The Format Tax: Answer-Only JSON Contracts Suppress Multi-Step Reasoning in Claude Sonnet 4.6, and One Schema Field Buys It Back

Yash Singh
IIIT Una

July 20, 2026

Abstract

Production LLM pipelines routinely instruct models to answer in strict JSON so that downstream code can parse the output. Prior work disagreed on whether such format restrictions degrade reasoning, but that debate predates current frontier models. We measure the effect directly on two current models, Claude Haiku 4.5 and Claude Sonnet 4.6, using 150 trials over 25 freshly generated multi-step arithmetic word problems under three prompt-level output contracts: free text with an answer anchor, an answer-only JSON object, and a JSON object with a schema-embedded reasoning field. The tax is real, large, and model-dependent. Sonnet 4.6 solves 25/25 problems in free text but only 10/25 (40%) under the answer-only contract (exact McNemar $p = 0.0001$); adding a single `reasoning` field to the schema recovers 24/25 (96%, $p = 0.0005$ vs. answer-only) while using 59% fewer visible output tokens than free text. Haiku 4.5 scores 25/25 under all three contracts, but its billing metadata shows a median of 251 output tokens for a 9-token visible payload, consistent with hidden reasoning tokens that let it pay the tax in compute rather than accuracy. Within Sonnet, the only two hardest-tier successes under the answer-only contract are exactly the two trials whose billed tokens exceed the visible payload by an order of magnitude. We conclude that answer-only schemas are an unsafe default for multi-step tasks and that a leading reasoning field is a near-free mitigation.

1 Introduction

Structured output is the connective tissue of modern LLM systems. Tool calling, agent frameworks, and extraction pipelines all require the model to emit machine-parseable objects rather than prose [Schick et al., 2023, Patil et al., 2023], and every major provider now ships first-class structured-output support [OpenAI, 2024, Anthropic, 2026]. The cheapest and still most common way to obtain structure, however, is not decoder-level constraining but a prompt instruction: "respond with ONLY a JSON object of the form …". This paper asks what that instruction costs.

The question has a contested history. Tam et al. [2024] reported that format restrictions significantly degrade reasoning performance, with stricter constraints causing larger drops. The rebuttal from the Outlines team argued that the effect disappears under well-designed constrained decoding and that the original comparison conflated schema design with the constraint itself [Willard and Louf, 2024]. Subsequent benchmarking effort has focused largely on whether engines can produce schema-compliant output efficiently [Geng et al., 2025], not on what compliance does to answer quality. Meanwhile the models themselves have changed: current frontier models are trained for stronger instruction following and, in some deployments, may spend internal reasoning tokens that

never appear in the visible output. Both changes plausibly alter the size, and even the sign, of the format tax.

We contribute a small, fully reproducible measurement on two current models, Claude Haiku 4.5 and Claude Sonnet 4.6, isolating one variable: the prompt-level output contract. Using 25 freshly authored multi-step arithmetic word problems in three difficulty tiers, we run each model under three contracts (free text, answer-only JSON, JSON with a reasoning field), for 150 total trials, and record correctness, strict and lenient parse compliance, API latency, and billed output tokens for every trial.

Our findings:

1. **The tax is alive on current models.** Sonnet 4.6 drops from 25/25 correct in free text to 10/25 (40%) under an answer-only JSON contract; the paired exact McNemar test gives $p = 0.0001$ with 15 discordant problems, all favoring free text.

2. **It is a reasoning-suppression tax, not a JSON tax.** Adding one schema field, `reasoning`, before `answer` recovers Sonnet to 24/25 (96%), statistically indistinguishable from free text ($p = 1.0$), while emitting a median of only 98 visible output tokens versus 240 for free text.

3. **Models pay the tax in different currencies.** Haiku 4.5 is 100% correct under every contract, but under the answer-only contract it bills 144–421 output tokens (median 251) while visibly emitting only a 9-token JSON object, and its median latency rises above its free-text latency. The accuracy tax it avoids is paid as hidden compute.

