CUSP: EVALUATING REASONING ON THE BOUNDARY OF SOLVABILITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models have made notable progress on reasoning and mathematics tasks. We consider reasoning in the context of set theory and propose a benchmark for evaluating how well language models reason about sets under constraints. Problem instances are constructed programmatically and every one is verified with an SMT solver. Our datasets sweep the fraction of constraints retained from a complete pairwise system, from redundant but uniquely solvable systems down to *exactly-minimal* ones in which every retained constraint has been deletion-tested; the minimal slice sits on the *boundary of (unique) solvability*. We evaluate two tasks: (1) generate an assignment of elements to sets satisfying all constraints, and (2) state how many solutions the problem has. Accuracy varies substantially along two levers that we pull simultaneously — the complexity of the problem instance and the pragmatics of the prompt — and we localize the pragmatic effect to whether a model commits to an assignment rather than to the assignment it computes. Some behaviors we observe are consistent with reports of short-cutting in earlier literature; others are novel and warrant further investigation.

1 INTRODUCTION

Language models were built for natural language applications (Brown et al., 1988), not for reasoning, much less mathematics; the analytical capabilities a community has long sought to imbue them with (Graves et al., 2014) became useful only recently. Benchmarks abound for language tasks and linguistic phenomena (Wang et al., 2018; Warstadt et al., 2019) and for fixed (Hendrycks et al., 2021b) or parameterized (Anil et al., 2022) analytical tasks. We believe it's time to ask: *what analytical capabilities are most important when evaluating a language model as a system that will reason operationally about the world?* We take the position that a foundational capability is reasoning about discrete objects under constraints.

That language models are intelligent systems queried and instructed in natural language also makes *pragmatics* more important than ever (Singh, 2025; Su & Zhang, 2025; Chun et al., 2026). This is adjacent to the alignment problem: what a prompt says explicitly, what it implies, and the facts of the world can all diverge. As we show, a pragmatic cue about whether a task's solution is unique changes accuracy substantially — but only when the prompt also offers the model a way out of answering, which locates the effect in abstention policy rather than in set computation.

We present CUSP, a benchmark that gives a language model a system of constraints over named sets drawn from a finite universe and asks it either to produce a satisfying assignment (*assignment task*) or to classify the solution space (*classification task*). Every instance is a solver-verified uniquely solvable constraint set: we begin with a complete set of pairwise constraints and prune them, using Monte Carlo sampling verified by a Satisfiability Modulo Theories (SMT) solver, toward the point where removing any more would destroy unique solvability. The main datasets sweep the retained fraction of constraints along that trajectory; a derived track carries the exactly-minimal cores at its endpoint (mean 7.0 constraints, every one deletion-tested, versus 59.2 in the parent sweep). This procedure lets us approach the boundary of solvability, where short-cutting behaviors are particularly consequential and where pragmatic framing can readily tip a model's behavior; CUSP pulls the levers of problem complexity and pragmatic variation simultaneously, unlike prior benchmarks, which, to our knowledge, fix one lever when the other is pulled. It is particularly diagnostic for

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

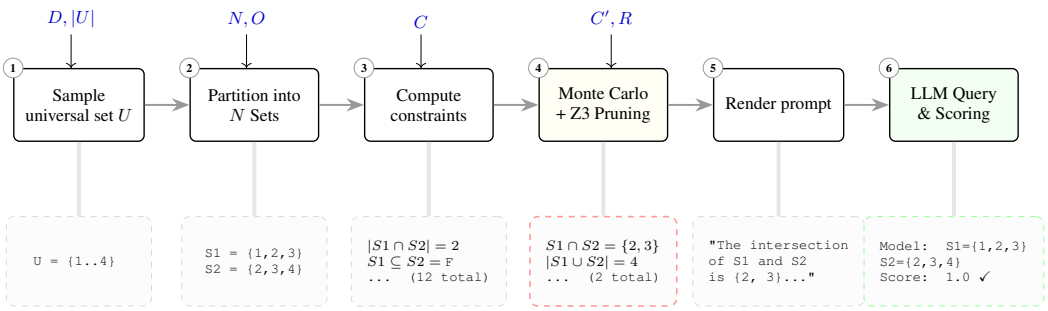


Figure 1: In CUSP, instance sampling and verification precede prompting and evaluation. Given a domain D (numbers or words), we sample a universal set U, partition it into N sets with overlap O, construct $C'_0 = C$ of $12 \times \binom{N}{2}$ constraints, and iteratively reduce C' toward the boundary of unique solvability, keeping each step that still admits exactly one solution. The main `Assignment` datasets sweep the retained fraction of these subsets; a derived track shrinks a subset to an *exactly-minimal* core in which deleting any single constraint is tested to destroy uniqueness. Step 5 ("render prompt") is deterministic: fixed Jinja templates, fixed per-operator verbalizations, no language model in the loop.

autoregressive models because some constraints require guess-and-check over a growing context window. As language models are increasingly deployed in high-stakes settings (Anthropic PBC v. U.S. Department of War, 2026), their evaluations should be multi-faceted and set a high bar.

We use CUSP to demonstrate the following. On the assignment task, accuracy depends strongly on instance parameters that would be incidental if models were genuinely reasoning: models do worse where the Monte Carlo search retains a smaller fraction of constraints, and worse again when the *distribution* of constraints demands structural inference rather than surface reading. Pulling the pragmatic lever, a uniqueness cue interacts with the answer space the prompt makes available: on the same instances, asserting uniqueness is worth -0.4 pp when an assignment is required and $+7.8$ pp when the prompt also permits a "no unique solution" response, while refusals on uniquely-solvable problems fall by 10.0 pp. On the classification task, most models fail to detect whether a problem has multiple solutions — solving a problem correctly does not entail knowing whether its solution is unique. Datasets and code are available, anonymized for review, at Harvard Dataverse and anonymous GitHub.

2 RELATED WORK

As the analytical capabilities of language models have increased, the community has built benchmarks for math (Cobbe et al., 2021; Hendrycks et al., 2021b), formal proof and entailment (Azerbayev et al., 2023; Tafjord et al., 2021), code generation (Chen et al., 2021), arithmetic over text (Dua et al., 2019), planning (Valmeekam et al., 2024), abstract pattern completion (Chollet, 2019; Chollet et al., 2024; 2025; Shi et al., 2024), and general capabilities (Wang et al., 2018; Hendrycks et al., 2021a; Srivastava et al., 2022). These studies typically report a single accuracy number that can be read as an indicator either of principled reasoning or of surface-level pattern matching. The distinction is not hypothetical: models lean on syntactic heuristics and annotation artifacts (McCoy et al., 2019; Gururangan et al., 2018), learn shortcut paths in place of the intended computation (Liu et al., 2022), exhibit symbolic brittleness (Welleck et al., 2022), fail compositional generalization when supporting local structure is absent (Bogin et al., 2022; Dziri et al., 2023), run into the practical limits of autoregression (Lin et al., 2021), and produce chain-of-thought (CoT) traces that are often unfaithful to the computation performed (Lanham et al., 2023; Turpin et al., 2023; Li et al., 2024; Chang, 2026). A benchmark that cannot distinguish principled from surface-level behavior is the wrong instrument for measuring operational reasoning.

Prior work addresses this gap in several ways. Parameterized benchmarks vary problem contents systematically to measure consistency (Anil et al., 2022; Liu et al., 2026) and are arguably less susceptible to contamination; human-baselined and inductive-search designs tie scores to a reference for what the task actually requires (Johnson et al., 2021; Wang et al., 2024); and a closer line of

work uses set operations themselves as a probe, building on a long tradition that casts inference as manipulation of sets and concepts (Boole, 1854; Pawlak, 1982; Fong & Spivak, 2019; Akhbari et al., 2024). The gap can also be addressed through prompting rather than benchmark redesign: chain-of-thought (Wei et al., 2022), scratchpads (Nye et al., 2021), analogical prompting (Yasunaga et al., 2024), thought propagation (Yu et al., 2024), Markov-property formulations (Bohde et al., 2024), and logic-guided pipelines (Cao et al., 2024) all improve accuracy, but none of them tells us whether the improvement is reasoning or more elaborate pattern matching.

We add both in CUSP. On the problem complexity side, a solver-verified minimal sufficient constraint set gives us three levers that make shortcut exploitation measurable: constraint *retention* (R) tests whether accuracy tracks logical sufficiency alone or also the *quantity* of constraints in the prompt; the distribution of constraint *types* separates models that read answers off the surface of a constraint ("$A \cap B = \{2, 3\}$" names the elements common to A and B) from models that reason about its structure ("$|A \cap B| = 2$" does not name elements); and universal-set *scaling* tells us whether frontier-model ceilings reflect genuine or merely bounded mastery. On the pragmatic side, we vary a single feature of the prompt — whether it implies that a unique solution exists — and complement the assignment task with a classification task. Together these choices make visible what a single accuracy number hides: whether a model abstains when it should, or answers with unfounded confidence when it should not (Wen et al., 2024; Kirichenko et al., 2025; Kalai et al., 2025); how it responds to what a prompt implies rather than what it states (Singh, 2025; Su & Zhang, 2025; Chun et al., 2026); and, because the uniqueness cue and the available answer space vary independently, what it computes as against what it commits to.

3 BENCHMARK

In our benchmark, reasoning is reasoning about sets. Grounding reasoning in set theory gives us precise, verifiable control over whether a problem is solvable and what shortcuts might be available for heuristic reasoning. Problem instances are created by sampling the entities of a world (a universal set from some domain), constructing premises (constraints on sets), sampling constraints up to and beyond the boundary of solvability, and testing along the way how many, if any, solutions exist (Figure 1).

There are *two tasks* — assignment and classification — that we evaluate separately. The *assignment task* is to identify the members of every set such that all constraints are simultaneously satisfied. The *classification task* is, given a task, determine how many solutions there are, if any. What differs between tasks is the instructions presented to the language model; the problem instances remain the same.

3.1 PROBLEM INSTANCES

Concretely, given $U = \{1, \ldots, 4\}$, two set identifiers S_1, S_2, and the constraints $S_1 \cap S_2 = \{2, 3\}$ and $S_1 \setminus S_2 = \{1\}$, the model should recover from the constraints that $S_1 = \{1, 2, 3\}$ and $S_2 = \{2, 3, 4\}$. A problem instance is thus a constraint satisfaction problem (CSP). Formally, an instance is parameterized by a universal set U of elements from some domain D ($U \in D \in \{\text{numbers}, \text{words}\}$), number N of sets, overlap O among sets, and a set of constraints on the sets themselves (C'). A constraint is expressed as the value V of a binary operation $f(S_i, S_j) = V$ over sets S_i, S_j. The type of V can be a set (via operations $\cap$, $\cup$, $\setminus$, or $\triangle$), an integer (via the cardinality of $\cap$, $\cup$, $\setminus$, or $\triangle$), or a boolean (via $\subset$ or $\supset$). $C' \subseteq C$, where C is the complete set of binary constraints over all pairs of sets (enumerated with worked examples in Appendix B, Table 6). Depending on the constraints in C', the number of solutions to a problem instance may be multiple, single, or null. A *solution* is any assignment of the elements of U to the N sets that satisfies every constraint in C', and "uniquely solvable" means that exactly one such assignment exists. We do not assume — in the prompt, the solver, or the scorer — that the sets are disjoint, that they partition U, or that they contain only elements forced by the constraints; uniqueness is therefore defined over exactly the constraints shown to the model. Appendix E reports an independent two-solver audit of this property on every uniqueness-claiming dataset.

To find C', we conduct an iterative Monte Carlo search on subsets of C. Initially, at step 0, $C'_0 = C$ and at any subsequent step $C'_i \subset C'_{i-1}$. The fraction of constraints, R_i, retained at any step i is

$|C'_i|/|C|$. We check the solvability Σ of the problem at step i by running a CSP solver with constraints C'_i and counting the number of solutions. Unless otherwise noted, R denotes $|C'_i|/|C|$ of the last Monte Carlo step i in which the constraints are sufficient for the solution to be unique. A subset that satisfies this property — a unique solution under C', but multiple if *any* constraint is removed — is a *minimal sufficient subset*, and we use that term only for the derived split in which every retained constraint has been deletion-tested. The main `Assignment` datasets are instead a *retention sweep* of uniquely-solvable subsets along the pruning trajectory: they span R up to 1.0 and average 59.2 constraints per instance, so they are uniquely solvable but not minimal. Retention distributions and constraint-type frequencies are in Appendix D.

3.2 METRICS

For the assignment task, we report accuracy at three scopes of decreasing difficulty: the *task* scope (Acc_{task}) rigidly assigns 0 or 1 to a task solution; at the *set* scope (Acc_{set}), partial credit is allotted as the average exact-match accuracy over a task's N sets; and *member* scope ($\text{Acc}_{\text{member}}$) gives partial credit via per-set Jaccard similarity. Unless otherwise noted, accuracy reported in this paper is Acc_{set}.

$$\text{Acc}_{\text{task}} = \mathbb{1}\big[\forall i : \hat{S}_i = S_i^*\big], \quad \text{Acc}_{\text{set}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[\hat{S}_i = S_i^*], \quad \text{Acc}_{\text{member}} = \frac{1}{N}\sum_{i=1}^{N} \frac{|\hat{S}_i \cap S_i^*|}{|\hat{S}_i \cup S_i^*|} \quad (1)$$

For the classification task, we report simple accuracy.

Additionally, we use *partition-level* accuracy (defined Appendix C using rules of inference defined in Table 7) to measure the extent to which model errors are related to rules of inference that a human might use when performing a task and *cardinality preservation* and *duplicate set rate* for diagnostics (see §L). These additional metrics are defined in Appendix C.

4 EXPERIMENTAL SETUP

We now describe the models, datasets, and prompts used in our evaluation. Aggregate accuracy is stable across repeated runs: the run-to-run standard deviation is 0.2–1.2 pp across four models spanning two providers, although individual instances do flip between runs, so we report aggregate stability only and treat close cross-model gaps as ties (Appendix F.8).

We evaluate fourteen models from seven model families across eight datasets (Table 1). All runs use temperature 0.0 and are single shot. Model access is through provider APIs (Gemini, Bedrock, OpenAI); latencies are in Appendix P and resolved model IDs, inference settings, and provenance limits in Appendix O. Prompts are rendered deterministically — fixed Jinja templates filled with fixed per-operator verbalizations, so no language model takes part in writing a prompt (representative prompts and responses in Appendix A) — and the maximum output token cap is 8,192 for the main `Assignment` and `Classification` runs, 16,384 for the $|U|$-scaling and hard-cell pools. Unless otherwise stated, we append a chain-of-thought instruction. For the assignment task, we run two prompt variants that differ in *three* respects, which we separate experimentally in Appendix J.1: `Implicit` asks the model to "solve for" the elements of the sets and accepts only an assignment, while `Explicit` (i) uses the verb "determine", (ii) appends the uniqueness cue "The constraints may or may not uniquely determine a solution", and (iii) enlarges the answer space by accepting `NO_UNIQUE_SOLUTION` in place of an assignment (full text in Appendix A.5). The cue is truthful for every instance in the sense that the constraints *do* determine a solution, but the two prompts are not logically equivalent as instructions, because they do not admit the same set of responses.

Native reasoning — an internal "thinking" budget or reasoning-effort setting, available for a subset of the models — is studied separately for two models (Sonnet 4, Gemini 2.5 Flash) to isolate its contribution from prompt-level CoT; unlike the eight main evaluations, that study drops the prompt-level CoT instruction and spans multiple reasoning conditions per model (Appendix F.5). The main evaluations otherwise use each provider's default inference settings, which for some systems means substantially different realized test-time compute; Appendix F.6 reruns two systems at corresponding nominal effort labels and shows that one cross-model ordering in Table 2 does not survive that

Table 1: The datasets in CUSP. All are drawn from the core pool (§3); $\texttt{Assignment}_{|U|+}$ and $\texttt{Assignment}^{\text{Hard}}_{|U|+}$ are drawn from the extended scaling pool. The `Assignment` rows are retention sweeps of uniquely-solvable systems; the exactly-minimal cores derived from them (mean 7.0 constraints, every constraint deletion-tested) are the split used for the prompt experiments in Appendix J.1. For brevity, we sometimes refer to these datasets by their subscripts in subsequent sections.

Dataset	Instances	Description
`Assignment`	900	Balanced over $N \in \{3,5,7\}$, $\lvert U\rvert \in \{8,12,16,20\}$, $O \in \{0.0, 0.5\}$, domains {numbers, words}, and R.
$\texttt{Assignment}_{\text{Value}}$	760	$\geq$70% value constraints.
$\texttt{Assignment}_{\text{Size}}$	472	$\geq$70% size constraints.
$\texttt{Assignment}_{\text{Hard}}$	100	Hardest regime: N=5, O=0.5, size-heavy, $R \leq 0.2$.
$\texttt{Assignment}_{\lvert U\rvert}$	480	Varies $\lvert U\rvert \in \{8,12,16,20\}$ at fixed N=5.
$\texttt{Assignment}_{\lvert U\rvert+}$	324	Extended sweep: $\lvert U\rvert \in \{30,\ldots,250\}$, $N \in \{3,5,7\}$, $\approx$80% of instances at $R \leq 0.2$.
$\texttt{Assignment}^{\text{Hard}}_{\lvert U\rvert+}$	229	Hard slice of $\texttt{Assignment}_{\lvert U\rvert+}$: $R \leq 0.15$ and $\leq$50% value constraints.
$\texttt{Assignment}_{\text{Deceptive}}$	450	225 paired cf 0.0 / cf 0.5 word instances.
`Classification`	2,616	891 `UNIQUE` + 825 `MULTIPLE`-solution + 900 `UNSATISFIABLE` (built by injecting a contradictory twin of an existing value constraint; audit in Appendix J.4).

control. The main table should therefore be read as a comparison of provider-default *systems*, not of compute-matched capabilities.

5 RESULTS

On the `Assignment` dataset, models score substantially lower under `Explicit` than under `Implicit` prompting (Table 2). Pragmatics matter — but this particular contrast overstates them, and we do not read it as a prompt-only effect: the two conditions also draw on different constraint regimes ($\approx$59 constraints per instance for `Implicit` versus $\approx$7 for the `Explicit` rows), and the `Explicit` prompt differs from `Implicit` in three ways at once (§4). Holding the instance, the constraint set, and the verb fixed and moving one factor at a time (Appendix J.1), the uniqueness cue moves Acc_{set} by -0.4 pp (95% CI $[-1.1, +0.4]$) when an assignment is required and by $+7.8$ pp $[+6.9, +8.8]$ once `NO_UNIQUE_SOLUTION` is available, with refusals falling 10.0 pp; the interaction is $+8.2$ pp $[+7.0, +9.4]$. The pragmatic lever therefore acts on whether a model commits to an assignment, not on the assignment it computes, and the refusals it removes are spurious because every instance admits exactly one solution. The `Value` and `Size` datasets show a second effect: although every retained instance in both is verified uniquely solvable, models do much better when C' is dominated by value constraints and much worse when it is dominated by size constraints (Table 2). Every model loses 4.4–32.8 pp on size-heavy constraint sets — a range covering all fifteen systems scored on these datasets, including the auxiliary Llama 3.3 70B of Appendix O, and computed after the uniqueness audit of Appendix E — consistent with a failure to run the guess-and-check procedure that size constraints force. Mistral Large and Qwen 3 32B trail the frontier by roughly 37–57 pp at Acc_{set} under the `Implicit` prompt but respond to both levers in the same direction as stronger models, which we read as capacity-limited rather than lever-insensitive.

