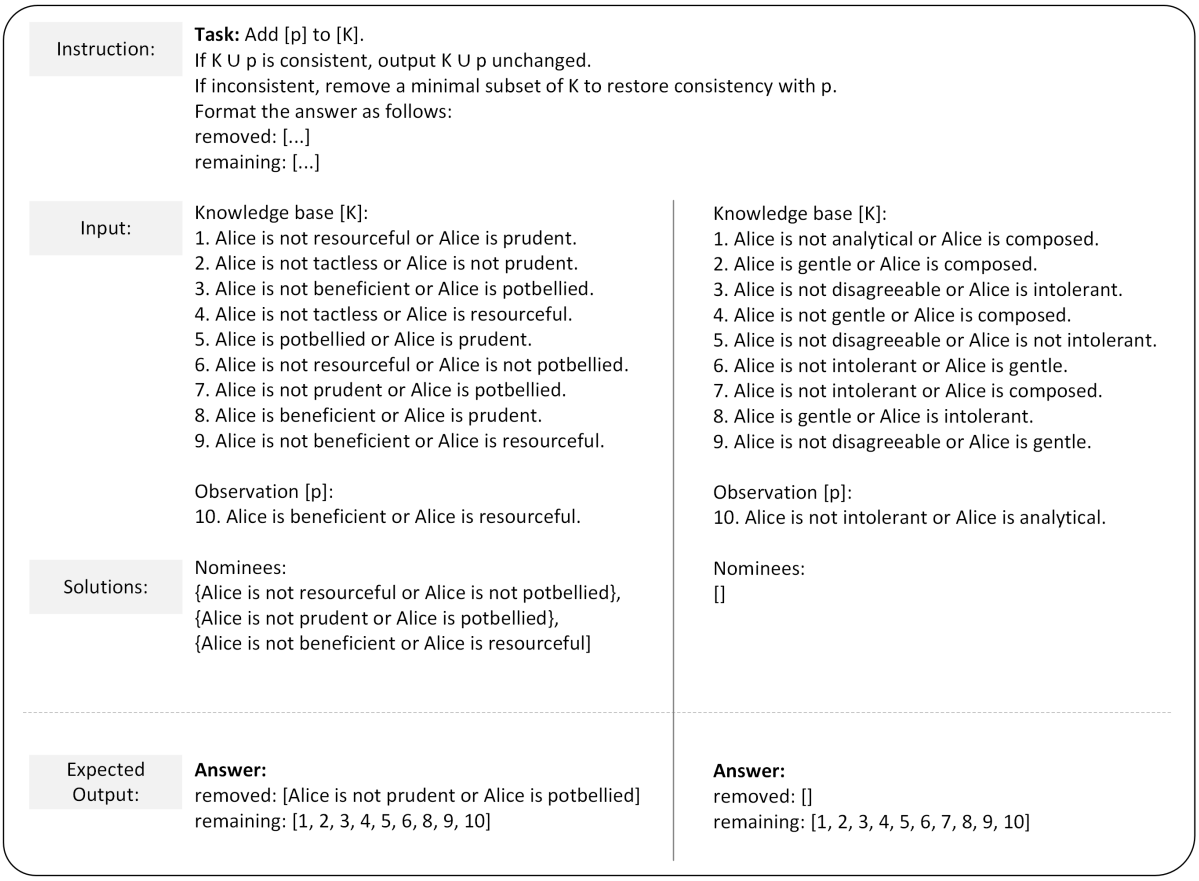


Figure 2: Illustration of two belief revision problem instances. LEFT: revision requiring clause removal to restore consistency. RIGHT: revision where no removal is necessary

Sat4J solver (Le Berre and Parrain, 2010). Similarly, during evaluation, to test the satisfiability of the returned set of clauses, we converted them back to symbolic form and again used Sat4J.

C Prompt Sensitivity Analysis

The prompt we used for the belief revision experiments in Table 1 is shown in Figure 3. We now compare this prompt with two alternatives. First, the prompt in Figure 4 is more structured than the default prompt, which might benefit particular models. Second, the prompt in Figure 5 is a more compact alternative to the default prompt. The results for the different prompts are compared in Table 5. The default prompt yields the best performance for almost all models, with o4-mini being a notable exception. Remarkably, some of the best-performing models achieve a similar performance with the compact prompt (o3, DeepSeek-V3.2 and Grok-4). For other models, however, the performance with the compact prompt collapses (e.g. DeepSeek-R1, Gemini-2.5-Pro, Qwen-3). Note that we have omitted Olmo-3-7B from Table 5 as we did not manage to obtain non-trivial results with the two alternative prompts.