4. **Compliance and correctness anticorrelate.** Sonnet obeys the contract perfectly (75/75 strictly parseable JSON objects) and pays the accuracy tax; Haiku wraps its JSON in Markdown fences in 20/50 JSON-contract trials, violating the letter of the instruction, and pays no accuracy tax. Compliance metrics alone are therefore a misleading measure of structured-output quality.

2 Related Work

Format restrictions and the debate. Tam et al. [2024] systematically compared free-form generation against JSON, XML, and YAML output across reasoning and classification tasks, finding that stricter format constraints degrade reasoning-heavy task performance. Willard and Louf [2024] disputed the generality of the conclusion, showing that with schema-aware prompting and proper constrained decoding the gap can close or reverse. Our design is deliberately neutral in this debate: we do not use decoder-level constraints at all, only prompt-level contracts, which remain the dominant practice in application code, and we include the schema-repair condition (a reasoning field) that the rebuttal identifies as decisive. Our results support both sides: the naive answer-only contract is catastrophic for one model, and the repaired schema erases nearly all of the loss.

Constrained decoding. A complementary line of work generates guaranteed-valid structure by masking invalid tokens during sampling: finite-state guided generation [Willard and Louf, 2023], non-invasive constrained decoding that minimizes distribution distortion [Beurer-Kellner et al., 2024], grammar-constrained decoding for structured NLP tasks [Geng et al., 2023], and grammar-railed decoding for enterprise SQL [Arjmandi, 2026]. Geng et al. [2025] benchmark such engines on efficiency and schema coverage over roughly 10,000 real-world schemas. Provider APIs now expose this machinery directly [OpenAI, 2024, Anthropic, 2026]. These systems make *invalid*

Contract	Instruction (followed by the problem text)
FREE	`Solve this problem step by step. On the last line, output exactly 'Answer: <integer>'.`
JSON-ANSWER	`Respond with ONLY a JSON object exactly of the form {"answer": <integer>}. No other text, no explanation, no markdown fences.`
JSON-REASON	`Respond with ONLY a JSON object exactly of the form {"reasoning": "<concise step-by-step reasoning>", "answer": <integer>}. No other text, no markdown fences.`

Table 1: The three prompt-level output contracts. The problem statement is appended verbatim to each instruction.

output impossible; our results concern what the *contract itself* does to the computation the model performs, which applies whether the contract is enforced by prompt or by grammar.

Reasoning as visible token computation. Chain-of-thought prompting shows that autoregressive models solve multi-step problems dramatically better when allowed to emit intermediate steps [Wei et al., 2022], even zero-shot [Kojima et al., 2022], and multi-step arithmetic word problems in the GSM8K style are the canonical instrument for measuring this [Cobbe et al., 2021]. The format-tax question is the contrapositive of chain-of-thought: if emitting steps helps, does a contract that forbids emitting steps hurt, and can the steps be smuggled inside the schema instead? Our `json_cot` condition embeds the chain of thought in a JSON field, testing exactly this.

3 Method

3.1 Problems

We generate 25 multi-step arithmetic word problems with exact integer answers using a seeded generator (`gen_problems.py`, seed 201), in three difficulty tiers: 8 three-step problems (tier 1), 9 four-to-five-step problems involving exact percentages (tier 2), and 8 six-to-eight-step chained inventory problems involving percentages and integer division with remainders (tier 3). Problems are freshly authored from templates rather than drawn from GSM8K [Cobbe et al., 2021], so no model has seen these exact instances in training. The generator, the emitted `problems.json`, and all raw trial records are included in the artifact.

3.2 Output contracts

Each problem is presented under three prompt-level contracts (Table 1). The problem text is identical across contracts; only the output instruction changes. FREE permits unbounded visible reasoning and anchors the final answer for parsing. JSON-ANSWER (script name `json_strict`) is the answer-only contract common in production extraction code. JSON-REASON (script name `json_cot`) is identical except that the schema places a `reasoning` string field before the `answer` field, so the model's autoregressive decoding passes through its own reasoning before committing to a number.