Other features of the benchmark also correlate with degraded accuracy: set-level accuracy falls as overlap O increases (Figure 10) and as constraint retention R decreases (Figure 2). Every retained instance is verified uniquely solvable under the prompt-visible semantics (§3.1; the $\leq$1% of rows that failed the audit are excluded from scoring, Appendix E), so lower retention does not mean a less determined problem. The dependence on retention is nevertheless an *association* rather than a demonstrated causal effect of constraint quantity: retention is nearly collinear with raw constraint count, and controlling for solver work reduces but does not eliminate the coefficient. A randomized intervention does show that added material must be informative — appending new true constraints raises accuracy by $+10.5$ pp $[+8.4, +12.5]$, whereas repeating present constraints or padding with unrelated text does not — but retention and constraint count move together too closely, and our

Table 2: Accuracy on `Assignment` varies along both levers: the pragmatic lever (whether the prompt says a unique solution may or may not exist) and the composition of C'. *The Δ column of the `Explicit` block is not a prompt-only contrast*: those rows use the derived minimal cores ($\approx$7 constraints) while the `Implicit` columns use the retention sweep ($\approx$59), and the prompts differ in verb, uniqueness cue, and answer space. On identical constraint sets the same-prompt-pair drop is 5.3–22.0 pp for the six models run on both (decomposition in Appendix J.1). GPT-5.4 Mini and Nano refuse on 100.0% and 99.2% of `Explicit` rows (Table 24), so their drop columns measure abstention, not reasoning. Rows grouped by family, best metric per family in bold; metrics in §3.2.

	`Implicit prompt`			`Explicit prompt`		`Implicit prompt`		
Model	Acc_{task}	Acc_{set}	$\text{Acc}_{\text{member}}$	Acc_{set}	Δ (pp)	`Value`	`Size`	Δ(pp)
o4-mini	**97.0**	**97.4**	**97.8**	**67.2**	−30.2	**92.3**	**79.2**	+13.1
GPT-5.4	43.1	67.4	84.3	50.1	−17.3	59.7	42.6	+17.0
GPT-5.4 Mini	20.7	42.2	66.6	0.0	−42.2	31.0	14.1	+16.9
GPT-5.4 Nano	27.4	36.9	60.7	0.5	−36.5	53.6	39.3	+14.3
Opus 4.6	93.3	**96.9**	**98.9**	**71.3**	−25.6	**97.5**	**92.9**	+4.6
Sonnet 4.6	**93.8**	96.7	98.5	66.0	−30.7	97.2	92.7	+4.6
Sonnet 4.5	91.0	95.1	98.0	77.7	−17.4	94.8	89.6	+5.2
Sonnet 4	85.6	92.1	96.7	73.5	−18.6	92.6	83.1	+9.5
Haiku 4.5	67.8	82.2	91.7	55.7	−26.5	83.0	67.4	+15.6
Gemini 2.5 Flash	**54.4**	**55.0**	**55.3**	**45.3**	−9.7	59.7	**54.0**	+5.8
Gemini 2.5 Flash-Lite	41.8	42.1	42.5	35.6	−6.5	**65.4**	41.5	+23.9
Mistral Large	17.0	39.6	63.7	12.5	−27.1	45.0	27.8	+17.1
Qwen 3 32B	41.0	59.6	73.8	42.9	−16.7	58.2	31.5	+26.7

hardness controls remain too coarse, to say whether models key on the quantity of constraints or on some other facet of difficulty it carries. *We therefore withdraw the stronger reading that models exploit constraint quantity rather than logical sufficiency.* Retention is a property of the base instance rather than of the pruning seed (ICC 0.89 across ten independently ordered prune draws, with identical model rankings in all ten), whereas the value/size composition of any single subset is largely sampler-induced (ICC 0.15), so we treat retention-conditioned results and rankings as robust and single-subset type composition as sampler-dependent. Controls and pruning-order checks are in Appendix H; the value/size gap is stratified in Appendix G.

Table 3: Accuracy drops on `Hard` under both prompts; Δ is the difference from `Assignment` under the same prompt. Under `Explicit`, Opus and Sonnet 4.6 are *more* accurate on `Hard`.

	`Implicit`		`Explicit`	
Model	Hard	Δ (pp)	Hard	Δ (pp)
Opus 4.6	88.2	−8.7	76.2	+4.9
Sonnet 4.6	89.8	−6.9	84.6	+18.6
Sonnet 4	61.6	−30.5	57.4	−22.9
Haiku 4.5	57.4	−24.8	52.8	−5.1

Table 4: `Hard` dataset results under the `Implicit` prompt reported on a per-$|U|$ basis.

	Per-$\lvert U\rvert$			
Model	$\lvert U\rvert$=8	$\lvert U\rvert$=12	$\lvert U\rvert$=16	$\lvert U\rvert$=20
Opus 4.6	95.2	89.6	87.2	80.8
Sonnet 4.6	93.6	94.4	89.6	81.6
Sonnet 4	78.4	56.8	58.4	52.8
Haiku 4.5	69.6	73.6	41.6	44.8

The gap between frontier and mid-tier models widens on the `Hard` subset, and the two tiers fail in different ways (Table 3). Opus 4.6 and Sonnet 4.6 never refuse, and their errors concentrate at $|U|$=20, where the combinatorial search space is largest (Table 4). Haiku 4.5, by contrast, refuses 16% of problems outright even under the `Implicit` prompt, which we read as a token-budget capacity limit.

Under the `Explicit` prompt, Sonnet 4.6 scores 84.6% on `Hard` but only 66.0% on the full `Assignment` dataset — a +18.6 pp advantage on the harder subset. The `Explicit`-prompt failure mode on `Assignment` is predominantly over-refusal on *easy* unique problems, where alternative assignments look consistent even though only one satisfies all constraints jointly. The

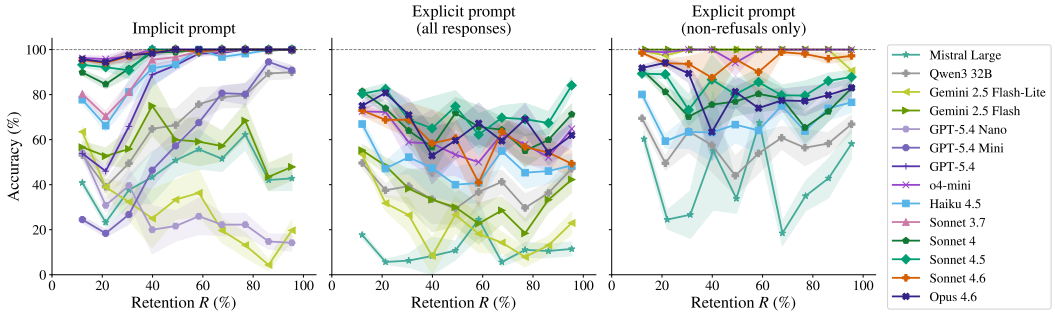


Figure 2: Models perform worse on uniquely-solvable problem instances with a lower constraint retention rate (R) (fewer retained constraints, lower accuracy; see Appendix H for the controls behind this association). Performance degradation is clear when the task is presented with the `Explicit` prompt, even when we do not penalize the model for refusing to respond. X axis shows the retention rate R. Y axis is Acc_{set}. (Shaded bands: $\pm$SEM.)

`Hard` subset has tight constraint structures with little room for spurious alternatives, so frontier models answer rather than refuse.

Partition-level accuracy (Table 9 in Appendix F.3; rules of inference in Table 7) shows that models fail not at deriving what the constraints imply but at assembling those inferences into a consistent assignment. Second-order partitions saturate at 98.2–100% for all four models and the first-order *must*/*cannot* partitions stay above 92%, so every model identifies which elements individual constraints force in or out. The discriminating partition is *can*: elements whose membership no single constraint determines but the full constraint set does, so the model must try assignments and check them. Opus 4.6 and Sonnet 4.6 score 96.8% and 97.4% there, while Sonnet 4 and Haiku 4.5 drop to 86.0% and 87.0% — the gap on *can* is where Sonnet 4.6 improves over Sonnet 4.

The error taxonomy corroborates this reading. Over-inclusion and partial displacement dominate set-level errors (Figure 19), and 98.8% of the extra elements belong to *another* target set rather than being hallucinated (Figure 22). Models shuffle *can* elements between plausible homes instead of settling on one globally consistent assignment.

Is the high accuracy of frontier models on `Assignment` an artifact of the bounded universe, where a 20-element universal set leaves the combinatorial search space small? Extending $|U|$ from 30 to 250 breaks the ceiling for every model, and the rate of loss separates two solving strategies (Figure 16, Appendix I): a model that enumerates candidate assignments should collapse as $|U|$ grows, since the subset count explodes, while a model that reasons over the constraints should be largely indifferent to $|U|$, since the constraint count is fixed. Haiku 4.5 loses 34.6 pp (91% to 56%), consistent with the search-heavy prediction; Sonnet 4 and Sonnet 4.6 lose 16.8 and 15.2 pp, primarily propagating but falling back on search once $|U|$ exceeds 100; Opus 4.6 loses only 12.2 pp (100% to 88%), a residual that suggests even Opus searches for the *can* elements (Table 9). The hard slice widens the gap: on $\texttt{Assignment}^{\text{Hard}}_{|U|+}$, Opus 4.6 and Sonnet 4.6 lose 5.1 and 2.3 pp against their $\texttt{Assignment}_{|U|+}$ scores, Sonnet 4 loses 10.5 pp and Haiku 4.5 loses 23.1 pp (Appendix K.1, Table 25). Across all problem-complexity factors ($|U|$, R, O, and the size-constraint share of C') the direction of each effect on Acc_{set} is consistent across models, as Table 5 summarizes.

We report the classification task first as a binary `UNIQUE`-versus-`MULTIPLE` decision, which drops the `UNSATISFIABLE` class and tests the central distinction without the known surface shortcut. On the 1,716 instances of the two remaining classes the best model reaches only 78.6% (Opus 4.6; o4-mini 75.7%, Sonnet 4.6 74.3%) against a 51.9% majority-class baseline, and several models sit at or below that baseline (Appendix J). The three-way result (Figure 3) is a secondary diagnostic, because accuracy on the `UNSATISFIABLE` class is compatible with a surface-level rule rather than with solvability reasoning: a rule that flags any instance containing the injected value-twin (§4) recovers that class with 100% accuracy and six of seven models score within 1 pp of it (Appendix J.4, Table 22), and its errors track the size of the injected constraint value (Table 23), which solvability reasoning would be indifferent to. The rule predicts `UNIQUE` for every `MULTIPLE`-solution instance, so it cannot explain why models differ on that class. We accordingly limit the claim to the binary

distinction: a matched batch of unsatisfiable instances whose contradictions require deduction from non-repeating constraints is specified but not yet built, and is scoped for a future release of CUSP.

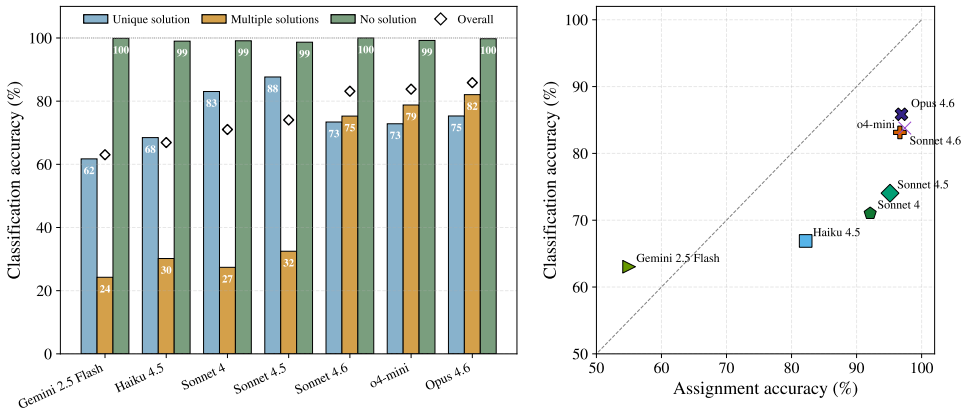


Figure 3: Assignment accuracy does not imply `Classification` competence. Seven models label each instance `UNIQUE`, `MULTIPLE`, or `UNSATISFIABLE` under an `Explicit` prompt; the `UNIQUE` instances are the solver-verified problems of the `Assignment` dataset. **(a)** per-class accuracy, diamond marking each model's overall score: `MULTIPLE` is where capability separates, while `UNSATISFIABLE` acts as a floor because our injected contradictions are surface-detectable. **(b)** `Assignment` versus `Classification` accuracy with the $y{=}x$ line; models below the diagonal are overrated by the `Assignment` evaluation's unique-solution cue, a dimension that discriminates within the frontier cluster `Assignment` ties. Confusion matrices in Appendix J.

Lastly, native reasoning improves accuracy wherever we could control it (§4; Appendix F.5, Figure 14: Sonnet 4 +7.1 pp, Gemini 2.5 Flash +21.2 pp), with diminishing returns per reasoning token, but cross-model rankings depend on the inference configuration: under corresponding nominal effort labels GPT-5.4 meets or exceeds o4-mini, reversing their provider-default ordering in Table 2 (Appendix F.6). Meanwhile, a training-free prompt-optimization approach (CePO) helps the weaker of two open-source models by +18.9 to +33.7 pp across three datasets while the stronger gains at most +3.5 pp (Appendix F.7).

Table 5: Factors X measured on `Assignment` to have a deleterious effect on accuracy. Each row names a factor X and shows the directional effect of a directional change in the factor.

Factor X	ΔX	$\Delta\text{Acc}_{\text{set}}$	Evidence
$\lvert U\rvert$	↑	↓	Fig. 16
R	↓	↓	Fig. 2
O (overlap)	↑	↓	Fig. 10
% size constraints in C'	↑	↓	Table 2

6 LIMITATIONS

Benchmark design. Elements are drawn from integers, random English words, and WordNet hyponym groups; semantic effects across these domains are small ($\leq$1.3 pp pooled, non-significant after correction; Appendix F.4), and non-English, code-like, and ontology-structured domains are left to future releases of CUSP. We calibrate difficulty against machine reference points: an exact SMT solver solves 100% of instances and a no-search local-propagation heuristic solves 33.6% of the `Assignment` sweep and 2.6% of the exactly-minimal cores (Appendix M). We report no human pass rate, since ground truth here is defined by the solver, and make no claim about human-level difficulty or human–model parity. On the classification task, `UNSATISFIABLE` instances carry an injected value-twin and are therefore reachable by a surface rule that matches six of seven models on that class (Table 22); the discriminating signal comes from the multi-solution class, which is why we

report the binary UNIQUE-versus-MULTIPLE decision as the primary classification result and the three-way task as a diagnostic.

Methodology. The pragmatic effect of §5 is an abstention-policy effect; its training-level cause is untested, and we claim no link between alignment training and set reasoning. We read visible chains of thought only as corroborative text (240 double-annotated traces, Appendix N), which cannot establish the procedure a model ran. Our evaluation is single-shot and chain-of-thought prompted, so our claims cover single-shot deductive set-constraint reasoning — including the long-context bookkeeping that protocol imposes — and not operational reasoning in general; length alone does not explain the failures (accuracy rises slightly with constraint count, $\rho = 0.11$, and with prompt length, $\rho = 0.09$; near-cap truncation is at most 0.7% for frontier reasoning models). Models are accessed through provider APIs, so we cannot verify reproducibility against open-weight checkpoints, and some historical runs predate unified provenance logging (Appendix O).

7 CONCLUSION

Intelligent systems that are instructed via natural language should be capable of reasoning robustly about the objects of the world. The system may refuse to perform a task when it is unsure about the task, and when it *does* perform the task, the quality of its performance should be consistent. The results we have presented here indicate that frontier models perform the assignment task reliably, but only up to a point. Both levers expose fragility: with the problem complexity lever, accuracy suffers when the constraint set is small but logically sufficient, when overlap among sets is large, and when the constraints require a guess-and-check approach; with the pragmatics lever, models abstain on problems they have in fact solved as soon as the prompt offers them that option, and a one-line assertion of uniqueness takes much of it back.

Accuracy on CUSP should thus be interpreted with care: the behaviors we report indicate a proclivity to read answers off the surface of value constraints, and a sensitivity to how many constraints a prompt carries that logical sufficiency alone does not predict, rather than uniformly genuine reasoning — consistent with findings elsewhere (Dziri et al., 2023).

Knowing *how* to solve a problem and knowing *that* it admits a unique solution are separable capabilities. Frontier models that cluster on the assignment task separate on the classification task, where weaker models collapse on the multi-solution class while saturating on the unsatisfiable one, and models that cannot recognize non-uniqueness answer where they should decline. Evaluations of language models as operational reasoners should measure both whether the answer is correct and whether the model knew it was the only correct answer; we offer CUSP as an instrument for doing so at the boundary of solvability.

AI USE STATEMENT

Generative AI tools were used to assist with literature discovery, manuscript drafting and editing, and the preparation of illustrative figures. The authors reviewed and verified the cited literature, experimental results, numerical values, and scientific claims, and take full responsibility for the content of the submission.

REFERENCES

Bardiya Akhbari, Manish Gawali, and Nicholas A. Dronen. Setlexsem challenge: Using set operations to evaluate the lexical and semantic robustness of language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 50381–50400. Curran Associates, Inc., 2024. doi: 10.52202/079017-1596. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/5a12909ffd7145c41139ad66ecf20fc0-Paper-Datasets_and_Benchmarks_Track.pdf.

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring Length Generalization in Large Language Models. In *Advances in Neural Information Processing Systems*, 2022.

Anthropic PBC v. U.S. Department of War. Docket no. 3:26-cv-01996 (n.d. cal.). CourtListener, 2026. URL https://www.courtlistener.com/docket/72379655/anthropic-pbc-v-us-department-of-war/. Accessed: May 5, 2026.

Zhangir Azerbayev, Bartosz Piotrowski, Hailey Schoelkopf, Edward W. Ayers, Dragomir Radev, and Jeremy Avigad. Proofnet: Autoformalizing and formally proving undergraduate-level mathematics. *arXiv*, 2023. doi: 10.48550/arxiv.2302.12433.