To further analyze whether better results are possible by improving the prompt, we carried out two additional experiments with DeepSeek-V3.2 and Qwen-3. In particular, we experimented with the instruction-driven prompt which is shown in Figure 6, and the CoT-driven prompt which is shown in Figure 7. DeepSeek-V3.2 correctly answered 82 instances with the instruction-driven prompt and 71 with the chain-of-thought prompt. Qwen-3 answered 64 and 49 instances, respectively. In both cases, the models performed substantially worse than with the default prompt. The instruction-driven prompt produces better results than the CoT-driven prompt, presumably because it uses formal language (e.g. *logical inconsistency*, *unsatisfiable* or *contradictory*) that is better suited for SAT-style reasoning.

D Additional Analysis

Entailment Checking Table 6 shows results for an entailment checking task, with problem in-

Model	Model ID	Param.	Max Tokens	License
o3†	o3	-	200K	OpenAI Terms / Services Agreement
o3-mini†	o3-mini	-	200K	OpenAI Terms / Services Agreement
o4-mini†	o4-mini	-	200K	OpenAI Terms / Services Agreement
DeepSeek-R1	deepseek/deepseek-r1	671B	164K	MIT License
DeepSeek-V3.2	deepseek/deepseek-v3.2	671B	164K	MIT License
Gemini-2.5-Flash†	google/gemini-2.5-flash	-	1.05M	Gemini API
Gemini-2.5-Pro†	google/gemini-2.5-pro	-	1.05M	Gemini API
Claude-3.7†	anthropic/claude-3.7-sonnet:thinking	-	200K	Anthropic
Qwen-3	qwen/qwen3-235b-a22b-thinking-2507	235B	262K	Apache-2.0
QWQ	qwen/qwq-32b	32B	33K	Apache-2.0
Olmo-3-7B	allenai/olmo-3-7b-think	7B	66K	Apache 2.0
Olmo-3-32B	allenai/olmo-3.1-32b-think:free	32B	66K	Apache 2.0
Phi-4	microsoft/phi-4-reasoning-plus	14B	33K	MIT License
Grok-4†	x-ai/grok-4	1.7T	256K	xAI Terms of Service
Grok-4.3†	x-ai/grok-4.3	ĨT	1.0M	xAI Terms of Service

Table 4: LRMs used in the experiments. Models marked with † are closed-source, while the remaining models are open-weight or fully open-source.

user:
Task:
You are an automated belief revision system.
Your goal is to integrate the observations into the premises while ensuring that the resulting set remains logically consistent.

Instructions:
1. You are given two inputs: Premises and Observations.
2. Check whether Premises ∪ Observations is logically consistent.
3. If Premises ∪ Observations is consistent, output: removed: [], remaining: [Premises ∪ Observations].
4. If Premises ∪ Observations is inconsistent, perform belief revision as follows:
a. Remove the minimal subset of premises such that (Premises − Removed) ∪ Observations becomes consistent.
b. Do not remove observations.
c. Minimise the number of removed premises, not their content.
5. Your output must be in JSON, exactly in this format:
{{remove: [], remaining: [], justification: explain the reasoning behind your answer.}}
6. Do not add new information. Do not modify the original statements. Only remove premises when necessary to restore consistency.

[Premises] = [insert a set of beliefs]
[Observations] = [insert observations]

Figure 3: Default prompt used for the main experiments (*default*).

user:
Below are instructions describing a task, paired with an input.