3.3 Formal setup

Definition 1 (Output contract)**.** *An output contract s is a mapping from a problem q to a prompt $s(q)$ that fixes the surface form of the required response while leaving the underlying question unchanged. We write $A_s(m)$ for the accuracy of model m over the problem set Q under contract s:*

$$A_s(m) \;=\; \frac{1}{|Q|}\sum_{q\in Q}\mathbf{1}[\,\mathrm{ans}_m(s(q)) = y_q\,], \tag{1}$$

where y_q is the ground-truth integer and ans_m is the parsed answer (lenient parsing; Section 3.4).

Definition 2 (Format tax)**.** *The format tax of contract s for model m is the accuracy it forfeits relative to the free contract:*

$$T_s(m) \;=\; A_{\textsc{free}}(m) - A_s(m). \tag{2}$$

Proposition 1 (Visible-computation view)**.** *Under an answer-only contract, the number of autoregressive decoding steps available for task computation after the prompt is bounded by the token length of the answer object itself (here 9–10 tokens), independent of the number of reasoning steps the problem requires; under a contract whose schema includes a leading free-text reasoning field, that budget grows with the emitted reasoning. Hence, for a model whose multi-step performance depends on emitted intermediate tokens [Wei et al., 2022, Kojima et al., 2022], $T_s(m)$ should increase with problem step count under* JSON-ANSWER *and vanish under* JSON-REASON*, unless the model can substitute hidden (non-visible) reasoning tokens.*

Proposition 1 is a framing rather than a theorem about transformers; the experiment tests both of its testable consequences: the difficulty interaction (Section 4.2) and the hidden-token escape hatch (Section 4.3).

For uncertainty we report 95% Wilson score intervals [Wilson, 1927] on each cell proportion,

$$\mathrm{CI}_{95}(k,n) \;=\; \frac{\hat p + \frac{z^2}{2n} \pm z\sqrt{\frac{\hat p(1-\hat p)}{n} + \frac{z^2}{4n^2}}}{1+\frac{z^2}{n}}, \qquad \hat p = k/n,\; z = 1.96, \tag{3}$$

and, because every problem is answered under every contract, we compare contracts within a model with the exact McNemar test [McNemar, 1947]: for discordant counts b (correct under s_1 only) and c (correct under s_2 only), the two-sided p-value is the exact binomial tail

$$p \;=\; \min\Big(1,\; \sum_{k:\,\binom{n}{k}\le\binom{n}{b}} \binom{n}{k} 2^{-n}\Big), \qquad n = b + c. \tag{4}$$

3.4 Harness and measurement

Each trial is one headless call to the Claude Code command-line interface (`claude -p`, version 2.1.183) with the model aliases `haiku` and `sonnet`, which on the run date (July 20, 2026) resolve to Claude Haiku 4.5 and Claude Sonnet 4.6. The CLI's JSON envelope (`--output-format json`) supplies, per trial, the result text, the API duration in milliseconds, the billed output-token count, and the cost in USD. Trials run 8-way parallel with a 180-second timeout; all 150 calls succeeded on the first attempt (zero errors, zero timeouts). Total measured cost of the experiment was $4.62.

Parsing is two-stage. *Strict* parsing requires the raw response to be exactly the contracted form: for FREE, a final `Answer: <integer>` line; for the JSON contracts, that the entire response

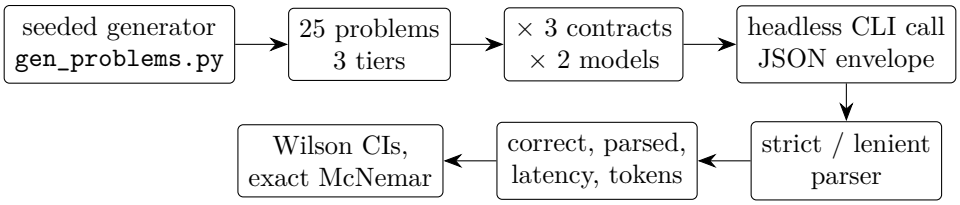

Figure 1: Experiment pipeline. Each of the 150 trials is one independent headless model call; per-trial records are appended to `trials.jsonl` and analyzed by `analyze.py`.