Ben Bogin, Shivanshu Gupta, and Jonathan Berant. Unobserved Local Structures Make Compositional Generalization Hard. In *Conference on Empirical Methods in Natural Language Processing*, 2022.

Montgomery Bohde, Meng Liu, Alexandra Saxton, and Shuiwang Ji. On the markov property of neural algorithmic reasoning: Analyses and methods. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Kn7tWhuetn.

George Boole. *An Investigation of the Laws of Thought on Which are Founded the Mathematical Theories of Logic and Probabilities*. Walton and Maberly, London, 1854. doi: 10.1017/CBO9780511693090. Reprinted by Cambridge University Press, 2011.

Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Frederick Jelinek, Robert L Mercer, and Paul S Roossin. A statistical approach to french/english language translation. In *Proceedings of the 2nd International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages*, pp. 71–86, 1988.

Chengzhi Cao, Yinghao Fu, Sheng Xu, Ruimao Zhang, and Shuang Li. Enhancing human-AI collaboration through logic-guided reasoning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=TWC4gLoAxY.

Cerebras Systems. CePO: Empowering Llama with reasoning using test-time compute. https://www.cerebras.ai/blog/cepo, 2024. Accessed: 2026-05-03.

Edward Y. Chang. Right for the wrong reasons: Epistemic regret minimization for causal rung collapse in llms, 2026. URL https://arxiv.org/abs/2602.11675.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv:2107.03374*, 2021.

François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.

Francois Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. Arc prize 2024: Technical report. *arXiv preprint arXiv:2412.04604*, 2024.

Francois Chollet, Mike Knoop, Gregory Kamradt, Bryan Landers, and Henry Pinkard. Arc-agi-2: A new challenge for frontier ai reasoning systems. *arXiv preprint arXiv:2505.11831*, 2025.

Jon Chun, Hannah Sussman, Adrian Mangine, Murathan Kocaman, Kirill Sidorko, Abhigya Koirala, Andre McCloud, Gwen Eisenbeis, Wisdom Akanwe, Moustapha Gassama, Eliezer Gonzalez Chirinos, Anne-Duncan Enright, Peter Dunson, Tiffanie Ng, Anna von Rosenstiel, and Godwin Idowu. Cei: A benchmark for evaluating pragmatic reasoning in language models, 2026. URL https://arxiv.org/abs/2603.09993.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv:2110.14168*, 2021.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning Over Paragraphs. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. Faith and Fate: Limits of Transformers on Compositionality. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Brendan Fong and David I. Spivak. *An Invitation to Applied Category Theory: Seven Sketches in Compositionality*. Cambridge University Press, Cambridge, 2019. ISBN 978-1-108-48229-5. doi: 10.1017/9781108668804.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014. URL https://arxiv.org/abs/1410.5401.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A Smith. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 107–112, 2018.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset. In *Advances in Neural Information Processing Systems*, 2021b.

Aysja Johnson, Wai Keen Vong, Brenden M Lake, and Todd M Gureckis. Fast and flexible: Human program induction in abstract reasoning tasks. *arXiv preprint arXiv:2103.05823*, 2021.

Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala, and Evan Zhang. Why language models hallucinate. *arXiv preprint arXiv:2509.04664*, 2025. Argues theoretically that the tendency to respond over-confidently is a byproduct of training objectives that do not adequately distinguish between facts and plausible-sounding guesses.

Polina Kirichenko, Mark Ibrahim, Kamalika Chaudhuri, and Samuel J. Bell. Abstentionbench: Reasoning llms fail on unanswerable questions. *arXiv preprint arXiv:2506.09038*, 2025. Evaluates state-of-the-art reasoning models on their inability to stay silent when presented with logically unanswerable or factually impossible prompts.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė Lukošiūtė, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, Saurav Kadavath, Shannon Yang, Thomas Henighan, Timothy Maxwell, Timothy Telleen-Lawton, Tristan Hume, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. Measuring faithfulness in chain-of-thought reasoning, 2023. URL https://arxiv.org/abs/2307.13702.

Bangzheng Li, Ben Zhou, Fei Wang, Xingyu Fu, Dan Roth, and Muhao Chen. Deceptive semantic shortcuts on reasoning chains: How far can models go without hallucination? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7675–7688, 2024.

Chu-Cheng Lin, Aaron Jaech, Xin Li, Matthew R. Gormley, and Jason Eisner. Limitations of Autoregressive Models and Their Alternatives. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2021.

Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. *arXiv:2210.10749*, 2022.

Junlin Liu, Shengnan An, Shuang Zhou, Dan Ma, Shixiong Luo, Ying Xie, Yuan Zhang, Wenling Yuan, Yifan Zhou, Xiaoyu Li, Ziwen Wang, Xuezhi Cao, and Xunliang Cai. General365: Benchmarking general reasoning in large language models across diverse and challenging tasks, 2026. URL https://arxiv.org/abs/2604.11778.

Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3428–3448, 2019.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv:2112.00114*, 2021.

Zdzisław Pawlak. Rough sets. *International Journal of Computer & Information Sciences*, 11(5): 341–356, 1982. doi: 10.1007/BF01001956.

Fan Shi, Bin Li, and Xiangyang Xue. Towards generative abstract reasoning: Completing raven's progressive matrix via rule abstraction and selection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=IcRlOOFzxm.

M. P. et al. Singh. Gricean norms as a basis for effective collaboration. *AAMAS Proceedings*, 2025. URL https://www.csc2.ncsu.edu/faculty/mpsingh/papers/societal/AAMAS-25-Gricean.pdf.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, and Amanda Dsouza. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2022. doi: 10.48550/arxiv.2206.04615.

Ruili Su and Yanfei Zhang. Conversational implicature: a diachronic cognitive pragmatic approach. *Humanities and Social Sciences Communications*, 12(1), 2025. URL https://ideas.repec.org/a/pal/palcom/v12y2025i1d10.1057_s41599-025-05248-2.html.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. Proofwriter: Generating implications and proofs over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3624–3634, 2021. doi: 10.18653/v1/2021.findings-acl.317.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 74952–74965. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/ed3fea9033a80fea1376299fa7863f4a-Paper-Conference.pdf.

Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Planbench: An extensible benchmark for evaluating large language models on planning and reasoning about change. *Advances in Neural Information Processing Systems*, 36, 2024.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2018.

Ruocheng Wang, Eric Zelikman, Gabriel Poesia, Yewen Pu, Nick Haber, and Noah Goodman. Hypothesis search: Inductive reasoning with language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=G7UtIGQmjm.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019. doi: 10.1162/tacl_a_00290. URL https://direct.mit.edu/tacl/article/7/625/43528/Neural-Network-Acceptability-Judgments.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 2022.

Sean Welleck, Peter West, Jize Cao, and Yejin Choi. Symbolic brittleness in sequence models: on systematic generalization in symbolic mathematics. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

Bo Wen, Jifan Yao, Shuo Feng, Chengrun Xu, Yulia Tsvetkov, Bill Howe, and Lucy Lu Wang. Know your limits: A survey of abstention in large language models. *arXiv preprint arXiv:2407.18418*, 2024. Provides a formal framework and taxonomy for why models fail to abstain when faced with unanswerable or ambiguous queries.

Michihiro Yasunaga, Xinyun Chen, Yujia Li, Panupong Pasupat, Jure Leskovec, Percy Liang, Ed H. Chi, and Denny Zhou. Large language models as analogical reasoners. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=AgDICX1h50.

Junchi Yu, Ran He, and Zhitao Ying. THOUGHT PROPAGATION: AN ANALOGICAL APPROACH TO COMPLEX REASONING WITH LARGE LANGUAGE MODELS. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=SBoRhRCzM3.

A EXAMPLE PROMPTS AND MODEL RESPONSES

A.1 CLAUDE OPUS 4.6, N=3, NUMBERS, LOW RETENTION — CORRECT

Problem `numbers_U12_N3_o0.00_s22_i96`; $|U|$=12, r=0.14 (5/36 constraints). **Ground truth** $A = \{1, 8, 15, 28\}$, $B = \{3, 11, 22, 23\}$, $C = \{2, 20, 25, 31\}$.

Prompt

You are given a universal set with elements 1, 2, 3, 8, 11, 15, 20, 22, 23, 25, 28, 31.
You are given the following constraints on sets:

- The difference between C and B is equal to the set {2, 20, 25, 31}
- The union of A and B is equal to the set {1, 3, 8, 11, 15, 22, 23, 28}
- The difference between B and A is equal to the set {3, 11, 22, 23}
- The intersection of B and C contains exactly 0 elements
- The symmetric difference of A and B is equal to the set {1, 3, 8, 11, 15, 22, 23, 28}

Use the given universal set and constraints to solve for the elements of each of the sets.
Think step-by-step before answering.
Provide the solution in XML tags `<solution></solution>`. Place each set's solution its own line, exactly like this example: `<solution>A = {1, 2, 3}
B = {2, 3, 4}</solution>` Do not use any other XML tags inside the solution.

Model Response (verbatim)

I need to find sets A, B, and C.
Step 1: Find A and B using the constraints.
From the symmetric difference of A and B: A △ B = {1, 3, 8, 11, 15, 22, 23, 28}
From the union of A and B: A ∪ B = {1, 3, 8, 11, 15, 22, 23, 28}
The symmetric difference equals (A ∪ B) minus (A ∩ B). Since A △ B = A ∪ B, this means A ∩ B = ∅.
So A and B are disjoint.
From the difference B \ A = {3, 11, 22, 23}. Since A and B are disjoint, B \ A = B, so B = {3, 11, 22, 23}.
From A ∪ B = {1, 3, 8, 11, 15, 22, 23, 28} and B = {3, 11, 22, 23}: A = {1, 8, 15, 28}
Step 2: Verify A and B.

- A ∩ B = ∅ ✓
- A ∪ B = {1, 3, 8, 11, 15, 22, 23, 28} ✓
- B \ A = {3, 11, 22, 23} ✓
- A △ B = {1, 3, 8, 11, 15, 22, 23, 28} ✓

Step 3: Find C.
From B ∩ C = ∅: C contains none of {3, 11, 22, 23}.
From C \ B = {2, 20, 25, 31}: These elements are in C but not in B. Since B ∩ C = ∅, all elements of C are not in B, so C \ B = C.
Therefore C = {2, 20, 25, 31}.
Verify: C ∩ B = {2, 20, 25, 31} ∩ {3, 11, 22, 23} = ∅ ✓

```
<solution>A = {1, 8, 15, 28}
B = {3, 11, 22, 23}
C = {2, 20, 25, 31}</solution>
```

A.2 CLAUDE OPUS 4.6, N=3, NUMBERS, LOW RETENTION — INCORRECT

Problem `numbers_U12_N3_o0.00_s15_i84`; $|U|$=12, r=0.11 (4/36 constraints). **Ground truth** $A = \{0, 2, 13, 33\}$, $B = \{7, 10, 29, 34\}$, $C = \{1, 4, 21, 25\}$.

Prompt

You are given a universal set with elements 0, 1, 2, 4, 7, 10, 13, 21, 25, 29, 33, 34.
You are given the following constraints on sets:

- The difference between A and B is equal to the set {0, 2, 13, 33}
- The union of A and B contains exactly 8 elements
- The symmetric difference of A and C is equal to the set {0, 1, 2, 4, 13, 21, 25, 33}
- The symmetric difference of B and C contains exactly 8 elements

Use the given universal set and constraints to solve for the elements of each of the sets.
Think step-by step before answering.
Provide the solution in XML tags `<solution></solution>`. Place each set's solution its own line, exactly like this example: `<solution>A = {1, 2, 3}
B = {2, 3, 4}</solution>` Do not use any other XML tags inside the solution.

Model Response (verbatim)

I need to find sets A, B, and C given the constraints.
Constraint 1: $A - B = \{0, 2, 13, 33\}$ (elements in A but not in B)
Constraint 2: $A \cup B$ has exactly 8 elements
Constraint 3: $A \triangle C = \{0, 1, 2, 4, 13, 21, 25, 33\}$
Constraint 4: $B \triangle C$ has exactly 8 elements
Step 1: Analyze A and B. From Constraint 1, $\{0, 2, 13, 33\} \subseteq A$ and $\{0, 2, 13, 33\} \cap B = \emptyset$. $A \cup B = (A - B) \cup B = \{0, 2, 13, 33\} \cup B$, and $|A \cup B| = 8$. So $|B| + 4 = 8$, meaning $|B| = 4$. Also, $A = \{0, 2, 13, 33\} \cup (A \cap B)$.
Step 2: Analyze A and C using Constraint 3. $\{0, 2, 13, 33\} \subseteq A$, so these must be in $A - C$ (in A but not C). The remaining elements of $A \triangle C$ are $\{1, 4, 21, 25\}$, which must be in $C - A$. So $\{1, 4, 21, 25\} \subseteq C$ and $\{1, 4, 21, 25\} \cap A = \emptyset$.
Step 3: Determine B. B has 4 elements, none from $\{0, 2, 13, 33\}$. $B \subseteq \{1, 4, 7, 10, 21, 25, 29, 34\}$. Elements of $A \cap B$ are also in A but not in $\{1, 4, 21, 25\}$ (since those aren't in A). So $A \cap B \subseteq \{7, 10, 29, 34\}$.
Step 4: Use Constraint 4. Let me try $S = \{7, 10, 29, 34\}$ (so $A \cap B$ has 4 elements, $B = \{7, 10, 29, 34\}$). $C = \{1, 4, 21, 25\} \cup T$ where $T \subseteq \{7, 10, 29, 34\}$. $B \triangle C = (\{7, 10, 29, 34\} \triangle T) \cup \{1, 4, 21, 25\}$. Size $= |\{7, 10, 29, 34\} \triangle T| + 4 = 8$, so $|\{7, 10, 29, 34\} \triangle T| = 4$, meaning $T = \emptyset$ or $T = \{7, 10, 29, 34\}$.
If $T = \emptyset$: $C = \{1, 4, 21, 25\}$, $B \triangle C = \{1, 4, 7, 10, 21, 25, 29, 34\}$ — 8 elements. ✓
$A = \{0, 2, 7, 10, 13, 29, 33, 34\}$, $B = \{7, 10, 29, 34\}$, $C = \{1, 4, 21, 25\}$.

```
<solution>A = {0, 2, 7, 10, 13, 29, 33, 34}
B = {7, 10, 29, 34}
C = {1, 4, 21, 25}</solution>
```

The model outputs $A \cap B = \{7, 10, 29, 34\}$ in Step 4 without ruling out $A \cap B = \emptyset$, the ground truth. B and C are correct; A is over-included by the four shared-with-B elements. Set-level accuracy is $2/3$ (over-inclusion on A, §L).

A.3 CLAUDE OPUS 4.6, N=7, WORDS, HIGH RETENTION — CORRECT

Problem `words_U8_N7_o0.00_s34_i67`; $|U|$=8, r=0.96 (242/252 constraints).

Prompt

You are given a universal set with elements Elaps, ampliative, anargyros, indignant, papyrographer, predismissory, unproportionateness, zoonomist.
You are given the following constraints on sets:

- The intersection of B and C contains exactly 0 elements
- The symmetric difference of D and F is equal to the set {unproportionateness, zoonomist}
- The difference between E and A is equal to the set {Elaps}

⋮ *(239 additional constraints omitted for brevity)*

Model Response (verbatim)

I'll work through this step-by-step, using key constraints to determine each set.
Finding A:

- A ∪ C = {ampliative, indignant, papyrographer}
- A ∩ C = { }
- A − C = {indignant, papyrographer}
- C − A = {ampliative}

So A = {indignant, papyrographer} and C = {ampliative}
Finding E: E − A = {Elaps}, A ∩ E = { }, so E = {Elaps}
Finding B: B ∪ E = {Elaps, predismissory}, B ∩ E = { }, so B = {predismissory}
Finding D: D ∪ E = {Elaps, unproportionateness}, D ∩ E = { }, so D = {unproportionateness}
Finding F: D ∩ F = { }, D ⊕ F = {unproportionateness, zoonomist}, so F = {zoonomist}
Finding G: F ∪ G = {anargyros, zoonomist}, F ∩ G = { }, so G = {anargyros}

```
<solution>A = {indignant, papyrographer}
B = {predismissory}
C = {ampliative}
D = {unproportionateness}
E = {Elaps}
F = {zoonomist}
G = {anargyros}</solution>
```

A.4 Claude Sonnet 4.6, N=5, words, mid retention — **Correct**

Problem `words_U20_N5_o0.50_s21_i85`; $|U|$=20, r=0.55 (66/120 constraints), overlap 0.5.

Prompt

You are given a universal set with elements Trisotropis, acatholic, aftershaft, corocleisis, dacryolith, dogmatist, endocyst, frogland, machopolyp, mackereler, niggler, nonconnotative, orthoceracone, overbrightly, panspermatism, relighter, shrug, triakisicosahedron, unberufen, unpropitious.
You are given the following constraints on sets:

- The intersection of B and E is equal to the set { }
- The difference between E and B contains exactly 0 elements
- The union of B and D contains exactly 11 elements

⋮ *(63 additional constraints omitted for brevity)*

Model Response (verbatim)

I'll work through this step-by-step.
From the constraints, I can extract direct set definitions:
Finding E: E − A = {acatholic, aftershaft, dacryolith, endocyst, triakisicosahedron, unberufen}, A ∩ E = {machopolyp}. So E = {acatholic, aftershaft, dacryolith, endocyst, machopolyp, triakisicosahedron, unberufen}
Finding A: A ∩ E = {machopolyp}, A − E has 3 elements. A − E = {corocleisis, mackereler, shrug}. So A = {corocleisis, machopolyp, mackereler, shrug}
Finding B: A ∪ B = {corocleisis, frogland, machopolyp, mackereler, nonconnotative, relighter, shrug, unpropitious}, A ∩ B = {shrug}, B − A = {frogland, nonconnotative, relighter, unpropitious}. So B = {frogland, nonconnotative, relighter, shrug, unpropitious}
Finding D: D − B = {dacryolith, mackereler, orthoceracone, overbrightly, panspermatism, triakisicosahedron}, B ∩ D = {}. So D = {dacryolith, mackereler, orthoceracone, overbrightly, panspermatism, triakisicosahedron}
Finding C: C − A = {Trisotropis, dogmatist, niggler, unberufen, unpropitious}, A ∩ C = {corocleisis, machopolyp, shrug}, C ∩ E = {machopolyp, unberufen}. So C = {Trisotropis, corocleisis, dogmatist, machopolyp, niggler, shrug, unberufen, unpropitious}

```
<solution>A = {corocleisis, machopolyp, mackereler, shrug}
B = {frogland, nonconnotative, relighter, shrug,
unpropitious}
C = {Trisotropis, corocleisis, dogmatist, machopolyp,
niggler, shrug, unberufen, unpropitious}
D = {dacryolith, mackereler, orthoceracone, overbrightly,
panspermatism, triakisicosahedron}
E = {acatholic, aftershaft, dacryolith, endocyst, machopolyp,
triakisicosahedron, unberufen}</solution>
```

A.5 CLAUDE SONNET 4.6, PAIRED PROMPT FRAMING: IMPLICIT VS. EXPLICIT

Both prompts below are rendered from the same instance and the same constraint list. The instructions are *not* logically equivalent: `Explicit` changes the verb ("solve for" → "determine"), adds the uniqueness cue, and enlarges the answer space with `NO_UNIQUE_SOLUTION`. Appendix J.1 separates those three factors experimentally; this pair only illustrates the surface difference between the templates.