Instruction:
Add [New] to [KB]. Carefully check whether [KB] ∪ [New] is logically consistent. If consistent, return removed: [], remaining: [all premises and observations]. If inconsistent, remove a minimal subset from KB to restore consistency such that (KB − removed) ∪ [New] becomes consistent.
Return in JSON format:
{ removed: [...], remaining: [...], justification: "" }

Question:
[KB] = [insert a set of beliefs]
[New] = [insert observations]

Response:
<think>{{}}</think>
{{}}

Figure 4: More structured alternative to the default prompt (*structured*).

stances derived from the unsatisfiable belief revision instances. Specifically, given a belief revision instance (K, α), we ask the model to verify whether $K \models \neg\alpha$.[9] Because only unsatisfiable belief revision instances were used, all examples are positive (i.e. the entailment always holds). This means that a trivial model could in principle achieve a perfect accuracy by always predicting entailment, hence the results of this analysis need to be interpreted with some degree of caution. We focus on these positive instance because they correspond to the reasoning that is needed to detect the inconsistency in the unsatisfiable belief revision problems. If models can solve these entailment problems but not the corresponding belief revision instances, it suggests that their failure is due to the particular challenges that arise from the belief revision task, rather than because of any limitations in their reasoning abilities.

As can be seen, several state-of-the-art models can solve this entailment task nearly perfectly, including o3, DeepSeek-V3.2, Qwen-3 and Grok-4. This demonstrates that the challenges faced by o3 and Qwen-3 on the belief revision task are not due to limitations in their deductive reasoning abilities. Similarly, while smaller models still struggle with entailment checking, all models perform sig-

[9]Note that $\neg\alpha$ is a conjunction of three literals.

developer: You are a logic assistant working in propositional logic.

user:
Task: Add [New] to [KB].
If KB ∪ [New] is consistent, output KB ∪ [New] unchanged.
If consistent, remove a minimal subset of KB to restore consistency with New.
Return the result in JSON format as:
{"removed": [...], "remaining": [...], "justification": "one short sentence"}
[KB] = [insert a set of premises]
New = [insert observations]

Figure 5: More compact alternative to the default prompt (*compact*).

nificantly better than on belief revision, showing that the latter task presents unique challenges. The prompt we used for this entailment task is shown in Figure 13.

Effect of Verbalization Strategy For the main experiments, we relied on a verbalization strategy in which each atom corresponds to a phrase of the form "Alice is *[adjective]*". This may potentially confuse the model if it attempts to apply some kind of commonsense reasoning based on the meaning of the adjectives. We have therefore repeated the experiment with an alternative verbalization strategy, where atoms correspond to phrases of the form "*[Name]* is *[adjective]*". In this case, each atom talks about a different person, thus avoiding this potential source of confusion. With this alternative verbalization strategy, Qwen-3 answered 80 instances correctly while DeepSeek-V3.2 answered 85 instances correctly. These results are in line with the variability reported in Table 2, suggesting that the impact of this verbalization strategy is limited.

We also evaluated a variant where we used a structured symbolic representation, using single letters for atoms and - for negation. For instance, we would write "-a or b or -c" for the clause $\neg a \vee b \vee \neg c$". In this case, Qwen-3 answered 85 instances correctly while DeepSeek-V3.2 answered 82 instances correctly.

Token Usage and Cost Table 7 shows the number of tokens that was generated, on average, by the models in the main experiments. Table 9 shows the number of tokens for the additional experiment involving belief revision instances that require two clauses to be removed. There is no clear relationship between the number of generated tokens and the correctness of the answer. However, on average, for satisfiable belief revision instances, the number of tokens is considerably lower than for unsatisfiable instances. This is in accordance with the fact that satisfiable instances are easier to solve (since they only require a single satisfiability check). As can be seen in Table 3, unsatisfiable belief revision problems where two clauses need to be removed lead to the highest number of generated tokens, which is in accordance with the increased difficulty of these problems.

You are a logical reasoning assistant specialising in belief revision.
You are given a set of PREMISES and an OBSERVATION.

PREMISES:
{premises}

OBSERVATION:
{observations}

TASK:
1. Check whether adding the OBSERVATION to the PREMISES creates a logical inconsistency (i.e., the combined set becomes unsatisfiable or contradictory).
2. If NO inconsistency exists, return the OBSERVATION added to the PREMISES as the remaining set with nothing removed.
3. If an inconsistency EXISTS, restore consistency by removing the minimum number of premises necessary so that the OBSERVATION is logically compatible with the remaining premises. The OBSERVATION must always be retained.
4. If multiple minimal removal sets exist, choose the one that preserves the most informative or relevant premises with respect to the OBSERVATION.