Model	Contract	n	Correct	Acc.	95% CI	Med. latency (ms)	Med. output tokens
Haiku 4.5	FREE	25	25	1.00	[0.867, 1.000]	6931	464
Haiku 4.5	JSON-ANSWER	25	25	1.00	[0.867, 1.000]	8144	251
Haiku 4.5	JSON-REASON	25	25	1.00	[0.867, 1.000]	6959	385
Sonnet 4.6	FREE	25	25	1.00	[0.867, 1.000]	6867	240
Sonnet 4.6	JSON-ANSWER	25	10	0.40	[0.234, 0.593]	5033	9
Sonnet 4.6	JSON-REASON	25	24	0.96	[0.805, 0.993]	5762	98

Table 2: Main results over 150 trials. Accuracy uses lenient parsing; intervals are 95% Wilson score intervals (Eq. 3). Latency is the API duration reported by the harness; output tokens are billed tokens, which may exceed the visible payload (Section 4.3).

body is a single parseable JSON object. *Lenient* parsing additionally strips Markdown code fences and extracts the first brace-delimited object. Correctness is always computed on the lenient parse, so accuracy differences below are never parsing artifacts: all 150 trials were leniently parseable (Table 4).

4 Results

4.1 Headline: the tax exists, is large, and one field removes it

Table 2 and Figure 2 give the headline numbers. Both models solve all 25 problems in the FREE condition, so the problem set is fully within their competence. Under JSON-ANSWER, Sonnet 4.6 falls to 10/25: a format tax of $T_{\text{JA}}(\text{Sonnet}) = 0.60$ by Eq. 2. The paired comparison is one-sided in the strongest sense: 15 problems flip from correct to incorrect and none flip the other way (exact McNemar $p = 0.0001$, Table 5). Under JSON-REASON the tax essentially vanishes: $T_{\text{JR}}(\text{Sonnet}) = 0.04$ (one problem), $p = 1.0$ versus FREE, and $p = 0.0005$ versus JSON-ANSWER (15 vs. 1 discordant problems). Haiku 4.5 shows zero tax in accuracy under either JSON contract; Section 4.3 shows it pays elsewhere.

4.2 The tax concentrates exactly where reasoning is needed

Table 3 and Figure 3 break Sonnet's collapse down by difficulty. Under JSON-ANSWER, Sonnet remains perfect on tier-1 problems (8/8), scores 0/9 on tier 2, and 2/8 on tier 3. This is the signature predicted by Proposition 1: a contract that caps visible computation at roughly ten tokens leaves enough slack for three-step arithmetic but not for four or more chained operations.

The failure contents are also informative, and the failure mode differs by tier. Tier-3 wrong