Caveat: this instance is a regression test, not evidence of a prompt effect. The instance shown here was *mislabeled* `UNIQUE` by the generator bug described in Appendix E: under exactly the five constraints displayed, four assignments satisfy the system (B may additionally contain 0 and/or 5). The `Explicit` response reproduced below is therefore mathematically *correct*, not a refusal error, and the instance is excluded from every score in this paper. We keep the example because it is now a regression test for the SAT-gated generator, which must label it `MULTIPLE`. The response boxes are annotated with the labels the original (buggy) ground truth assigned, which is exactly the failure this appendix documents.

Problem `numbers_U8_N3_o0.00_s20_i78`; $|U|$=8, N=3, r=0.14 (5/36 constraints). **Original (incorrect) ground truth** $A = \{4, 21, 23\}$, $B = \{3, 8, 10\}$, $C = \{0, 5\}$ — one of the four satisfying assignments, wrongly recorded as the only one.

Shared constraints (verbatim). Universal set $U = \{0,\ 3,\ 4,\ 5,\ 8,\ 10,\ 21,\ 23\}$.

- The intersection of A and C is equal to the set $\{\}$.
- The difference between B and C contains exactly 3 elements.

- The difference between B and C is equal to the set $\{3,\ 8,\ 10\}$.
- The difference between A and C is equal to the set $\{4,\ 21,\ 23\}$.
- The difference between C and A is equal to the set $\{0,\ 5\}$.

`Implicit` prompt — scored Correct (accuracy $= 1.00$) against the mislabeled ground truth.

Prompt instruction (`Implicit`)

Use the given universal set and constraints to solve for the elements of each of the sets.
Think step-by-step before answering.
Provide the solution in XML tags `<solution></solution>`. Place each set's solution on its own line, exactly like this example: `<solution>A = {1, 2, 3}`
`B = {2, 3, 4}</solution>`.

Sonnet 4.6 response (excerpted)

I'll work through this step-by-step.
Analyzing the constraints: 1. $A \cap C = \{\}$, 2. $|B - C| = 3$, 3. $B - C = \{3, 8, 10\}$, 4. $A - C = \{4, 21, 23\}$, 5. $C - A = \{0, 5\}$.
Finding A and C. From (1), A and C are disjoint, so $A - C = A$ and $C - A = C$. Therefore $A = \{4, 21, 23\}$ and $C = \{0, 5\}$.
Finding B. From (3), B contains $\{3, 8, 10\}$ and cannot contain $\{4, 21, 23\}$. B may also contain elements from C (these would not appear in $B - C$), but *no constraint forces* 0 *or* 5 *into* B*, and no constraint forces other elements into* B*, so the minimal solution is* $B = \{3, 8, 10\}$. Constraint (2) is then satisfied with $|B - C| = 3$.
Verification. $B - C = \{3, 8, 10\}$ ($= 3$ elements) ✓.

```
<solution>A = {4, 21, 23}
B = {3, 8, 10}
C = {0, 5}</solution>
```

`Explicit` prompt — scored Incorrect (accuracy $= 0.00$) against the mislabeled ground truth, but correct under the displayed constraints.

Prompt instruction (`Explicit`)

Use the given universal set and constraints to **determine** the elements of each of the sets. **The constraints may or may not uniquely determine a solution.**
Think step-by-step before answering.
If the constraints uniquely determine a solution, provide it in `<solution></solution>`.
If the constraints do not uniquely determine a solution, respond with `<solution>NO_UNIQUE_SOLUTION</solution>` and explain why.

Sonnet 4.6 response (excerpted)

I'll work through this step-by-step.
Finding A and C. From $A \cap C = \{\}$: A and C are disjoint, so $A - C = A$ and $C - A = C$. Therefore $A = \{4, 21, 23\}$ and $C = \{0, 5\}$.
Finding B. $B - C = \{3, 8, 10\}$ means exactly these elements are in B but not in C. So B must contain $\{3, 8, 10\}$, *and may optionally contain* 0 *and/or* 5 (elements of C).
This gives multiple possibilities for B:

$$\{3, 8, 10\},\ \{0, 3, 8, 10\},\ \{3, 5, 8, 10\},\ \{0, 3, 5, 8, 10\}.$$

Constraint 2 ($|B - C| = 3$) is already satisfied by $\{3, 8, 10\}$ in all four cases; it does not narrow things down further.
Since B is not uniquely determined:

<solution>NO_UNIQUE_SOLUTION</solution>
$A = \{4, 21, 23\}$ and $C = \{0, 5\}$ are uniquely determined, but B is not [...] four possible sets for B.

A.6 Example prompt: Classification task

Prompt instruction (Classification)

You are given a universal set with elements ⟨*universal set*⟩.
You are given the following constraints on sets:

- ⟨*constraint 1*⟩
- ⟨*constraint 2*⟩

⋮

Determine the elements of each of the sets from the given universal set and constraints. The constraints may have exactly one solution, multiple solutions, or no solution at all.
Think step-by-step before answering.
Respond with exactly one of the following, inside <solution></solution> tags:

1. If the constraints uniquely determine a solution, provide it — one set per line:
   <solution>A = {1, 2, 3}
   B = {2, 3, 4}</solution>
2. If the constraints are consistent but admit two or more solutions, respond with <solution>MULTIPLE_SOLUTIONS</solution> and briefly explain why.
3. If the constraints are contradictory (no assignment satisfies them), respond with <solution>NO_SOLUTION</solution> and briefly explain which constraints conflict.

Do not use any other XML tags inside the solution.

UNSATISFIABLE instances are built by injecting a contradictory twin of an existing value constraint (two constraints over the same target set with different right-hand sides), which makes the class surface-detectable by scanning for duplicate constraint heads — see Appendix J.4, Table 22.

B Constraints

C Metric definitions

We formalise the metrics from §3.2.

For each target set S_i and each element of U, the inference rules in Appendix B classify the element as *must* ($\mathcal{M}_i$), *cannot* ($\mathcal{X}_i$), or *can* ($\mathcal{C}_i$). First-order partitions follow from a single constraint (e.g. $A \setminus B = V$ forces $V \subseteq \mathcal{M}_A$). Second-order partitions require pairs of constraints (e.g. $A \cup B = V \wedge C \setminus B = W$ forces $V \cap W \subseteq \mathcal{M}_A$). Let $T_i \subseteq U$ be the tier's element set for S_i. Then:

$$\mathrm{Acc}_T = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\left[\hat{S}_i \cap T_i = S_i^* \cap T_i\right].$$

We report $\mathrm{Acc}_{\text{must}}$, $\mathrm{Acc}_{\text{cannot}}$, $\mathrm{Acc}_{\text{can}}$, $\mathrm{Acc}_{\text{2nd-must}}$, and $\mathrm{Acc}_{\text{2nd-cannot}}$, with T_i set to the corresponding tier. The aggregate $\mathrm{Acc}_{\text{2nd}} = \mathrm{Acc}_{\mathcal{M}^{(2)} \cup \mathcal{X}^{(2)}}$ combines both second-order tiers. High $\mathrm{Acc}_{\text{must}}$ with low $\mathrm{Acc}_{\text{can}}$ indicates a model that reads answers from constraints but does not reason under uncertainty.

Implementation note on $\mathrm{Acc}_{\text{cannot}}$. The ground truth for cannot-elements is empty ($S_i^* \cap \mathcal{X}_i = \emptyset$), so $\mathrm{Acc}_{\text{cannot}}$ reduces to a binary check of whether the prediction contains any forbidden element. It

Table 6: We define twelve pairwise constraint types over three categories. All twelve are instantiated for every unordered set pair, so the complete constraint set has $|C| = 12\binom{N}{2}$ members and the evaluated subset C' is selected only by the Monte Carlo pruning, not by a per-type rule. The V column shows the value each constraint type takes for the single illustrative pair $S_i{=}\{1,2,3\},\ S_j{=}\{2,3,4\}$; it is an example of the value, not a generation or selection parameter.

Type	Notation	V
Value	$S_i \cap S_j = V$	$\{2,3\}$
	$S_i \cup S_j = V$	$\{1,2,3,4\}$
	$S_i \setminus S_j = V$	$\{1\}$
	$S_i \triangle S_j = V$	$\{1,4\}$
Size	$\lvert S_i \cap S_j\rvert = V$	2
	$\lvert S_i \cup S_j\rvert = V$	4
	$\lvert S_i \setminus S_j\rvert = V$	1
	$\lvert S_i \triangle S_j\rvert = V$	2
Boolean	$S_i \subset S_j = V$	False
	$S_i \supset S_j = V$	False

Table 7: For diagnostic purposes, we assign elements to *must* ($\mathcal{M}$), *can* ($\mathcal{C}$), or *cannot* ($\mathcal{X}$) using these inference rules. $\mathcal{M}_A \leftarrow V$ denotes $\mathcal{M}_A = \mathcal{M}_A \cup V$.

Constraint(s)	Inference
$A \setminus B = V$	$\mathcal{M}_A \leftarrow V$
$A \cap B = V$	$\mathcal{M}_A \leftarrow V$ $\mathcal{M}_B \leftarrow V$
$A \subseteq B =$ True	$\mathcal{M}_B \leftarrow A$
$A \cup B = V$ $C \setminus B = W$	$\mathcal{M}_A \leftarrow (V \cap W)$
$A \cup B = V$	$\mathcal{C}_A \leftarrow V$ $\mathcal{C}_B \leftarrow V$
$A \setminus B = V$	$\mathcal{X}_B \leftarrow V$
$A \triangle B = V$ $B \cap C = W$	$\mathcal{X}_A \leftarrow (V \cap W)$

awards no partial credit, so $\text{Acc}_{\text{cannot}}$ should be read as a coarse binary diagnostic rather than as a graded score; a fraction-excluded variant is scoped for the next release.

C.1 Error-analysis diagnostics

We use two set-level diagnostics in §L. **Cardinality preservation** $\mathbb{1}[|\hat{S}_i| = |S_i^*|]$ is the share of wrong sets whose size matches the target. **Duplicate-set rate** is the share of incorrect problems with $\hat{S}_i = \hat{S}_j$ for some $i \neq j$.

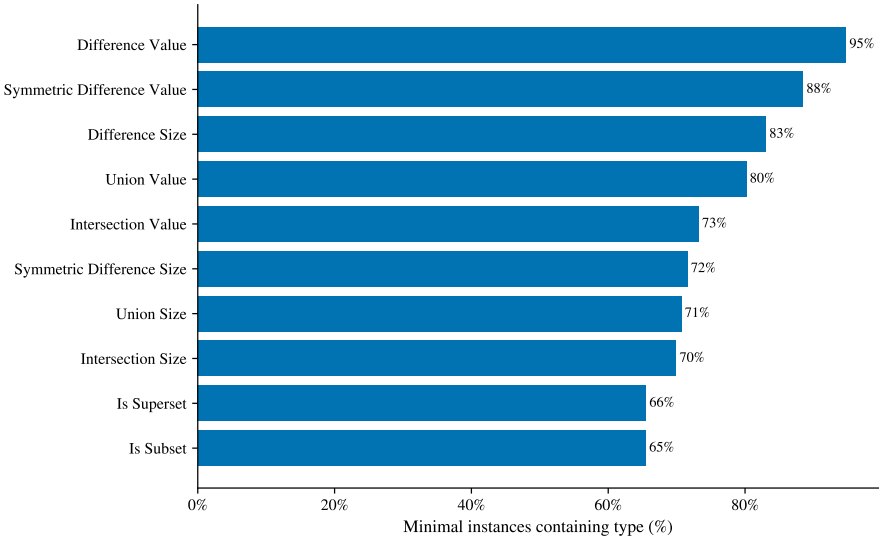
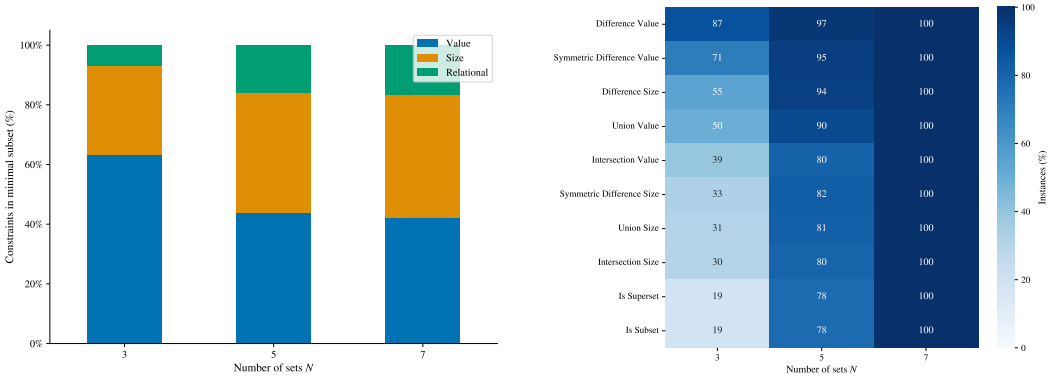


Figure 4: Difference constraints are retained in nearly every minimal sufficient subset; subset and superset relations are retained far less often.



(a) Value constraints dominate at $N{=}3$; size and relational constraints catch up as N grows.

(b) Every constraint type appears in almost every minimal subset at $N{=}7$.

Figure 5: Value constraints dominate at small N; every type becomes structurally required at $N{=}7$.

D Benchmark Characterization

D.1 Constraint-type frequency in minimal sufficient subsets (Figure 4)

D.2 Constraint-type composition by N (Figures 5, 6)

D.3 Retention by N, median (Figure 7)

D.4 Retention distribution by $(N, |U|)$ grid (Figure 8)

D.5 $(N, O) \times |U|$ retention landscape (Figure 9)

E Uniqueness Audit

Uniqueness is the property the benchmark depends on, so we audit it directly. An earlier version of the generator contained an acceptance fast path that ran constraint propagation only — without a

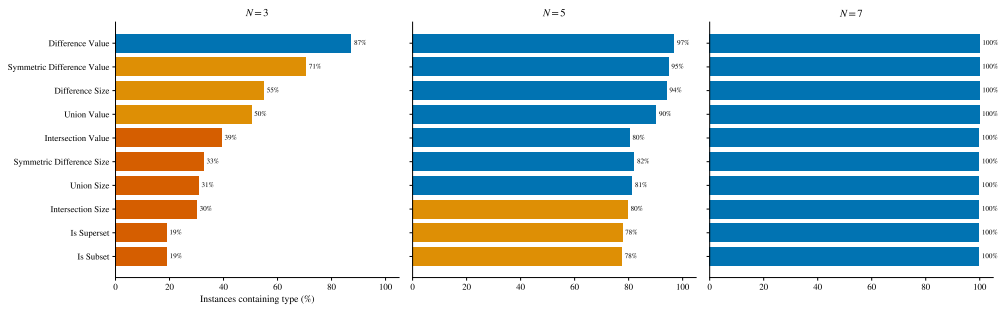


Figure 6: Subset and superset relations become structurally required only at $N \geq 5$; other types are common from $N{=}3$. Bars are faceted by N and shaded at the 80% and 50% retention thresholds.

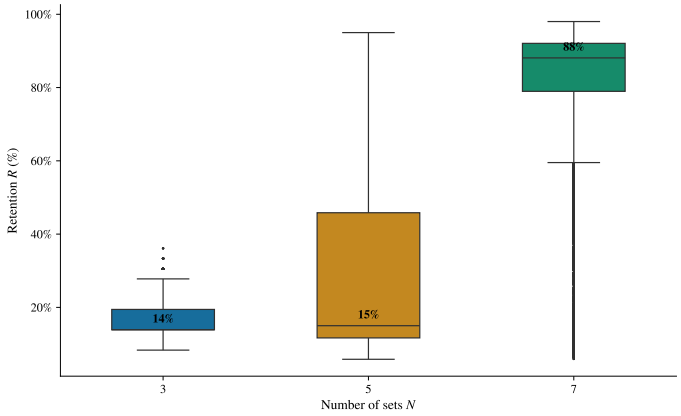


Figure 7: Median retention jumps sharply from $N{=}5$ to $N{=}7$.

second-model SAT check — for $|U| \leq 20$, and could therefore accept a subset as uniquely solvable while a second global assignment still satisfied it. We removed that path (acceptance is now SAT-gated at every universe size) and added the instance of Appendix A.5 as a regression test, which the corrected gate must label `MULTIPLE`.

We then re-audited *every* uniqueness-claiming dataset through two independent solver paths — direct block-and-recheck, and bounded model enumeration — which agree on every row. Failures concentrate at the lowest retention, where an under-specified subset is most likely (Table 8). Because the historical model outputs exist only for the original identifiers, we exclude the affected rows from all scoring in this paper rather than substitute unevaluated replacements: rescoring shifts the main `Assignment` result by at most 0.34 pp, the value-heavy result by at most 1.3 pp, and the deceptive result by at most 0.73 pp, and leaves the frontier ranking unchanged. Matched replacements generated with the corrected gate — preserving domain, $|U|$, N, overlap, and constraint count — restore the original split sizes for future evaluation, and we release the old-to-new identifier mapping, both audit outputs, the offending identifiers, seeds, and the generator commit. As a further check, we generated one fresh accepted instance in each of the 96 configured generation cells and audited all of them through both paths (0/96 failures).

The audit adopts exactly the semantics stated in §3.1: a solution satisfies the listed constraints and nothing more. Neither the solver nor the scorer assumes disjointness, a partition of U, or that a set contains only elements forced by some constraint.

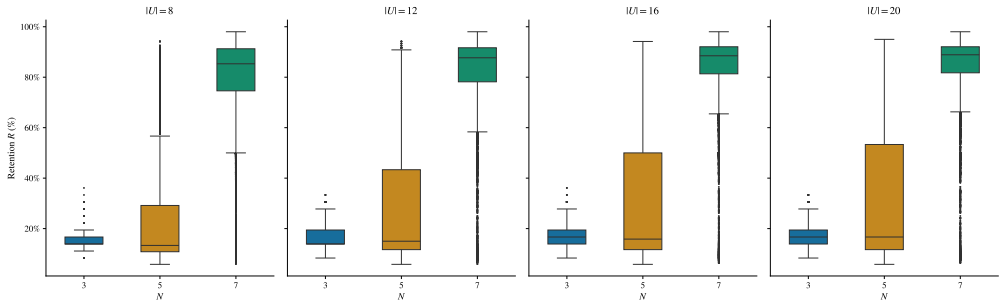


Figure 8: Retention broadens at $N{=}5$ as $|U|$ grows, while $N{=}3$ stays at 10–20% and $N{=}7$ stays near 90%.