Return your answer strictly in the following JSON format:
{{
"removed": ["<removed premise 1>", "<removed premise 2>", ...],
"remaining": ["<remaining premise 1>", "<remaining premise 2>", ..., "<observation>"],
"justification": "<step-by-step explanation of why the inconsistency exists and why those specific premises were selected for removal>"
}}

Rules:
- The OBSERVATION must always appear in "remaining" and must never appear in "removed".
- Remove as FEW premises as possible.
- If no premises need to be removed, "removed" must be an empty list [].
- Do not output anything outside the JSON block.

Figure 6: Instruction-driven alternative to the default prompt (*instruction-driven*).

Table 8 shows the overall cost of the main experiments (Table 1) and the entailment experiment (Table 6). Unsurprisingly, o3, Claude-3.7 and Grok-4 emerge as the most expensive models. The cost of using these models also quickly increases when more challenging problem instances are considered. For instance, the experiment with Grok-4 from Table 3, where two clauses had to be removed, was 35.11 USD, compared to 22.16 USD for the problems where only one clause had to be removed.

Varying the Number of Atoms and Clauses Table 10 shows the detailed results for DeepSeek-V3.2 and Qwen-3 when the number of atoms and

You are an expert in belief revision. Your job is to maintain a consistent set of beliefs when new information arrives.

Here is a set of things currently believed to be true (PREMISES):
{premises}

A new piece of information has been observed (OBSERVATION):
{observations}

Your task:
- First, reason step by step about whether the OBSERVATION conflicts with any of the PREMISES.
- If there is NO conflict, simply add the OBSERVATION to the belief set.
- If there IS a conflict, identify which premises are responsible for the inconsistency and remove the smallest possible number of them to make the full belief set, including the OBSERVATION, logically consistent again.
- When deciding which premises to remove, prefer removing premises that are less specific, less directly supported, or most in conflict with the OBSERVATION.
- The OBSERVATION is always treated as the ground truth and must be kept.

Think through this carefully, then produce your final answer in the following JSON format only:
{{
"removed": ["<premise that was removed>", ...],
"remaining": ["<premise still believed>", ..., "<the observation>"],
"justification": "<your reasoning: what conflict was found, which premises caused it, and why you chose to remove those specific ones over others>"
}}

Important reminders:
- Minimise removals by only removing what is strictly necessary.
- The OBSERVATION must always be in "remaining".
- If no conflict exists, "removed" should be [].
- Output only the JSON. No extra text before or after.

Figure 7: Chain-of-thought-driven alternative to the default prompt (*chain-of-thought-driven*).

clauses is varied. This corresponds to the experiment which was summarized in Figure 1 in the main paper.

Analysis of Reasoning Traces Figures 8–12 show fragments of reasoning traces of the considered LRMs, illustrating some of the observations that have been reported in Section 5. In particular, Figure 8 illustrates the heuristic nature of the DeepSeek-R1 reasoning process, where the model starts checking all the clauses, but then gives up and simply writes *Etc.* While this does not necessarily prove that the actual reasoning process followed by the model is heuristic, this observation is consistent with the weaker performance of DeepSeek-R1, especially when it comes to providing answers that are inconsistent. Figure 9 illustrates how DeepSeek-R1 sometimes gives up and switches strategy when reasoning becomes too complex. In contrast, Figure 10 illustrates the thorough process followed by DeepSeek-V3.2, where the impact of setting a particular literal to *true* is

```
statement. Therefore, to maintain consistency, I
      need a model that satisfies both KB - {1} and
      the new statement.
But perhaps there is such a model. For example,
      suppose I set abnormal=false. Then the new
      statement is satisfied (since ¬abnormal is
      true). Now, can I satisfy the remaining KB?

Statement 1 is removed, so I don't have it.
Statement 2: blunt ∨ fussy ∨ orderly. Okay.

With abnormal=false, let's see other statements.

Statement 6: ¬fussy ∨ clever ∨ abnormal → ¬fussy
      v clever (since abnormal false).
Statement 8: ¬abnormal ∨ ¬blunt ∨ ¬clever →
      since abnormal false, this is true.
Statement 9: ¬abnormal ∨ ¬fussy ∨ blunt → true.
Statement 10: ¬blunt ∨ clever ∨ fussy.

Etc.

I need to find a model where all statements are
      satisfied with abnormal=false.
Suppose abnormal = false.
```