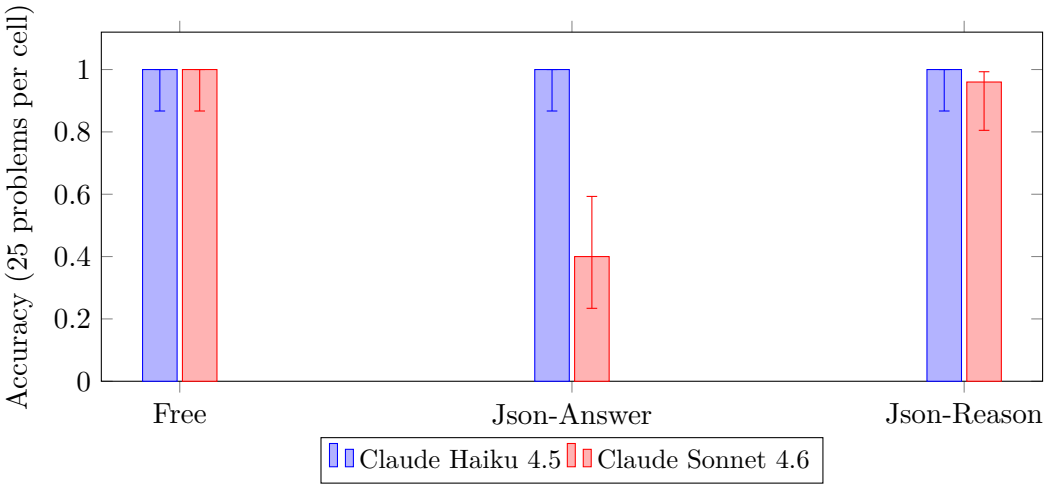

Figure 2: Accuracy by output contract (measured values from Table 2; error bars are 95% Wilson intervals). Sonnet 4.6 collapses to 0.40 under the answer-only contract and recovers to 0.96 when the schema includes a reasoning field. Haiku 4.5 is at ceiling throughout.

Model	Contract	Tier 1 ($n = 8$)	Tier 2 ($n = 9$)	Tier 3 ($n = 8$)
Haiku 4.5	FREE	8	9	8
Haiku 4.5	JSON-ANSWER	8	9	8
Haiku 4.5	JSON-REASON	8	9	8
Sonnet 4.6	FREE	8	9	8
Sonnet 4.6	JSON-ANSWER	8	0	2
Sonnet 4.6	JSON-REASON	7	9	8

Table 3: Correct answers per difficulty tier. Sonnet's failures under JSON-ANSWER are confined to the multi-step tiers: 3-step problems survive (8/8), 4–5-step problems collapse entirely (0/9), 6–8-step problems nearly so (2/8).

answers are near-miss magnitudes (e.g., 744 instead of 928; 888 instead of 944), consistent with dropped intermediate steps. Tier-2 wrong answers are instead far too small (e.g., 9 instead of 169; 14 instead of 314), consistent with the model emitting an intermediate quantity, such as a single step's result, in place of the final total. Neither pattern resembles random guessing. Notably, two *distinct* tier-3 problems (ground truths 928 and 840) received the identical wrong answer 744, suggesting shallow pattern completion in place of computation when no reasoning tokens are available.

4.3 Where the tax is paid: visible versus billed tokens

The token accounting (Figure 4) explains the cross-model difference. Under JSON-ANSWER, Sonnet's billed output is a median of 9 tokens, exactly the visible `{"answer": N}` payload: the model complies literally, performs no visible reasoning, and its billed tokens confirm essentially no hidden reasoning either. Haiku's visible payload under the same contract is the same 9-token object, yet its billed output tokens range from 144 to 421 (median 251), an order of magnitude above what it prints. The billing metadata thus indicates a substantial non-visible generation phase, consistent with internal reasoning tokens. Haiku also gets *slower* under JSON-ANSWER (median 8144 ms vs. 6931 ms free), while Sonnet gets faster (5033 ms vs. 6867 ms): Haiku spends time thinking privately,