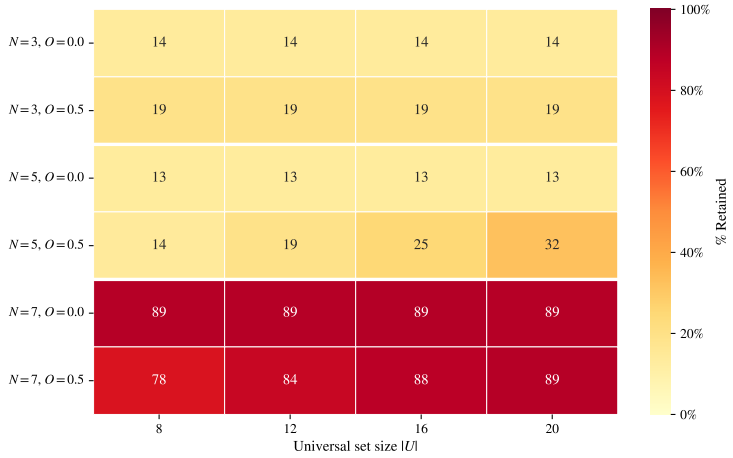


Figure 9: $N{=}5$ with overlap is the only regime where both $|U|$ and O move retention.

F Assignment Evaluation: Additional Results

F.1 Overlap Penalty per Model

Mean accuracy falls 14.0 pp when sets share half their elements. We use $O{=}0.5$ as one of the conditions defining $\texttt{Assignment}_{\text{Hard}}$ (Table 3).

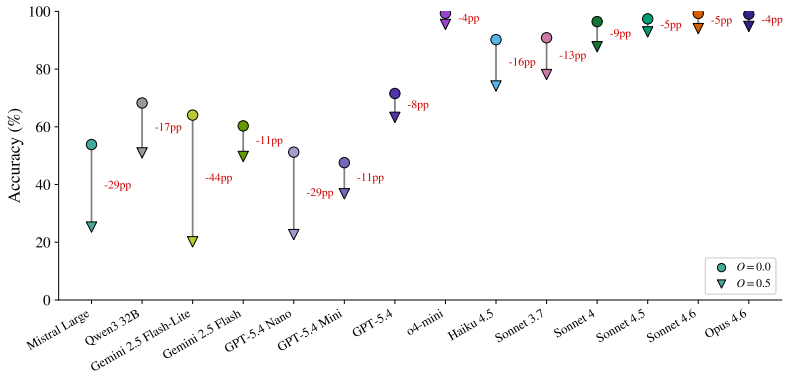


Figure 10: Every model loses accuracy at $O{=}0.5$, with drops from 3.9 pp (o4-mini) to 43.9 pp (Gemini 2.5 Flash Lite).

Table 8: Two independent solver paths agree on every row; failures are $\leq$4.2% and concentrate at low retention. Audited datasets are those that claim uniqueness; the failed rows are excluded from every score reported in this paper.

Dataset	Failed audit	Rate
`Assignment` (900 rows)	9	1.0%
$\texttt{Assignment}_{\lvert U\rvert}$ (480)	1	0.2%
$\texttt{Assignment}_{\text{Value}}$ (760)	32	4.2%
$\texttt{Assignment}_{\text{Deceptive}}$ (450)	10	2.2%
$\texttt{Assignment}_{\text{Size}}$, $\texttt{Assignment}_{\text{Hard}}$, scaling pools, `Classification` `UNIQUE`	0	0.0%

F.2 Per-model accuracy across N and retention (Figure 11)

F.3 Per-model partition-level accuracy

Table 9: Partition-level accuracy on the `Hard` dataset of Table 3, by the inference each element requires: *must*/*cannot* need a single constraint, *can* needs several jointly, *2nd* columns need cross-constraint deduction. All four models reach 98–100% on second-order inference, yet Sonnet 4 and Haiku 4.5 drop to 86% on *can*.

Model	std.	must	cannot	can	2nd	2nd-must	2nd-cann.
Haiku 4.5	57.4	92.2	97.8	87.0	98.2	98.6	99.6
Sonnet 4	61.6	97.6	98.6	86.0	100.0	100.0	100.0
Opus 4.6	88.2	99.2	99.4	96.8	100.0	100.0	100.0
Sonnet 4.6	89.8	99.4	99.4	97.4	100.0	100.0	100.0

Table 10 reports all seven per-problem accuracy columns for the full 14-model `Assignment` evaluation. *Standard* is set-level exact match (Table 2). *Must*, *cannot*, and *can* check, for each element of U, whether the model identifies it as belonging to every consistent solution, to none, or to some-but-not-all. The 2nd-order columns check the constraint rules the model derives.

Table 10: Models reach near-ceiling on element-level partition checks but lose ground on set-level exact match. Columns: set-level exact match (*std.*); first-order *must*, *cannot*, *can*; second-order constraint-rule checks.

Model	**std.**	**must**	**cannot**	**can**	**2nd**	**2nd-must**	**2nd-cann.**
o4-mini	97.4	98.7	99.9	99.1	99.9	99.9	100.0
Opus 4.6	96.9	99.8	99.9	98.9	100.0	100.0	100.0
Sonnet 4.6	96.7	99.5	99.6	98.6	99.8	99.9	99.9
Sonnet 4.5	95.1	99.7	99.7	98.0	99.8	99.8	100.0
Sonnet 4	92.1	99.4	99.6	96.6	99.6	99.8	99.8
Sonnet 3.7	84.5	99.6	99.0	94.7	99.3	99.4	99.7
Haiku 4.5	82.2	98.2	97.6	93.6	99.1	99.4	99.7
GPT-5.4	67.4	97.4	95.5	88.4	98.6	99.0	99.5
Qwen 3 32B	59.6	86.8	94.1	87.6	98.6	99.0	99.6
Gemini 2.5 Flash	55.0	65.8	100.0	89.9	98.9	98.9	100.0
GPT-5.4 Mini	42.2	83.1	85.6	80.5	98.4	98.9	99.3
Gemini 2.5 Flash-Lite	42.1	51.6	99.9	89.3	98.4	98.4	100.0
Mistral Large	39.6	74.5	87.3	83.2	97.8	98.3	99.4
GPT-5.4 Nano	36.9	80.9	71.9	84.2	98.3	99.4	98.9

F.4 Domain effect

Models average 72.3% on numerical domains and 68.0% on word-based domains, a 4.3 pp gap that shrinks to under 1.3 pp at the frontier (Figure 12). The paired deceptive-words evaluation (450 rows, 14 models) shows small, noisy effects: semantic grouping adds $+1.3$ pp ($p = 0.034$, at N=3) and semantic contamination shifts accuracy by -0.86 pp ($p = 0.18$, not significant). The numbers–words

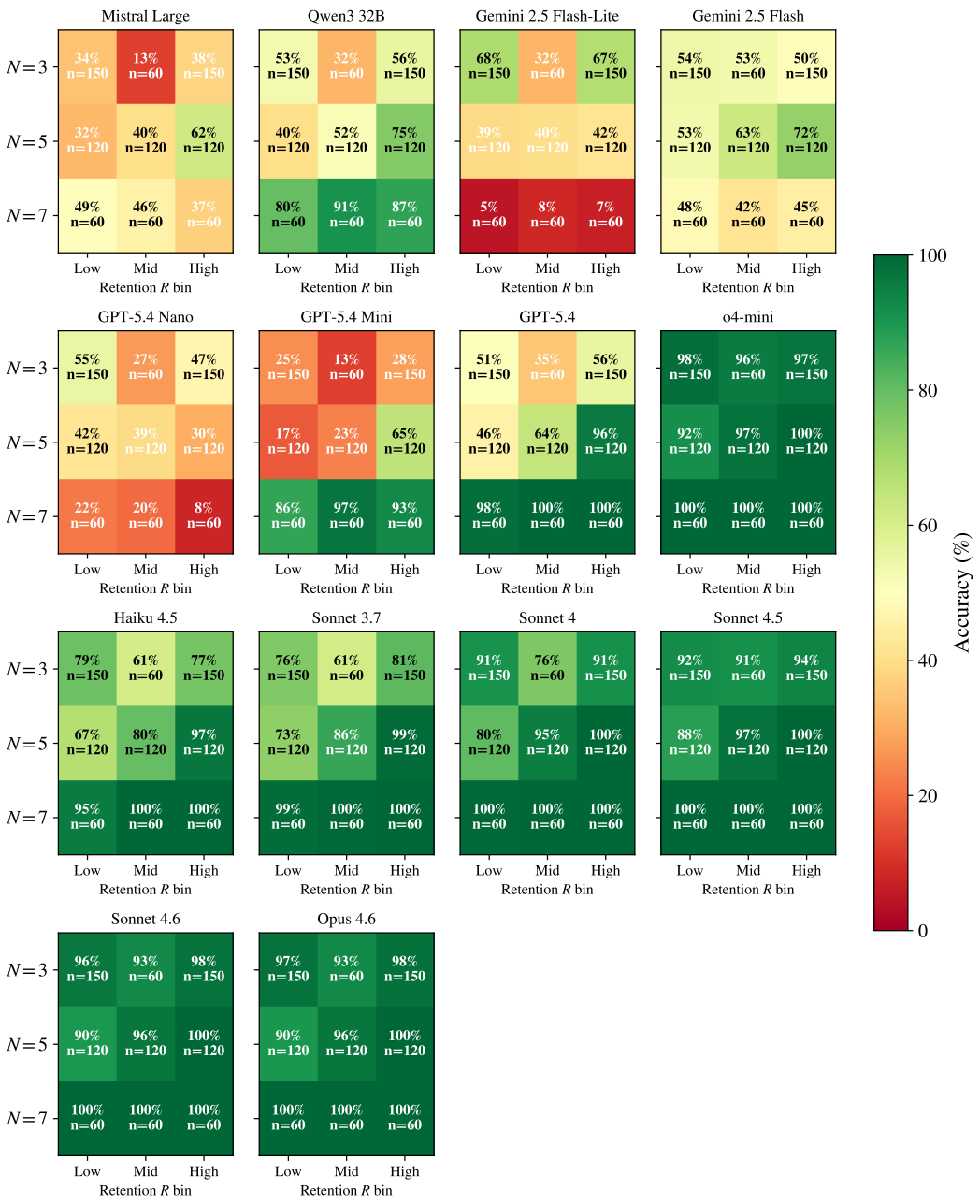


Figure 11: The N=5, low-retention cell produces the widest per-model accuracy range on the `Assignment` dataset and separates the mid-tier most clearly. N=7 instances appear easy because mean retention is 83% at N=7, making those problems effectively over-specified. Each panel shows one model's Acc_{set} in an $N \times R$-bin grid (Low ≤ 0.15 / Mid / High > 0.40), with per-cell sample sizes annotated.

gap peaks at +5.1 pp at $|U|$=16 (Figure 13), well below SetLexSem1's +11.5 pp estimate (Akhbari et al., 2024). We therefore do not treat domain as a diagnostic axis.

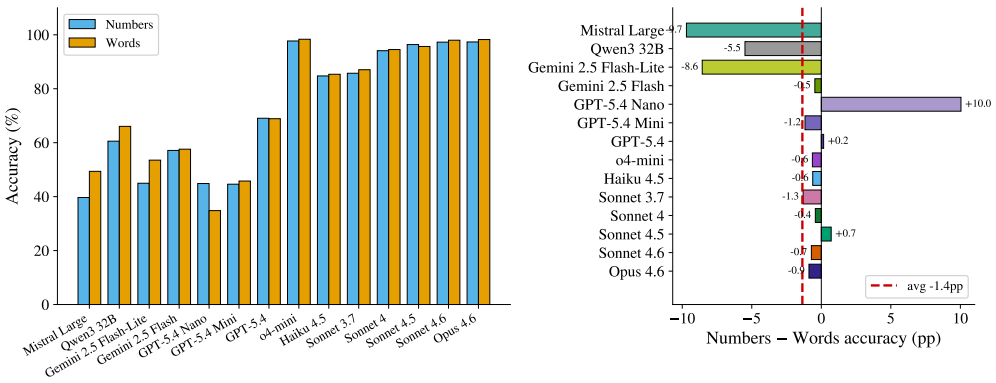


Figure 12: Most models score higher on numerical than on word domains, and the gap shrinks toward the frontier. Left: per-model accuracy on numbers against words (averaged across random, cf 0.0, and cf 0.5 word domains). Right: the numbers−words gap. Frontier models (o4-mini, Opus 4.6, Sonnet 4.6) stay within 1.3 pp; mid-tier and weaker models scatter on both sides of zero.

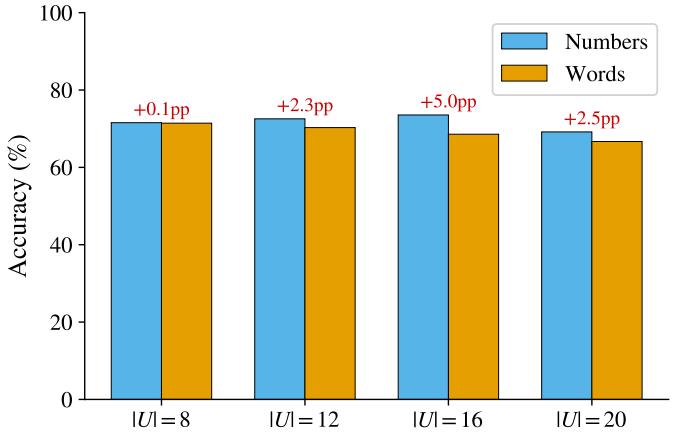


Figure 13: The numbers–words gap peaks at +5.1 pp at $|U|$=16 and narrows by $|U|$=20, well below the +11.5 pp SetLexSem1 estimate. Averaged across 16 models at N=5.

F.5 Native reasoning

We test two models (Sonnet 4 and Gemini 2.5 Flash) under controlled native-thinking conditions, plus one effort check on o4-mini (Table 11). We use this study as a prompt-CoT confound control in §7.

Table 11: Native reasoning raises accuracy for both tested models, with diminishing per-token returns at higher budgets. Δ is the change from the direct baseline.

Model	Condition	Accuracy (%)	Δ (pp)
Sonnet 4	direct	91.3	—
Sonnet 4	1K thinking budget	93.6	+2.3
Sonnet 4	8K thinking budget	98.4	+7.1
Gemini 2.5 Flash	direct	66.6	—
Gemini 2.5 Flash	thinking	87.8	+21.2
o4-mini	default effort	97.4	—
o4-mini	low effort	95.2	−2.2

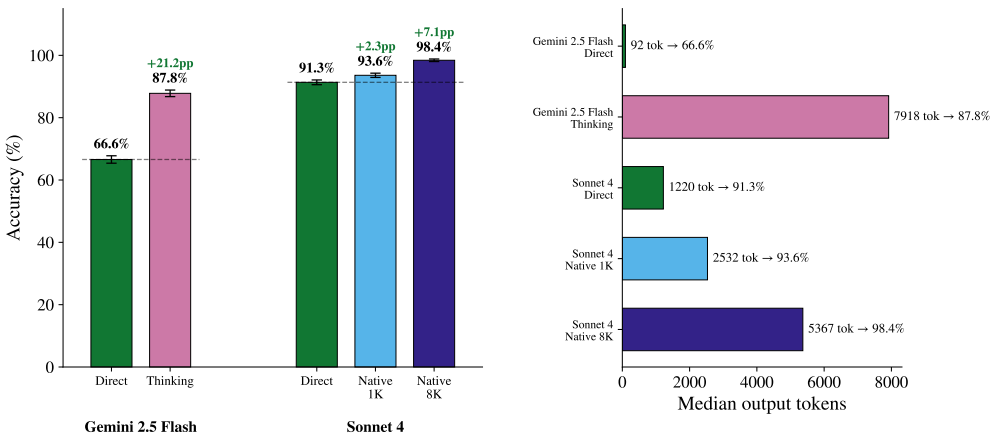


Figure 14: Accuracy gains from native reasoning come at sharply increasing token cost. Left: accuracy by reasoning mode. Right: token cost per condition.

F.6 Matched nominal reasoning effort

The main-table runs use provider defaults, and those defaults do not equalize test-time compute: the main-table GPT-5.4 run used effectively no native reasoning (≈4 reasoning tokens per instance), whereas o4-mini spent thousands. The default-effort ordering in Table 2 therefore cannot be read as a capability comparison. We rerun both systems at corresponding nominal provider effort labels on the 891 audited instances (Table 12). All six cells use temperature 0, an assignment-only prompt, and a 16,384-token cap; o4-mini runs on the direct OpenAI Responses API and GPT-5.4 on Bedrock Mantle's Responses API, accessed 2026-07-26.

Table 12: At every corresponding nominal effort label GPT-5.4 meets or exceeds o4-mini, reversing their provider-default ordering. Acc_{set} counts cap-truncated outputs as incorrect; Acc_{ex} excludes them. Token and latency columns are per-instance means. Reasoning tokens rise monotonically with requested effort for both systems, so the control reaches both provider paths; realized token counts and serving paths nevertheless still differ, which makes this a nominal-effort sensitivity analysis rather than compute matching. These cells are the $|U| \leq 20$ `Assignment` regime, which effort control saturates; it is not the benchmark's headroom (cf. Figure 16 and the binary classification result in Appendix J).

Model / effort	Acc_{set}	Acc_{ex}	Trunc.	Reason. tok.	Output tok.	Lat. (s)
o4-mini / low	92.5	92.5	0.0%	2,085	2,183	12.1
o4-mini / medium	97.6	98.4	0.8%	4,360	4,458	22.8
o4-mini / high	95.5	99.6	4.2%	7,567	7,660	43.4
GPT-5.4 / low	99.6	99.6	0.0%	801	888	6.6
GPT-5.4 / medium	99.9	99.9	0.0%	1,191	1,278	9.2
GPT-5.4 / high	100.0	100.0	0.0%	1,576	1,663	11.2

The historical main runs used an 8,192-token cap and predate unified provenance logging, so exact configured effort and finish reasons are unavailable for some providers (Appendix O).

F.7 Inference-time optimization with CePO

We evaluate Cerebras Planning & Optimization (CePO) (Cerebras Systems, 2024), a test-time framework that adds planning and verification to frozen models, on two open-source models across three datasets (`Assignment`, $\texttt{Assignment}_{\text{Deceptive}}$, $\texttt{Assignment}_{|U|}$). The weaker model gains $+18.9$, $+33.7$, and $+26.9$ pp. The stronger model gains only $+3.5$, -0.6, and $+0.5$ pp. Gains shrink as the base model grows stronger, mirroring the native reasoning study (§F.5).