Figure 8: Example where DeepSeek-R1 skips some clauses in consistency checking with "Etc.".

systematically checked for every clause. Figure 11 provides an example of the output of Grok-4, showing how its justification mentions a satisfying assignment (even though the reasoning trace for this model is hidden). Finally, Figure 12 shows how smaller models such as Olmo-3-7b can sometimes make trivial mistakes.

```
6. Alice is not fussy or Alice is clever or Alice is
      abnormal. (¬fussy ∨ clever ∨ abnormal)
7. ... and so on.
Hmm, this is quite a long list. Trying to parse each
      one individually is going to be time-consuming
     . I think I can look for statements that could
     conflict with the new statement.
The new statement is ¬abnormal ∨ ¬fusst ∨ ¬blunt.
     Now, look at statement 1: ¬fussy ∨ ¬blunt ∨
     abnormal. If I negate statement 1, it would be
     fussy ∧ blunt ∧ ¬abnormal. But the new
     statement is ¬abnormal ∨ ¬fussy ∨ ¬blunt,
     which is equivalent to abnormal → (¬fussy ∨
     ¬blunt). Wait, that's getting messy.
Alternatively, combining this new statement with
     statement 1:
Statement 1: ¬fussy ∨ ¬blunt ∨ abnormal
New: ¬abnormal ∨ ¬fussy ∨ ¬blunt
If I consider both, adding them together:
(¬fussy ∨ ¬blunt ∨ abnormal) ∧ (¬abnormal ∨ ¬
     fussy ∨ ¬blunt)
Notice that if abnormal is true, the new statement
     requires ¬fussy ∨ ¬blunt.
```

Figure 9: Example where DeepSeek-R1 switches strategy in mid-trace when reasoning becomes complex.

```
We can try to set Ab = true? That might simplify
     many clauses. Let's try: Ab = true. Then New
     becomes:¬true ∨ ¬F ∨ ¬B = false ∨ ¬F ∨ ¬B
     = ¬F ∨ ¬B. So New requires ¬F ∨ ¬B, i.e.,
     not both F and B true. Clause 1: ¬F ∨ ¬B ∨
     true = true, satisfied. Clause 2: B ∨ F ∨ O,
     ok. Clause 3: ¬S ∨ ¬C ∨ O, ok. Clause 4: ¬S
     ∨ ¬F ∨ ¬B, ok. Clause 5: ¬O ∨ ¬F ∨ ¬B, ok.
      Clause 6: ¬F ∨ C ∨ true = true. Clause 7: ¬
     O ∨ ¬C ∨ true = true. Clause 8: ¬true ∨ ¬B
     ∨ ¬C = false ∨ ... = ¬B ∨ ¬C, so requires ¬
     B ∨ ¬C. Clause 9: ¬true ∨ ¬F ∨ B = false ∨
     ... = ¬F ∨ B. Clause 10: ¬B ∨ C ∨ F, ok.
     Clause 11: ¬B ∨ F ∨ true = true. Clause 12:
     ¬B ∨ ¬C ∨ true = true. Clause 13: ¬true ∨ ¬
     O ∨ C = false ∨ ... = ¬O ∨ C. Clause 14: ¬O
     ∨ C ∨ true = true. Clause 15: ¬S ∨ ¬B ∨ ¬C,
      ok. Clause 16: ¬C ∨ B ∨ true = true. Clause
     17: ¬true ∨ C ∨ F = false ∨ ... = C ∨ F.
     Clause 18: ¬O ∨ ¬C ∨ F, ok. Clause 19: ¬S ∨
      B ∨ true = true. Clause 20: ¬S ∨ ¬F ∨ C, ok.
      Clause 21: ¬O ∨ F ∨ true = true. Clause 22:
     ¬S ∨ C ∨ B, ok. Clause 23: ¬S ∨ ¬true ∨ ¬O
      = ¬S ∨ false ∨ ¬O = ¬S ∨ ¬O. Clause 24: ¬
     O ∨ ¬F ∨ B, ok. Clause 25: F ∨ O ∨ true =
     true.
```