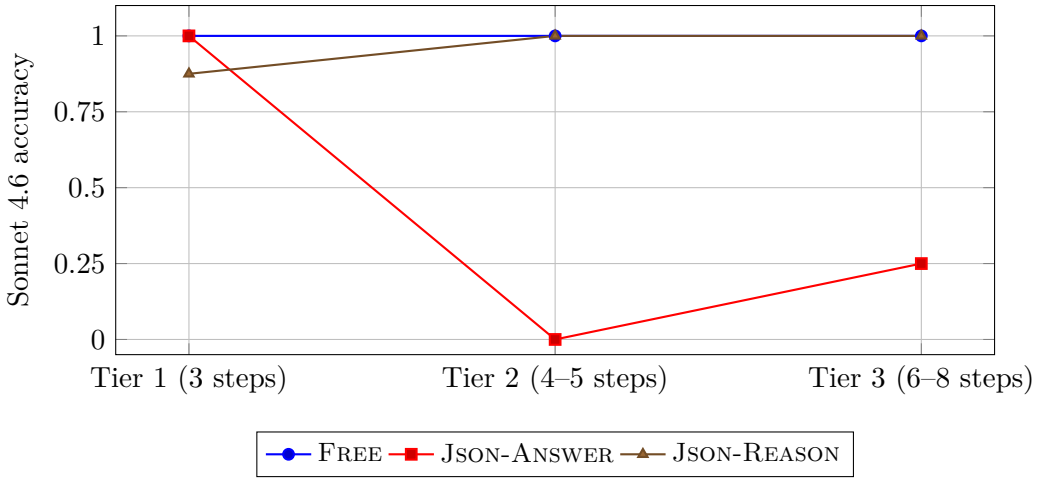

Figure 3: Sonnet 4.6 accuracy by difficulty tier under each contract (measured fractions from Table 3). The answer-only contract is harmless on 3-step problems and catastrophic beyond them, the interaction predicted by Proposition 1.

Model	Contract	Strict parse	Lenient parse	Fence-wrapped
Haiku 4.5	Free	25/25	25/25	0
Haiku 4.5	Json-Answer	22/25	25/25	3
Haiku 4.5	Json-Reason	8/25	25/25	17
Sonnet 4.6	Free	25/25	25/25	0
Sonnet 4.6	Json-Answer	25/25	25/25	0
Sonnet 4.6	Json-Reason	25/25	25/25	0

Table 4: Format compliance. Strict parse requires the response to be exactly the contracted form; lenient parse strips Markdown fences and extracts the first JSON object. Every Haiku strict-parse failure was a ```` ```json ```` fence around an otherwise valid object, despite the explicit "no markdown fences" instruction.

Sonnet simply stops thinking.

The strongest single piece of evidence is within-Sonnet. Its two tier-3 successes under Json-Answer are precisely the two trials whose billed output tokens (187 and 199) vastly exceed the 9–10 tokens billed on every one of the 15 failures. In other words, on the rare occasions Sonnet did spend hidden tokens, it got the hard problem right; whenever it billed only the answer object, it failed everything beyond tier 1.

Finally, the recovery condition is cheap. Sonnet under Json-Reason emits a median of 98 output tokens, 59% fewer than its own free-text condition (240), with lower median latency (5762 ms vs. 6867 ms) and 96% accuracy. For this workload the repaired schema dominates free text on cost and latency at equal accuracy.

4.4 Compliance and correctness are different axes

Table 4 reports parse compliance. Sonnet is perfectly compliant in all 75 of its trials. Haiku violates the letter of the contract frequently, wrapping its JSON in Markdown fences in 3/25 Json-Answer trials and 17/25 Json-Reason trials; one line of fence-stripping recovers 100% of them. The juxtaposition with Table 2 is the practical point: the model that follows the format instruction