F.8 RUN-TO-RUN STABILITY (TABLE 13)

Table 13: *Aggregate* run-to-run variance is small relative to the effects we report, but individual instances move. We evaluate the same 100 `Assignment` instances three times each at temperature 0.0 on four models spanning two provider paths (GPT-5.4 via Bedrock Mantle's Responses API at medium effort; Qwen3 32B via a pinned Bedrock ID; the two Anthropic rows are reparsed from the original runs). The largest standard deviation across runs is 1.2 pp. *Disagree* columns give the share of the 100 instances whose set score, respectively whose all-sets-correct outcome, differs across the three runs; these are substantial for mid-tier models, so we claim aggregate stability only and read close cross-model gaps as ties. Qwen3 32B hits the 16,384-token cap on 13–15% of rows per run; on the 71 instances that never cap, its score and outcome disagreement remain 53.5% and 26.8%.

Model	Run 1	Run 2	Run 3	Std (pp)	Range (pp)	Disagree (score)	Disagree (exact)
GPT-5.4	100.0	99.0	100.0	0.6	1.0	1%	1%
Sonnet 4.6	97.0	98.4	98.0	0.7	1.4	11%	9%
Haiku 4.5	80.8	82.8	82.8	1.2	2.0	13%	9%
Qwen3 32B	55.8	55.8	55.4	0.2	0.4	62%	29%

G VALUE VS. SIZE CONSTRAINTS (FIGURE 15, TABLES 14, 15)

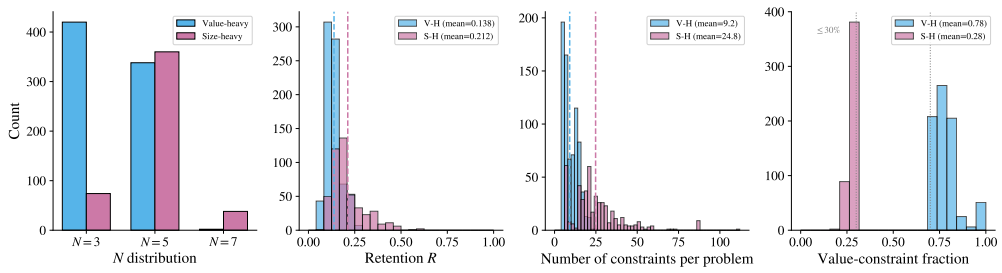


Figure 15: Size-heavy problems have higher retention and more constraints per problem than value-heavy ones, ruling out prompt length as a confound. Panels show N, retention R, constraints per problem, and value-constraint fraction for the value-heavy (760 rows) and size-heavy (472 rows) subsets.

G.1 WITHIN-N BREAKDOWN

We omit N=7 (n_{VH}=2). The N=3 null results for Sonnet 4 and Sonnet 4.6 fall within the size-heavy confidence interval (±10–11 pp at n=74).

Table 14: The value–size gap persists within each N bin and cannot be explained by retention differences. We report accuracy for the four models evaluated on the ablation, stratified by N.

N	**Model**	**Value-heavy**	**Size-heavy**	**Gap**	**Retention (V / S)**
3	Haiku 4.5	86.3%	66.7%	+19.6 pp	
	Sonnet 4	93.3%	93.2%	+0.1 pp	
	Sonnet 4.6	98.3%	98.6%	−0.3 pp	
	Opus 4.6	99.3%	94.6%	+4.7 pp	
5	Haiku 4.5	78.9%	65.9%	+13.0 pp	
	Sonnet 4	91.7%	80.3%	+11.4 pp	
	Sonnet 4.6	95.9%	90.9%	+5.0 pp	
	Opus 4.6	95.3%	91.9%	+3.4 pp	

G.2 WITHIN-CONSTRAINT-COUNT BREAKDOWN

Sample sizes are unbalanced: the 4–7 and 8–13 bins are value-heavy; the 14+ bin is size-heavy. The 8–13 size-heavy cell (n=16, CI $\pm$25 pp) carries little weight.

Table 15: The value–size gap persists after we match on constraint count, ruling out prompt length as a confound. Bin **c** denotes $4 \le c < 8$, $8 \le c < 14$, $c \ge 14$.

Constraints	Model	Value-heavy	Size-heavy	Gap
4–7	Haiku 4.5	86.0%	68.9%	+17.2 pp
	Sonnet 4	93.1%	92.9%	+0.2 pp
	Sonnet 4.6	98.6%	100.0%	−1.4 pp
	Opus 4.6	99.2%	94.0%	+5.2 pp
8–13	Haiku 4.5	82.5%	58.3%	+24.1 pp
	Sonnet 4	93.4%	94.6%	−1.2 pp
	Sonnet 4.6	96.5%	88.8%	+7.8 pp
	Opus 4.6	96.7%	96.7%	+0.0 pp
14+	Haiku 4.5	76.4%	67.5%	+8.9 pp
	Sonnet 4	90.2%	81.1%	+9.0 pp
	Sonnet 4.6	95.1%	91.7%	+3.4 pp
	Opus 4.6	94.8%	92.6%	+2.2 pp

H RETENTION: CONTROLS AND PRUNING-ORDER ROBUSTNESS

Controlling for structure and solver work. On 14,400 model-instance rows with within-cell fixed effects for domain, model, N, $|U|$, and overlap, a unit increase in retention R is associated with $+0.334$ in accuracy. Adding solver-work covariates (Z3 decisions, conflicts, and propagations) lowers the coefficient to $+0.229$ (95% cluster-bootstrap CI $[0.172, 0.281]$), and to $+0.220$ $[0.161, 0.282]$ on the universe-size subset. So the association survives a dynamic-hardness control at about two thirds of its size. We nonetheless decline a causal reading for two reasons that we state rather than imply. First, R is nearly collinear with the raw constraint count (Pearson $r = 0.93$ on the 891 audited instances), so the specification cannot separate "how many constraints" from "what fraction of them was kept." Second, this is regression adjustment, not the matched design that would identify the effect.

Randomized redundancy intervention. To ask whether added constraints help because they inform or because the prompt grows, we took 200 exactly-minimal cores and built three matched longer prompts per core, equal in added length: repeating constraints already present (D), adding distinct true constraints from the full closure (R), and padding with unrelated sentences (P), with five independent R draws per base. Across Qwen3 32B, Haiku 4.5, Opus 4.6, and Sonnet 4.6 (12,800 prompts), only the informative arm helps: $+10.5$ pp $[+8.4, +12.5]$ for R, and no effect for D (-1.5 pp $[-3.3, +0.2]$) or P (-1.1 pp $[-2.9, +0.6]$). Added length alone therefore does not raise accuracy, and mere repetition does not either.

Pruning-order robustness. Because a single Monte Carlo prune order produces one of many admissible subsets, we drew 100 base assignments stratified across the generation grid and took 10 independently ordered prune draws per base, shrinking each to an exactly-minimal core and re-verifying uniqueness with the corrected SAT gate (1,000 deletion-tested subsets). Retention is largely a property of the base instance (ICC 0.89), whereas the value/size composition of the resulting subset is largely prune-order-induced (ICC 0.15). Evaluating four systems on all 1,000 subsets (Table 16), the per-order aggregate range is at most 14.4 pp and the full ranking is identical in all ten orders (Spearman 1.00). Single-instance outcomes are order-sensitive, however: for mid-tier models within-target variance exceeds between-target variance (accuracy ICC 0.06–0.16), so whether a model solves a *specific* base instance depends on which minimal subset it sees. We therefore report retention-conditioned results and model rankings as robust to prune order, and treat single-subset type composition and single-instance outcomes as sampler-dependent.

Table 16: Model rankings are invariant across ten independent prune orders while single-instance outcomes are not. Accuracy is the mean set-exact score over the 1,000 exactly-minimal subsets; the band is the range of the ten per-order aggregate means, each averaging 100 base targets. GPT-5.4 is at ceiling, so its accuracy ICC is uninformative — read its band instead.

Model	Accuracy	Accuracy ICC	Prune-order band (pp)
GPT-5.4 (medium effort)	99.9	—	0.7
Opus 4.6	91.2	0.13	4.6
Haiku 4.5	71.4	0.16	12.1
Qwen3 32B	44.6	0.06	14.4

I $|U|$ SCALING (FIGURE 17, TABLE 17)

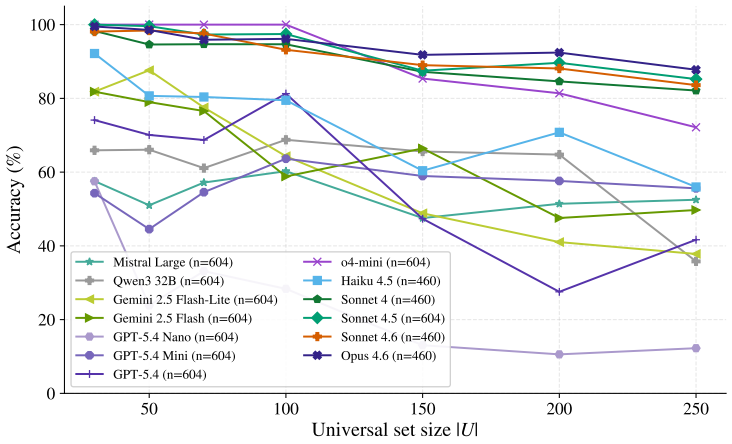


Figure 16: Accuracy vs. $|U|$ at low retention, four Anthropic models, pooled across $N \in \{3, 5, 7\}$, with the output token limit at 16,384 except where a provider caps it at 8,192. Separation holds at $|U|$=250: Opus 4.6 85.1% > Sonnet 4.6 80.0% > Sonnet 4 78.2% > Haiku 4.5 47.0%. Per-$|U|$ breakdowns in Appendix I.

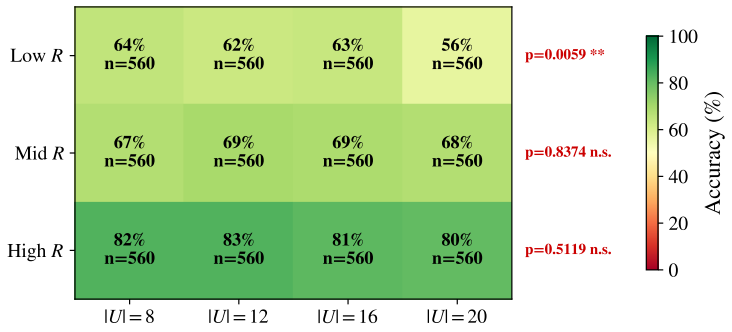


Figure 17: Increasing $|U|$ lowers accuracy only in the low-retention tercile; mid- and high-retention cells stay near ceiling. $|U| \times R$ grid at N=5, pooled across 16 models (`Assignment`$_{|U|}$ subset).

J EXPLICIT PROMPT AND SOLVABILITY DETECTION

J.1 CUE AND ANSWER SPACE: A FACTORIAL DECOMPOSITION

The `Implicit`/`Explicit` contrast of Table 2 moves three things at once (verb, uniqueness cue, answer space) and additionally compares different constraint regimes. We therefore rerun the prompt

Table 17: Accuracy falls as $|U|$ grows for all four Anthropic models, with Haiku 4.5 degrading fastest. `Implicit` prompt, low retention; Haiku 4.5's non-monotonic $|U|$=200 row (68.2%) reflects the $n\approx45$ per-cell sample size.

Model	$\|U\|$=30	$\|U\|$=50	$\|U\|$=70	$\|U\|$=100	$\|U\|$=150	$\|U\|$=200	$\|U\|$=250
Opus 4.6	98.9	97.1	91.3	93.7	90.6	92.2	85.1
Sonnet 4.6	97.0	96.8	98.4	91.1	87.4	85.4	80.0
Sonnet 4	97.2	92.1	95.0	94.4	85.3	82.1	78.2
Haiku 4.5	91.1	75.3	72.3	72.6	54.4	68.2	47.0

manipulation on one fixed instance set — the 891 audited exactly-minimal cores, which are derived from *every* one of the 900 base `Assignment` instances with identifiers preserved, so every cell is paired by construction — and move one factor at a time. We use the minimal cores rather than the parent sweep because a minimal core is where a uniqueness cue could plausibly matter: no redundant constraint remains to fall back on. The verb is held fixed *within* each contrast ("determine" in both cue cells, "solve for" in both answer-space cells), verified mechanically against the template files.

Table 18: The uniqueness cue acts through the answer space, not through the assignment. Each row is a paired contrast on the same 891 instances; entries are the mean of the per-model paired risk differences with 95% intervals from a cluster bootstrap over base instances (10,000 resamples). Weighting by rows instead of by models moves each estimate by at most 0.3 pp. Matched sample sizes are 658–891 after pairwise exclusion of token-cap truncations.

Contrast (same instances, verb fixed)	$\Delta\text{Acc}_{\text{set}}$ (95% CI)	Range (9 models)	ΔRefusal
Uniqueness cue, assignment required	-0.4 $[-1.1, +0.4]$	-2.6 to $+2.6$	0.0 (unavailable)
Uniqueness cue, abstention available	$+7.8$ $[+6.9, +8.8]$	$+0.3$ to $+22.2$	-10.0
Adding `NO_UNIQUE_SOLUTION` under fixed "solve for"	-5.7 $[-6.5, -4.8]$	-13.2 to $+4.8$	$+8.8$
Cue $\times$ answer-space interaction	$+8.2$ $[+7.0, +9.4]$	—	—

With an assignment required, asserting uniqueness is indistinguishable from no effect; with abstention available, the identical wording change is worth $+7.8$ pp and cuts refusals by 10.0 pp. On truncation-excluded Acc_{task} the two cue effects are $+0.9$ and $+7.8$ pp, so the conclusion does not depend on the metric. Since every instance in the set admits exactly one solution, the refusals the cue removes were wrong answers: the gain is not the cue talking models out of warranted caution. The per-model decomposition (Table 19) shows the signature of an abstention effect — accuracy rises almost exactly as refusals fall, while parseable-but-wrong assignments stay nearly flat. In 8.4% of Opus 4.6's refusals its visible trace had already derived every target set correctly (Appendix N).

Definition of refusal. A response is a *refusal* when its `<solution>` block matches `NO_UNIQUE_SOLUTION` (case-insensitive), the sentinel the abstention-enabled templates instruct the model to emit. The test precedes any set-block check, so a hedged response emitting both the sentinel and an assignment counts as a refusal. We track two other outcomes separately: *truncation* (empty response at the reasoning-token cap, excluded pairwise) and *wrong assignment* (parsed but set-inexact). The sentinel is not offered in the assignment-only cells, which is why the refusal change is exactly 0.0 pp there.

Bounding the verb. This design is not a full $2 \times 2 \times 2$: completing it requires four further cells ("solve for" crossed with cue and answer space), roughly 25,000 additional API calls at this grid size, which we scope for a future release. The verb is nevertheless bounded empirically. Exactly one cell pairing in the grid changes the verb, and it changes the cue along with it ("solve for" with no cue versus "determine" with the neutral cue); that joint effect is null, -2.2 to $+2.5$ pp across nine models, with one of nine bootstrap intervals excluding zero (GPT-5.4, -0.7 pp). Since this bounds verb and cue together, it bounds the verb alone.

What we withdraw. Three readings of the original prompt result do not survive this decomposition: that the uniqueness cue degrades set-theoretic reasoning; that the aggregate

Table 19: Per-model effect of asserting rather than withholding uniqueness, with abstention available in both cells (891 paired instances each). Accuracy gains track refusal reductions; wrong-assignment rates move little. The magnitude is decoupled from capability: o4-mini and GPT-5.4 both solve near ceiling yet sit at opposite ends.

Model	$\Delta\text{Acc}_{\text{set}}$ (pp)	ΔRefusal (pp)	ΔWrong assignment (pp)
Sonnet 4.6	$+22.2$	-25.0	$+2.8$
Opus 4.6	$+14.3$	-17.3	$+3.0$
o4-mini	$+11.6$	-11.2	-0.5
Haiku 4.5	$+6.2$	-13.4	$+7.2$
GPT-5.4 Mini	$+5.7$	-6.2	$+0.2$
Sonnet 4	$+5.3$	-9.9	$+4.6$
Gemini 2.5 Flash	$+3.7$	-4.5	$+0.8$
GPT-5.4 Nano	$+1.6$	-2.2	$+1.0$
GPT-5.4	$+0.3$	-0.6	$+0.2$

`Implicit`→`Explicit` drop is a prompt-only effect; and the speculative alignment-training explanation of it. What survives, better supported, is the cue-by-answer-space interaction, its mechanism in abstention policy, the spuriousness of the induced refusals, and all of the structural-complexity results, which this experiment does not touch.

J.2 Binary `UNIQUE` versus `MULTIPLE` classification

We report the classification task primarily as the binary decision between `UNIQUE` and `MULTIPLE`, which drops the surface-detectable `UNSATISFIABLE` class (see the audit in Appendix J.4) and tests the distinction the benchmark is built around. On the 1,716 instances of those two classes, the best model reaches 78.6% (Opus 4.6), followed by o4-mini at 75.7% and Sonnet 4.6 at 74.3%, against a 51.9% majority-class baseline; several models sit at or below the baseline. This is the result we intend the classification task to carry, and it is much weaker than the three-way accuracies below, which the unsatisfiable class inflates.

J.3 Three-way classification: secondary diagnostic

Table 20: On the three-way `Classification` task, models separate on the multi-solution class while saturating on unsatisfiable. We keep this table as a secondary diagnostic: the unsatisfiable class is surface-detectable, so overall accuracy here overstates solvability reasoning. We ask seven models to classify each of the 2,616 instances as *unique*, *multiple*, or *unsatisfiable* under an `Explicit` prompt. Per-class columns report classification accuracy conditional on the true class; Overall is the share of all instances whose predicted class matches. Truncation at the 8,192-token cap counts as incorrect, which makes Overall a floor for o4-mini (256 truncated, 9.8%; rises to 92.9% on finished rows) and Haiku 4.5 (126, 4.8%; rises to 70.2%); the other six models truncate $\leq$2.4% of rows.

Model	Unique solution *(n=891)*	Multiple solutions *(n=825)*	Unsatisfiable *(n=900)*	Overall
Opus 4.6	75.3	82.1	99.8	**85.9**
o4-mini	72.8	78.8	99.2	**83.8**
Sonnet 4.6	73.4	75.3	100.0	**83.1**
Sonnet 4.5	87.7	32.5	98.7	74.0
Sonnet 4	83.1	27.4	99.1	71.0
Haiku 4.5	68.5	30.2	99.0	66.9
Gemini 2.5 Flash	61.7	24.2	99.9	63.0

We break down the solvability-detection results in Table 20 with per-model confusion matrices (Figure 18).