Figure 10: A reasoning fragment of thorough procedural consistency checking in DeepSeek-V3.2

		Cor	NM	MO	Inc
o3	default	81	2	1	16
	structured	69	5	2	24
	compact	81	3	0	16
o3-mini	default	30	10	31	33
	structured	13	10	51	29
	compact	34	10	10	48
o4-mini	default	30	12	6	54
	structured	25	6	6	65
	compact	16	3	5	76
DeepSeek-R1	default	51	5	1	43
	structured	39	3	1	57
	compact	26	1	5	68
DeepSeek-V3.2	default	91	4	0	5
	structured	76	13	1	10
	compact	85	6	4	5
Gemini-2.5-Flash	default	33	20	0	47
	structured	28	24	2	47
	compact	8	11	2	79
Gemini-2.5-Pro	default	51	7	1	41
	structured	51	12	0	37
	compact	23	14	0	63
Claude-3.7	default	40	28	0	32
	structured	38	17	2	43
	compact	25	12	0	63
Qwen-3	default	78	13	1	8
	structured	59	7	23	11
	compact	25	3	59	13
QWQ	default	2	14	36	53
	structured	4	7	70	21
	compact	3	11	35	56
Olmo-3-32B	default	16	15	26	45
	structured	1	2	96	2
	compact	0	19	72	10
Phi-4	default	3	21	53	35
	structured	4	10	23	64
	compact	1	9	79	14
Grok-4	default	99	0	0	1
	structured	92	2	4	2
	compact	89	1	9	1
Grok-4.3	default	83	1	0	16
	structured	6	1	21	73
	compact	3	1	12	84

Table 5: Comparison of the performance on 100 unsatisfiable belief revision instances (using the same instances as in Table 1), for three different prompts: the default prompt (Figure 3), a more structured alternative (Figure 4) and a more compact alternative (Figure 5). With the compact prompt, performance collapses for most models; the default prompt provides the best results overall; Grok-4, o3, and DeepSeek-V3.2 are robust across all three prompt variants.

	Acc
o3	**99**
o3-mini	75
o4-mini	85
DeepSeek-R1	83
DeepSeek-V3.2	95
Gemini-2.5-Flash	73
Gemini-2.5-Pro	76
Claude-3.7	90
Qwen-3	98
QWQ	43
Olmo-3-7B	17
Olmo-3-32B	74
Phi-4	86
Grok-4	**99**
Grok-4.3	84

Table 6: Results for 100 entailment checking problems. The problem instances were derived from the unsatisfiable belief revision problems that were used for the main experiments (Table 1). As a result, all problem instances are positive.

```
"removed": [
        "Alice is not abnormal or Alice is not blunt
       or Alice is not clever."
],
"remaining": [
        "Alice is not fussy or Alice is not blunt or
       Alice is abnormal.",
        "Alice is blunt or Alice is fussy or Alice
      is orderly.",
        ...
        "Alice is not abnormal or Alice is not fussy
       or Alice is not blunt."
],
"justification": "The original premises are
      inconsistent with the observation because the
      premises logically imply that Alice is abnormal
      , fussy, and blunt, which contradicts the
      observation that at least one of these is false
      . Removing the specified premise breaks the
      chain of implications that forced this
      contradiction, making the remaining premises
      consistent with the observation. A satisfying
      assignment is: Alice is abnormal, blunt, clever
      , not fussy, not orderly, not sedate."
```