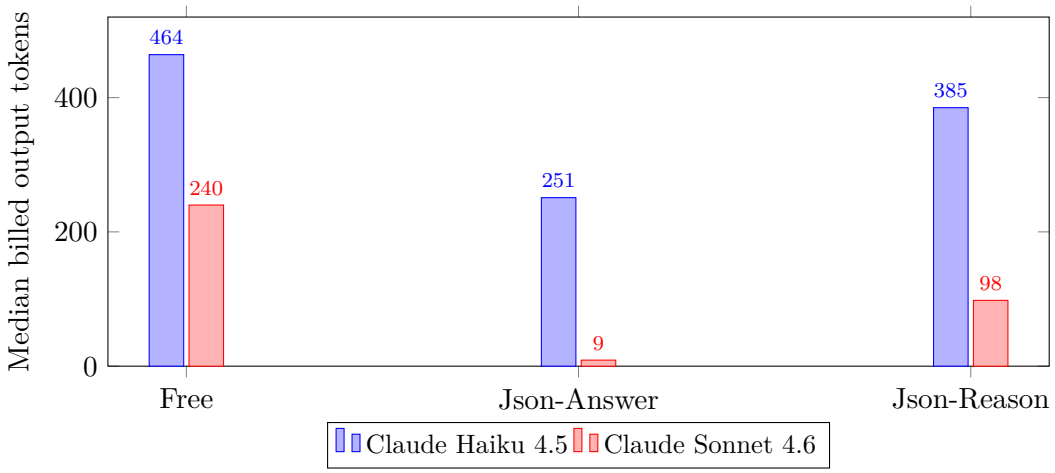

Figure 4: Median billed output tokens per contract (measured values). Under JSON-ANSWER, Sonnet 4.6 bills a median of 9 tokens, the answer object itself, and loses 60 accuracy points. Haiku 4.5 bills a median of 251 tokens for the same 9-token visible payload and loses nothing: the tax is paid in hidden compute instead of accuracy.

Model	Comparison s_1 vs. s_2	s_1 only	s_2 only	Exact p
Haiku 4.5	FREE vs. JSON-ANSWER	0	0	1.0000
Haiku 4.5	JSON-REASON vs. JSON-ANSWER	0	0	1.0000
Haiku 4.5	FREE vs. JSON-REASON	0	0	1.0000
Sonnet 4.6	FREE vs. JSON-ANSWER	15	0	0.0001
Sonnet 4.6	JSON-REASON vs. JSON-ANSWER	15	1	0.0005
Sonnet 4.6	FREE vs. JSON-REASON	1	0	1.0000

Table 5: Paired exact McNemar tests (Eq. 4) over the 25 shared problems. Columns give discordant counts: problems solved under s_1 but not s_2, and vice versa.

perfectly is the one that loses 60 accuracy points to it. Benchmarks that score structured-output quality by schema compliance alone [Geng et al., 2025] would rank Sonnet's JSON-ANSWER behavior as ideal.

5 Discussion

The tax is reasoning suppression, and the schema is the lever. Every observation lines up with the visible-computation account of Proposition 1. The tax appears only past three reasoning steps (Figure 3); it appears only in the model that actually stops generating reasoning tokens (Figure 4); it disappears when the schema itself carries the reasoning; and within the taxed model, the only hard-problem successes coincide with anomalously high billed token counts. This unifies the apparently conflicting prior findings: format restrictions do hurt [Tam et al., 2024] when the contracted object leaves no room for intermediate computation, and the effect is repairable at the schema level [Willard and Louf, 2024] because the mechanism is token budget, not JSON syntax.

Practical guidance. First, answer-only schemas are an unsafe default for any task that might require multi-step reasoning; the failure mode is silent, schema-valid, and confidently wrong, the

worst combination for production pipelines. Second, the mitigation costs one schema field: place a `reasoning` string *before* the answer field so decoding order forces the computation to happen first. In our data this recovered 56 of the 60 lost accuracy points and was simultaneously cheaper and faster than free text. Third, do not infer safety from compliance dashboards: perfect schema compliance coexisted here with a 60-point accuracy collapse. Fourth, models with hidden reasoning phases blunt the tax but do not make it free; Haiku paid roughly 27× the visible token count in billed tokens and higher latency under the answer-only contract. Teams using constrained decoding or provider structured-output modes [OpenAI, 2024, Anthropic, 2026, Willard and Louf, 2023, Beurer-Kellner et al., 2024, Geng et al., 2023, Arjmandi, 2026] should apply the same schema discipline: the grammar guarantees validity, while the field order and the presence of a reasoning slot govern accuracy.