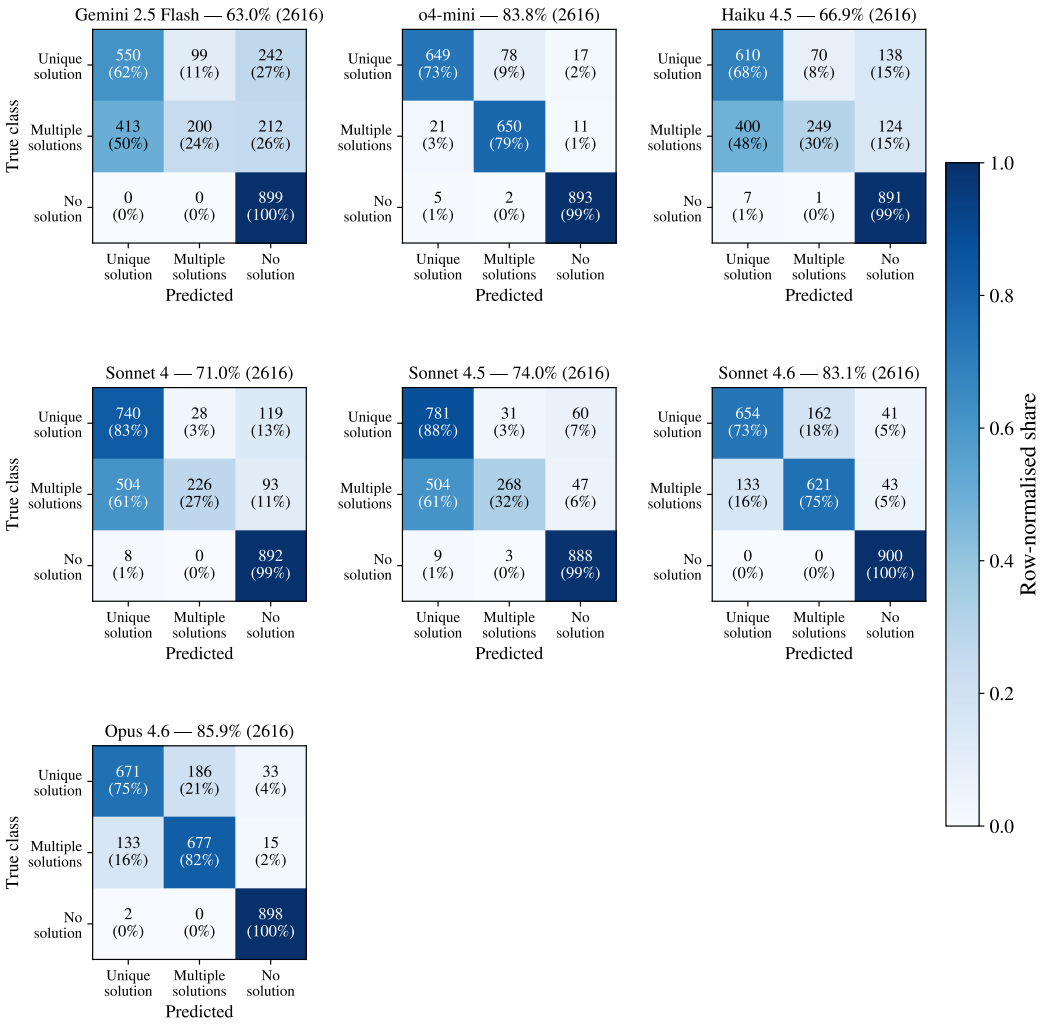


Figure 18: All models recover nearly every unsatisfiable instance; frontier errors are near-symmetric swaps between the unique and multiple-solution classes. Cells are row-normalized shares with counts (2,616 `Classification` instances per model); panels are sorted by overall classification accuracy.

Patterns in the confusion matrices. Two findings add to Table 20. First, every model recovers 96–100% of unsatisfiable instances (caveat below). Second, frontier errors are near-symmetric swaps between the unique and multiple-solution classes. Opus 4.6, o4-mini, and Sonnet 4.6 have comparable off-diagonal mass in both directions (133–175 multiple-solution rows predicted unique; 162–242 the other way), which reflects the difficulty of deciding whether a remaining constraint forces uniqueness.

Set-exact accuracy conditional on "unique". On the 891 unique-solution rows we ask a stricter question: did the model predict the unique-solution label *and* produce the correct set-block? Table 21 reports both unconditional set-exact and set-exact conditional on the model predicting "unique". o4-mini is almost always correct when it predicts "unique"; Gemini's "unique" predictions rarely yield the correct set, which matches its class-level over-prediction of "unique".

Parse failures: truncation, not format drift. Malformed-parse rates stay at or below 1.0% for six of eight models. Haiku 4.5 (4.8%, 126/2616) and o4-mini (9.8%, 256/2616) are the exceptions. All of their malformed instances terminate at the 8,192-token cap, with multi-thousand-token case enumeration running out of budget before `<solution>` can close. The failures therefore reflect the token cap, not template or parser defects. Haiku's 126 malformed instances skew to harder classes

Table 21: o4-mini is nearly always correct when it predicts "unique," but most models are not. We report set-exact accuracy (%) on the 891 unique-solution rows under the `Explicit` prompt; *Unconditional* counts all rows, *Conditional* restricts to rows where the model predicted "unique."

Model	Unconditional	Conditional on "unique"
o4-mini	72.5	99.5
Sonnet 4.6	64.9	88.4
Opus 4.6	62.0	82.3
Sonnet 4.5	61.5	70.2
Sonnet 4	58.4	70.3
Haiku 4.5	38.2	55.7
Gemini 2.5 Flash	5.7	9.3

(73 unique, 52 multiple, 1 unsat). A 16,384-token cap should recover most of them and is scoped for v2.

Unsatisfiable-class surface-detectability caveat. Our unsatisfiable instances contain two constraints that name the same target set with different right-hand sides (e.g. two `A△C` $= \{\ldots\}$ lines, or a `size` $= k$ paired with a `value` $= \{\ldots\}$ where $|\{\ldots\}| \neq k$). A model can flag such a problem by scanning for duplicate constraint heads, without reasoning about satisfiability. The 96–100% unsatisfiable-class accuracy in Table 20 is therefore a ceiling on surface detection, not on reasoning. A harder variant that requires deducing a contradiction from non-repeating constraints is scoped for v2. We quantify this ceiling below.

J.4 Shortcut audit for the unsatisfiable class

We test whether the near-saturated unsatisfiable-class accuracy in Table 20 reflects surface matching rather than solvability reasoning. We consider two candidate shortcuts: **S1**, predicting `UNSOLVABLE` whenever a target set name is absent from C'; and **S2**, predicting it whenever two value constraints over the same pair and type carry different right-hand sides. S1 is foreclosed by construction — every target set appears in at least one constraint in all 2,616 `Classification` instances — and S2 is the exact injection pattern that generates the unsatisfiable class, appearing incidentally on 16.1% of unique and 12.0% of multiple-solution rows.

We score both as no-reasoning classifiers (default `UNIQUE` otherwise) in Table 22. B1 misses every unsatisfiable instance; the specific shortcut named in the reviewer question cannot produce the observed accuracy. B2 recovers the unsatisfiable class perfectly, and six of seven models score within 1 pp of it on this class, so unsatisfiable-class accuracy is compatible with a surface-matching rule and does not on its own evidence solvability reasoning. B2's overall accuracy is lower than every real model because it misclassifies the 16% of unique rows that incidentally carry the cue; the useful comparison is per-class, not overall.

A second check corroborates this reading. The construction injects one of four value-type twins whose right-hand sides differ sharply in size: `intersection_value` and `difference_value` typically name four elements or fewer, while `union_value` and `symmetric_difference_value` name roughly $|U|/2$ to $|U|$, so twins often differ by a single swapped element. A model performing class-uniform solvability reasoning should be insensitive to this axis. Table 23 shows the opposite: accuracy tracks RHS size for every model, with the large-RHS types carrying essentially all residual error.

Table 22: A surface rule reaches the models' unsatisfiable-class ceiling. Per-class accuracy (%) of two no-reasoning baselines on the 2,616 `Classification` instances; both default to `UNIQUE`.

Baseline	Unique *(n=891)*	Multiple *(n=825)*	Unsatisfiable *(n=900)*	Overall *(n=2,616)*
B1: missing-name ⇒ unsat	100.0	0.0	0.0	34.1
B2: value-twin ⇒ unsat	84.0	0.0	**100.0**	63.0

Table 23: Residual unsat errors concentrate on the hardest-to-spot twins. Unsat accuracy (%) by `InjectedConstraintType`; columns are ordered by mean right-hand-side size (left smallest). Per-model row totals: intersection 94, difference 369, symmetric difference 253, union 184.

Model	intersection	difference	sym. diff.	union
Opus 4.6	100.0	100.0	100.0	98.9
Sonnet 4.6	100.0	100.0	100.0	100.0
Sonnet 4.5	100.0	99.5	98.8	96.2
Sonnet 4	97.9	99.7	98.8	98.9
Haiku 4.5	100.0	100.0	98.0	97.8
o4-mini	100.0	100.0	98.8	97.8
Gemini 2.5 Flash	100.0	100.0	100.0	99.5
Llama 3.3 70B	100.0	97.8	93.3	91.8

Refusal vs. inference in the `Explicit`-prompt drop. Under the `Explicit` prompt a model can fail in two ways. It can output `<solution>NO_UNIQUE_SOLUTION</solution>` on an instance that does admit a unique solution, or it can output the wrong answer. We separate the two by looking only at the rows where the model answered (Table 24). The two behaviors do not track capability in the same way. o4-mini, Sonnet 4.6, Gemini 2.5 Flash, and GPT-5.4 recover to $\geq$96% once we drop the refusals, so refusal alone explains their drop. Opus 4.6, Sonnet 4.5, Sonnet 4, and Haiku 4.5 still lose 9–17 pp on the rows they did answer, so the explicit wording hurts their reasoning as well. The `Implicit` prompt does not accept `NO_UNIQUE_SOLUTION`, so the `Implicit` "refusals" in Table 24 are parse failures (mostly Gemini Flash-Lite and Flash truncating at the 8,192-token cap). One caveat: when a model refuses often, the rows it does answer may not be a random sample, so we do not read cross-model differences in answered-row accuracy as pure capability gaps.

Table 24: `Explicit`-prompt refusal decomposition on the 900 unique-solution instances. *Refuse%* is the share with empty `LMSolution` (`NO_UNIQUE_SOLUTION` or parse failure). *Uncond. acc* counts refusals as incorrect; *Attempted acc* drops them. `Implicit`-prompt columns are given for reference. Rows are grouped by decomposition pattern. Sonnet 3.7 omitted (no `Explicit`-prompt run).

Model	`Explicit` (same 900)			`Implicit` prompt	
	Refuse%	Uncond. acc	Attempted acc	Refuse%	Attempted acc
Drop consistent with refusal alone (attempted accuracy recovers to $\geq$96%):					
o4-mini	32.3	67.2	**99.3**	1.8	99.2
Gemini 2.5 Flash-Lite	63.9	35.6	**98.6**	56.9	97.7
Gemini 2.5 Flash	54.7	45.3	**100.0**	44.4	99.0
Sonnet 4.6	31.8	66.0	**96.7**	0.7	97.3
GPT-5.4	48.1	50.1	**96.5**	0.0	67.4
Residual inference gap remains after dropping refusals:					
Opus 4.6	19.1	71.3	88.1	0.1	97.0
Sonnet 4.5	10.6	77.7	86.9	0.4	95.6
Sonnet 4	11.9	73.5	83.5	0.7	92.7
Haiku 4.5	23.1	55.7	72.4	2.1	83.9
Over-refusal (`Implicit` prompt also degraded):					
Qwen3 32B	30.2	42.9	61.5	14.9	70.0
Mistral Large	73.2	12.5	46.5	7.3	42.7
GPT-5.4 Nano	99.2	0.5	61.9	7.0	39.7
GPT-5.4 Mini	100.0	0.0	—	0.0	42.2

K ASSIGNMENT$_{\text{HARD}}$

Stacking the `Explicit` prompt on `Assignment`$_{\text{Hard}}$ does not uniformly lower accuracy (Table 3, `Explicit` block). Sonnet 4.6 and Opus 4.6 actually score higher on `Explicit`-

prompt `Assignment`$_{\text{Hard}}$ than on `Explicit`-prompt `Assignment`. Every `Assignment`$_{\text{Hard}}$ instance is unique by construction, so the main `Explicit`-prompt failure mode — over-flagging `MULTIPLE_SOLUTIONS` on tractable unique problems — does not trigger. Sonnet 4 and Haiku 4.5, whose `MULTIPLE_SOLUTIONS` over-flagging is more frequent in absolute terms, do fall.

K.1 ASSIGNMENT$^{\text{HARD}}_{|U|+}$

We report per-model accuracy on `Assignment`$^{\text{Hard}}_{|U|+}$, a 229-instance subset of the extended scaling pool at low retention ($R \leq 0.15$) and value fraction at most 0.5 (Table 25). The subset differs from `Assignment`$_{\text{Hard}}$ in two ways, both forced by the contents of the extended pool rather than by design: overlap is fixed at O=0.0, because the pool was not generated with O=0.5, and the value-fraction threshold is 0.5 rather than 0.3, because only 2 of 658 instances in the extended pool meet VF $\leq$ 0.3 — the Monte Carlo minimization retains value constraints preferentially at high $|U|$, so the size-heavy regime is not reachable by filtering alone. All four models were scored across the full $|U| \in \{30, 50, 70, 100, 150, 200, 250\}$ range (n=156 each).

Table 25: Frontier models degrade gracefully on `Assignment`$^{\text{Hard}}_{|U|+}$ while Haiku 4.5 collapses at large $|U|$. We pool all 156 instances per model; per-$|U|$ columns report Acc_{set} (cell n varies, 17–51).

	Overall		Per-$\lvert U\rvert$ (low range)				Per-$\lvert U\rvert$ (high range)		
Model	n	All	30	50	70	100	150	200	250
Opus 4.6	156	91.4	100.0	95.2	88.8	97.0	88.0	89.6	77.1
Sonnet 4.6	156	90.8	96.4	95.5	98.4	89.0	88.0	84.2	81.7
Sonnet 4	156	86.7	97.0	88.4	93.1	92.8	82.6	75.4	75.1
Haiku 4.5	156	68.0	89.4	75.5	74.0	73.2	45.7	68.6	39.5

L ERROR TAXONOMY

We define the six error categories referenced in the main text and give per-model, per-retention, and per-N breakdowns.

L.1 CATEGORY DEFINITIONS

For each incorrect set $\hat{S}_i \neq S_i^*$, let $E_i = \hat{S}_i \setminus S_i^*$ (extras) and $M_i = S_i^* \setminus \hat{S}_i$ (drops). Each set-level error falls into one of six categories:

- **Over-inclusion**: $E_i \neq \emptyset$ and $M_i = \emptyset$. The prediction is a strict superset of the target.
- **Under-inclusion**: $E_i = \emptyset$ and $M_i \neq \emptyset$. The prediction is a strict subset of the target.
- **Partial displacement**: $E_i \neq \emptyset$, $M_i \neq \emptyset$, and $\hat{S}_i \cap S_i^* \neq \emptyset$. Both extras and drops, but the prediction retains at least one correct element.
- **Complete substitution**: $E_i \neq \emptyset$, $M_i \neq \emptyset$, and $\hat{S}_i \cap S_i^* = \emptyset$. The prediction and target are disjoint.
- **Reasoning timeout**: the model returned a response but no `<solution>...</solution>` block could be parsed. Attributed at the problem level.
- **Malformed output**: `<solution>` tags are present but the body is not parseable (e.g. truncated JSON, stray tokens, wrong number of sets).

The first four categories partition parseable set-level errors. The last two apply at the problem level when no parseable prediction exists.

L.2 PER-MODEL BREAKDOWNS

Figures 19–24 break errors down by model, retention, N, element flow, severity, and the three calibration diagnostics.

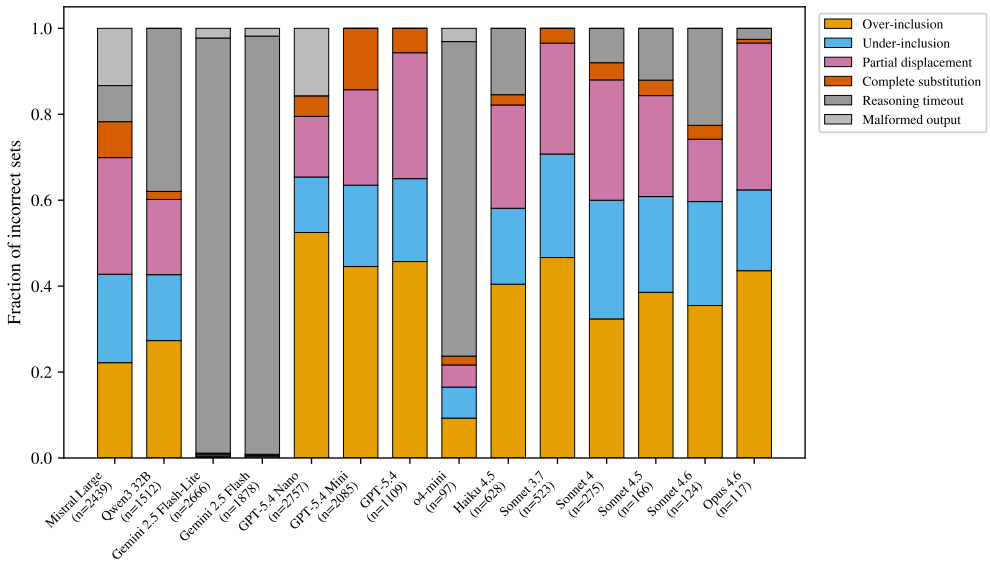


Figure 19: Over-inclusion and partial displacement dominate set-level errors; reasoning timeout concentrates in the two Gemini models. Bars split each model's errors into the six categories of Appendix L.1.