Figure 11: Grok-4 output containing justifications with a satisfying assignment.

```
19. Premise 19: ¬D ∨ ¬U ∨ S (not distraught or not
       unsure or sedate)
20. Premise 20: ¬D ∨ ¬F ∨ S (not distraught or not
       fastidious or sedate)
21. Premise 21: ¬S ∨ ¬D ∨ F (not sedate or not
      distraught or fastifious)
22. Premise 22: ¬S ∨ ¬I ∨ D (same as Premise 17?
      Wait, checking: Premise 17 is ¬S ∨ ¬I ∨ D,
      and Premise 22 is the same as Premise 17. Hmm,
      maybe a duplicate? Not sure, but I'll treat
      them as separate for now.)
23. Premise 23: ¬D ∨ ¬U ∨ I (not distraught or not
       unsure or illogical)
24. Premise 24: ¬S ∨ ¬I ∨ F (not sedate or not
      illogical or fastidious)
25. Premise 25: ¬D ∨ ¬F ∨ I (not distraught or not
       fastidious or illogical)
26. Premise 26: ¬S ∨ ¬I ∨ ¬F (not sedate or not
      illogical or not fastidious)
27. Premise 27: ¬N ∨ ¬D ∨ ¬F (not natural or not
      distraught or not fastidious)
Wait, let me recount. Wait, the user listed 27
      premises? Let me check again the original list.
       The user provided:
The [Premises] list has 27 entries? Let me recount
      the premises given in the problem: Looking at
      the user's input:
Premises listed as:
1. Alice is not natural or Alice is not illogical or
       Alice is fastidious. (¬N ∨ ¬I ∨ F)
```

Figure 12: An example of a trivial mistake from Olmo-3-7b where the model unintentionally duplicates some premises.

user:
You are a logic assistant working in propositional logic.
You will be given input consisting of:
- A list of propositional premises
- A hypothesis

Your task: Determine whether the set of premises entails the hypothesis.
Important rules:
- Output only a single word: "Yes" or "No".
- Do not provide explanations.

[Premises] = [insert a set of beliefs]
[Observations] = [insert observations]

Figure 13: Entailment prompt used for evaluating all models in a zero-shot setting.

	Belief revision				Entailment
	Unsatisfiable		Satisfiable		
	Correct	Incorrect	Correct	Incorrect	
o3	13870	10186	3228	0	10237
o3-mini	14058	13493	6656	5778	5573
o4-mini	5825	8889	3271	3006	6148
DeepSeek-R1	12162	11907	5045	5481	6810
DeepSeek-V3	13726	11149	6111	0	6375
Gemini-2.5-Flash	19459	17069	5182	7587	12398
Gemini-2.5-Pro	18759	19651	10814	195866	13882
Claude-3.7	20014	16262	8692	16997	14658
Qwen-3	19790	19525	9820	11580	9842
QWQ	10133	13187	7639	8756	10432
Olmo-3-7B	17468	15625	13671	13925	10839
Olmo-3-32B	18110	19596	13699	13970	10813
Phi-4	1205	1306	1149	1136	1130
Grok-4	14572	8507	10280	15136	6103
Grok-4.3	12775	9953	4243	8374	4826

Table 7: Average number of tokens generated for each of the considered models and problem types. For belief revision, the results are shown separately for instances that were solved correctly and for those where a mistake was made (i.e. the revision was not minimal, the observation was missing, or the resulting set of clauses is still inconsistent).

	Belief revision		Entailment
	Unsat.	Sat.	
o3	10.60	2.58	8.29
o3-mini	6.04	2.92	2.51
o4-mini	3.13	1.43	2.76
DeepSeek-R1	3.06	1.33	1.74
DeepSeek-V3.2	0.56	0.26	0.27
Gemini-2.5-Flash	4.60	1.49	3.11
Gemini-2.5-Pro	19.22	11.68	13.95
Claude-3.7	28.44	14.04	22.17
Qwen-3	0.00	0.00	0.00
QWQ	0.52	0.32	0.42
Olmo-3-7B	0.33	0.29	0.22
Olmo-3-32B	0.97	0.70	0.55
Phi-4	0.02	0.02	0.02
Grok-4	22.16	15.91	9.39

Table 8: Total cost in USD for the belief revision experiments in Table 1 and the entailment checking experiments in Table 6.

	Belief revision (2 clauses)	
	Correct	Incorrect
o3	17092	11386
Grok-4	23388	15257
DeepSeek-V3.2	20210	20183
Qwen-3	24176	23461

Table 9: Average number of tokens generated for the belief revision instances that require two clauses to be removed to restore consistency.

#atoms	#clauses	Deepseek-V3.2	Qwen-3
6	26	91	78
6	29	83	68
6	32	77	68
6	35	75	47
6	38	72	32
6	41	69	47
9	26	99	75
12	26	91	84

Table 10: Results across problem instances with varying numbers of atoms and a varying number of clauses. The reported results are for 100 problem instances per configuration, with all instances requiring one clause to be removed.