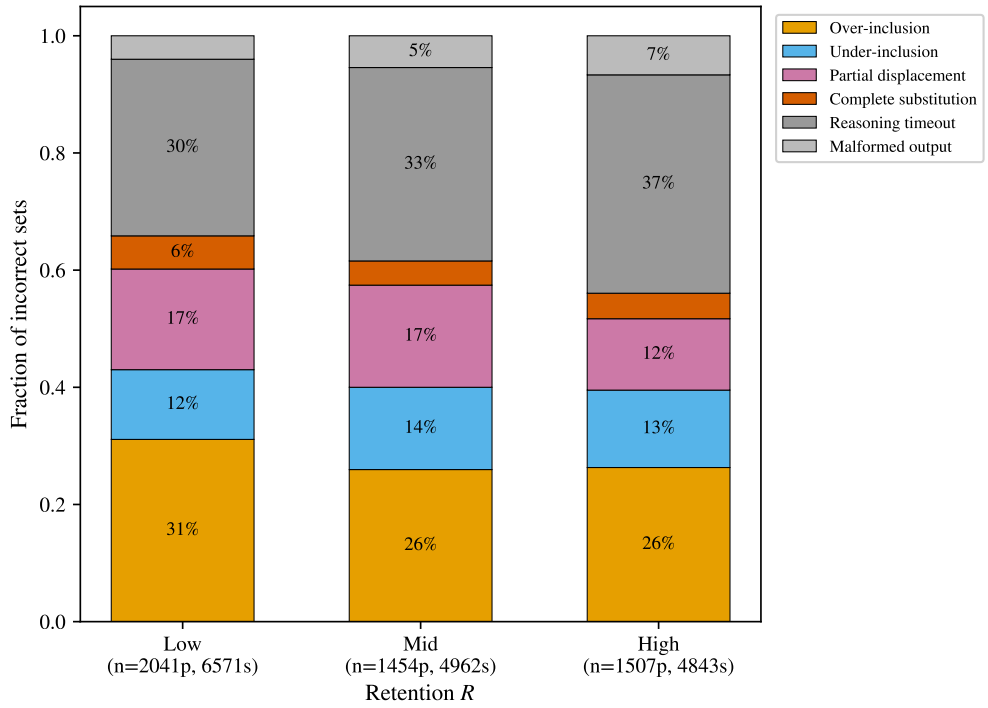


Figure 20: Over-inclusion dominates at high R; partial displacement and complete substitution rise at low R. Pooled across 16 models.

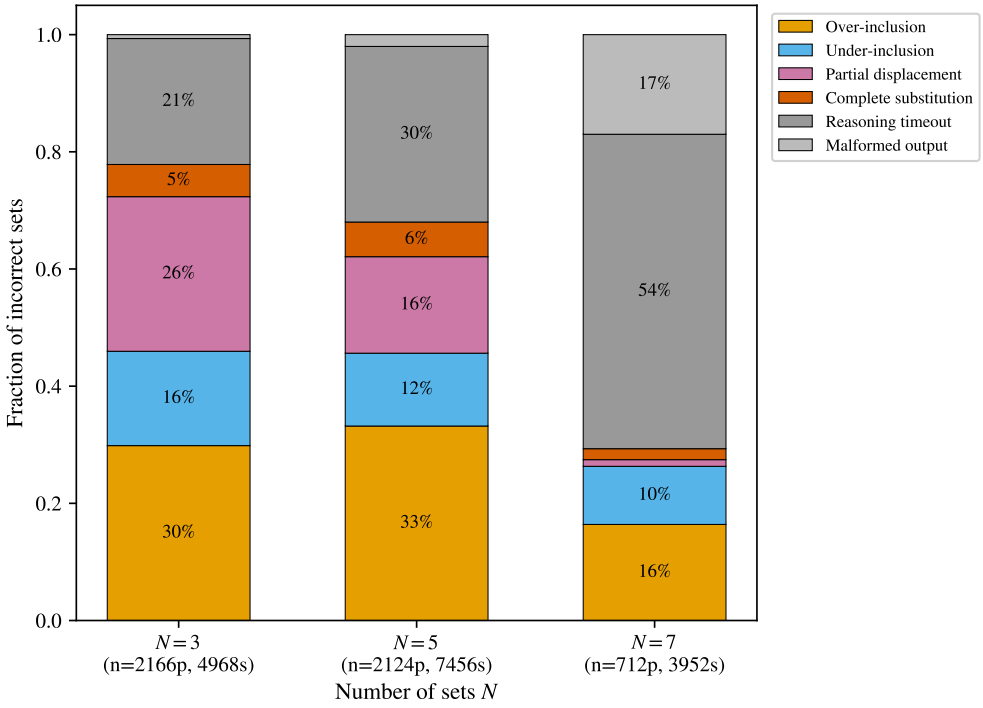


Figure 21: Over-inclusion grows with N while reasoning timeout diminishes. Pooled across 16 models.

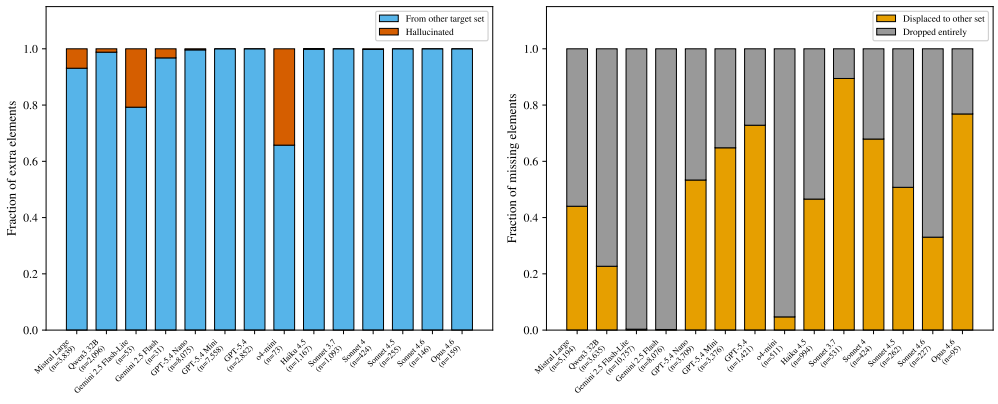


Figure 22: Models shuffle elements between sets far more often than they invent them, and drop missing elements more often than they displace. Left: 98.8% of extras belong to another target set; 1.24% (416/33,539) are hallucinated. Right: 28.5% of missing elements land in a different predicted set; 71.5% are dropped outright. Pooled across 16 models.

L.3 Violation rate by constraint type (Table 26)

M Machine baselines and solver runtime

We bracket both tasks with two machine reference points, so that model scores can be read against something other than each other. The upper reference is an exact SMT solver, which solves 100% of instances on both tasks; per-instance Z3 runtime has a median of 0.545 s, a 99th percentile of 1.315 s, and a maximum of 1.81 s. The lower reference is a no-search local-propagation heuristic that applies the inference rules of Table 7 to fixpoint and never guesses: it fully solves 33.6% of the

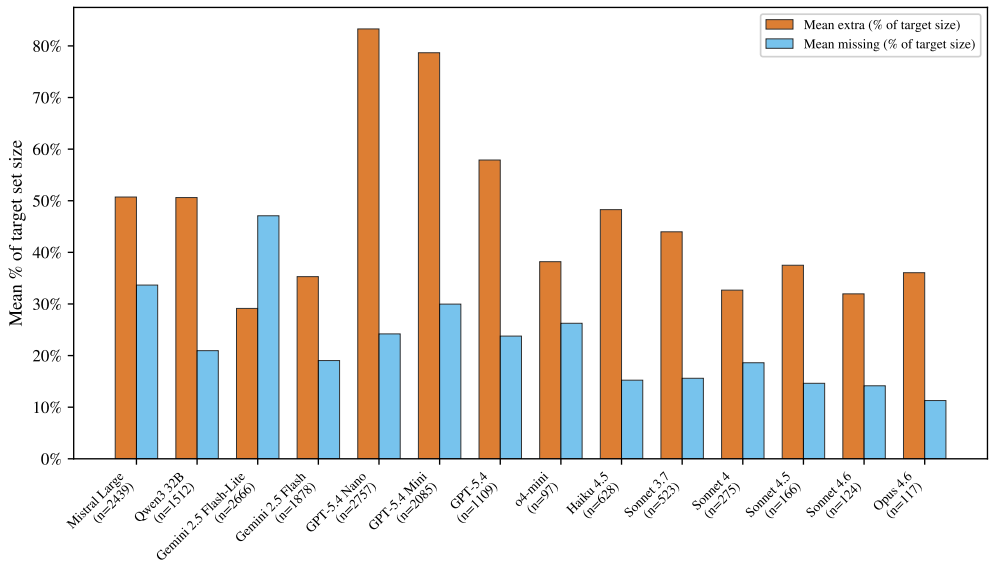


Figure 23: Error severity tracks capability: GPT-5.4 Nano is most severe (83% extra, 24% missing), while frontier models stay balanced (around 35% extra, 12–15% missing). Values are mean extras and drops per wrong set, as a fraction of target size.

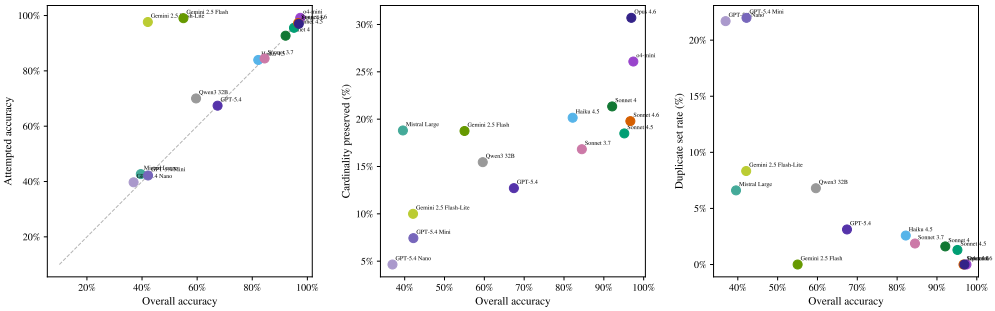


Figure 24: Three per-model diagnostics against overall accuracy. Left: attempted accuracy; points above the diagonal refuse wisely (Gemini 2.5 Flash Lite reaches 98% attempted accuracy by refusing 57% of problems). Center: cardinality preservation — frontier models preserve target size more reliably. Right: duplicate-set rate, concentrated in the weaker GPT-5.4 models.

Table 26: Mean rank of each pairwise constraint type by violation rate across the 14-model roster (↑= more frequently violated). Symmetric-difference variants (value: $\triangle$, size: $|\triangle|$) are violated most; subset/superset least.

$\cap$	$\setminus$	$\cup$	$\triangle$	$\lvert\cap\rvert$	$\lvert\setminus\rvert$	$\lvert\cup\rvert$	$\lvert\triangle\rvert$	$\subset$	$\supset$
3.5	6.1	5.5	**8.9**	4.5	6.8	7.1	**7.9**	2.4	2.3

`Assignment` sweep but only 2.6% of the exactly-minimal cores, which quantifies how much of the sweep is reachable without search and why the minimal cores are the harder slice. On classification, the two surface rules of Table 22 give floors of 34.1% and 63.0% against a 100% exact ceiling and a best model score of 85.9% on the three-way task. Both baselines and the runtime distribution are reproduced by a single released command.

N CHAIN-OF-THOUGHT INSPECTION

Final-answer scoring cannot distinguish a model that never derived the sets from one that derived them and then declined to commit, so we ran a trace pass over 240 visible chains of thought (120 correct, 120 incorrect, drawn across models with the audit-failing instances excluded). Two independent LLM annotators (Haiku 4.5 and Sonnet 4.6) labeled each trace under a six-category scheme — direct read-off, propagation, enumeration, contradiction search, abstention, and unverifiable narrative — with Cohen's $\kappa = 0.71$ (95% CI $[0.58, 0.81]$; raw agreement 90.8%). This is an automated double pass, not human ground truth, and we treat traces as corroborative text rather than as a faithful record of the computation performed (Lanham et al., 2023; Turpin et al., 2023).

Two findings corroborate the main text. First, abstention is often unwarranted on its own terms: in 8.4% of Opus 4.6's refusals under the abstention-enabled prompt, the trace had already derived every target set correctly before emitting `NO_UNIQUE_SOLUTION`. Second, enumerative markers are more frequent on size-constraint failures than on value-constraint failures, but by less than a naive comparison suggests. Holding correctness fixed (size-fail versus value-fail rather than the cross-diagonal size-fail versus value-correct), the pooled marker ratio is $1.84\times$ (95% CI $[1.43, 2.38]$), and $1.38\times$ for the frontier model, against $5.1\times$ for the confounded contrast. The difference persists but is modest, so we describe guess-and-check behavior behaviorally and do not claim it as the models' internal procedure.

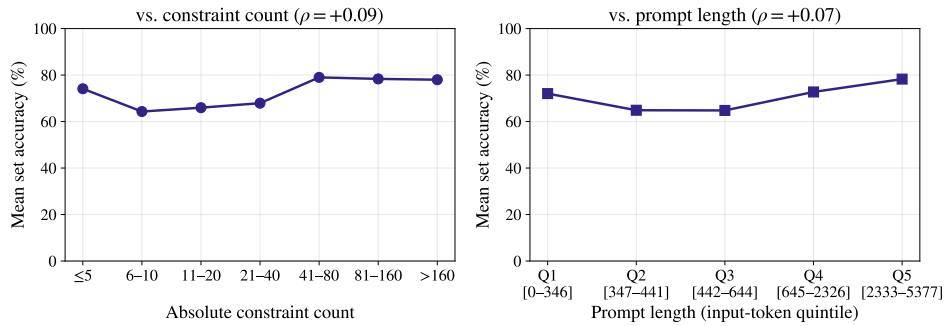


Figure 25: Longer prompts are not harder prompts. Accuracy rises slightly with both the number of constraints (Spearman $\rho = 0.11$) and prompt length ($\rho = 0.09$), the opposite of what a length-or-truncation account of the low-retention failures predicts. Length still covaries with redundancy, which the randomized intervention in Appendix H addresses directly.

O MODEL PROVENANCE

Model names in this paper are display names, not anonymization aliases. Table 27 gives the resolved provider version string for every evaluated system, so identifiability can be checked without running anything; Bedrock identifiers appear with the `us.` region prefix and `:0` version suffix as issued, and every results file on disk is named after its identifier or registry key, so the mapping is checkable against the released runs.

The release adds raw outputs, per-run provenance (provider, inference settings, code commit), the solver version, the generator commit, and one reproduction command per headline table. Two limits are worth stating plainly. The historical main-run records predate unified provenance logging, so the results files' 2026-04-21 git commit date is a *proxy* for an access date rather than an API timestamp, and exact configured effort or finish-reason fields are unavailable for some providers; we release every recoverable token, truncation, and latency field rather than infer the missing ones. The matched-effort study of Appendix F.6 carries an exact 2026-07-26 access date.

P LATENCY

Response latency varies by more than an order of magnitude across the 14-model roster (Figure 26; selected medians in Table 28). Latency does not predict accuracy. GPT-5.4 Mini is $26\times$ faster than

Table 27: Display names and resolved provider model IDs. Llama 3.3 70B appears only in the auxiliary rows of Table 23 and the value/size range of §5, not in the main roster of fourteen systems.

Display name	Provider model ID
o4-mini	`o4-mini`
GPT-5.4	`gpt-5.4`
GPT-5.4 Mini	`gpt-5.4-mini`
GPT-5.4 Nano	`gpt-5.4-nano`
Opus 4.6	`us.anthropic.claude-opus-4-6-v1`
Sonnet 4.6	`us.anthropic.claude-sonnet-4-6`
Sonnet 4.5	`us.anthropic.claude-sonnet-4-5-20250929-v1:0`
Sonnet 4	`us.anthropic.claude-sonnet-4-20250514-v1:0`
Sonnet 3.7	`us.anthropic.claude-3-7-sonnet-20250219-v1:0`
Haiku 4.5	`us.anthropic.claude-haiku-4-5-20251001-v1:0`
Gemini 2.5 Flash	`gemini-2.5-flash`
Gemini 2.5 Flash-Lite	`gemini-2.5-flash-lite`
Mistral Large	`mistral.mistral-large-2402-v1:0`
Qwen3 32B	`qwen.qwen3-32b-v1:0`
Llama 3.3 70B (auxiliary)	`us.meta.llama3-3-70b-instruct-v1:0`

o4-mini but scores 55 pp lower. A smaller open-source model is faster still but only slightly more accurate. Figure 27 plots latency against accuracy.

Table 28: Median response latency spans more than an order of magnitude across selected models (seconds).

Model	Median (s)
GPT-5.4 Mini	1.2
Sonnet 4	14.2
Opus 4.6	15.0
Sonnet 4.5	15.7
Gemini 2.5 Flash Lite	18.7
o4-mini	31.9

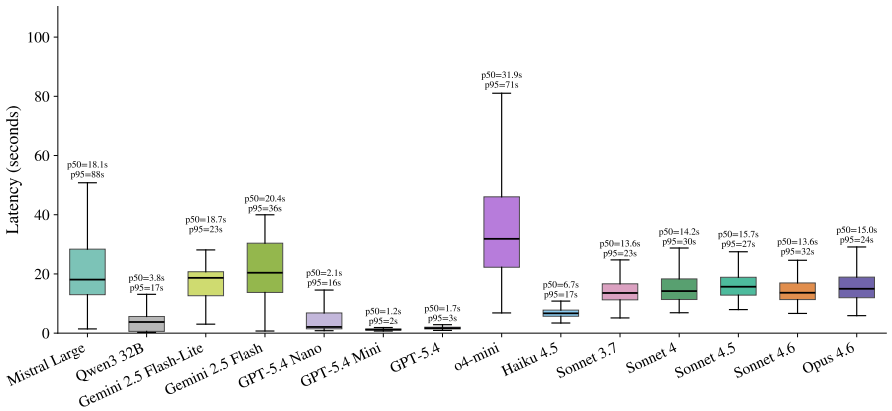


Figure 26: Response latency varies by more than an order of magnitude across the model roster. We annotate p50 and p95 for each model.

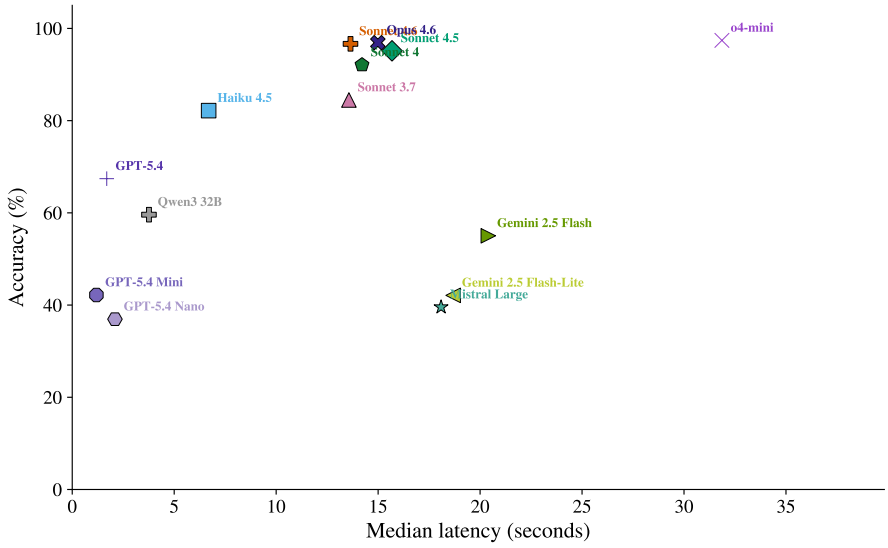


Figure 27: Per-model latency against accuracy. Slower models do not consistently score higher; o4-mini (31.9 s) and Gemini 2.5 Flash Lite (18.7 s) sit at opposite ends of the accuracy axis.