Why the models diverge. We can measure the divergence but only conjecture its cause. The billing evidence indicates Haiku 4.5 ran a non-visible generation phase on every answer-only trial in this harness, while Sonnet 4.6 did so in only 2 of 25. Whether this reflects different default thinking configurations in the harness, different trained dispositions about when to think privately, or different instruction-following aggressiveness cannot be distinguished from the outside. The actionable reading is that the format tax is a per-deployment property, a function of model, harness defaults, and schema together, and should be measured, not assumed, for any given stack.

6 Limitations

Scale. Each cell is 25 problems and each trial a single sample, so cell accuracies carry wide Wilson intervals (Table 2); the headline Sonnet contrast is nevertheless a 15-vs-0 paired flip with $p = 0.0001$. **Task narrowness.** Problems come from three synthetic template families of integer arithmetic; the size of the tax on other reasoning genres (code, logic, multi-hop QA) is not established here, though the mechanism is generic. Both models are at ceiling in tier 1, so the easy-problem comparison is uninformative. **Harness opacity.** Trials ran through the Claude Code CLI, which contributes its own system prompt and default generation settings, including any internal-reasoning configuration; we did not control temperature or thinking budgets, and the "hidden reasoning" interpretation rests on billed-versus-visible token arithmetic rather than on inspection of reasoning traces. Absolute latencies also include harness overhead and parallel-execution contention. **Contract mechanism.** We tested prompt-level contracts only; grammar- or API-level constrained decoding may shift the numbers, though prior work suggests the schema-shape effect persists there [Willard and Louf, 2024, Geng et al., 2025]. **Coverage.** Two models from a single family were tested on a single day; the cross-model divergence we found is itself evidence that these numbers should not be extrapolated to other models without measurement.

7 Conclusion

On current frontier models, prompt-forced answer-only JSON can silently destroy multi-step reasoning: Claude Sonnet 4.6 went from 25/25 to 10/25 on problems it demonstrably knows how to solve, with every failure schema-valid. The fix costs one schema field and, in our measurements, less than half the visible tokens of free-form output. The deeper lesson is that an output contract is not a serialization detail; it is a constraint on how much computation the model may perform where you can see it, and different models compensate in different, billing-visible ways. Measure the tax on your own stack before you ship a schema.

Reproducibility. The artifact contains the seeded problem generator (`gen_problems.py`), the emitted problem set (`problems.json`), the trial runner with crash-safe incremental output (`run_trials.py`), all 150 raw trial records (`trials.jsonl`), and the analysis script and summary (`analyze.py`, `results.json`). Total API cost of the full run was $4.62.

References

Anthropic. Structured outputs - Claude developer documentation. https://docs.anthropic.com/en/docs/build-with-claude/structured-outputs, 2026.

Mohsen Arjmandi. GRID: Grammar-railed decoding for enterprise SQL generation. *arXiv preprint arXiv:2607.11951*, 2026. URL https://arxiv.org/abs/2607.11951.

Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. Guiding LLMs the right way: Fast, non-invasive constrained generation. *arXiv preprint arXiv:2403.06988*, 2024. URL https://arxiv.org/abs/2403.06988.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. URL https://arxiv.org/abs/2110.14168.

Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. Grammar-constrained decoding for structured NLP tasks without finetuning. *arXiv preprint arXiv:2305.13971*, 2023. URL https://arxiv.org/abs/2305.13971.

Saibo Geng, Hudson Cooper, Michał Moskal, Samuel Jenkins, Julian Berman, Nathan Ranchin, Robert West, Eric Horvitz, and Harsha Nori. JSONSchemaBench: A rigorous benchmark of structured outputs for language models. *arXiv preprint arXiv:2501.10868*, 2025. URL https://arxiv.org/abs/2501.10868.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022. URL https://arxiv.org/abs/2205.11916.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947. doi: 10.1007/BF02295996.

OpenAI. Introducing structured outputs in the API. https://openai.com/index/introducing-structured-outputs-in-the-api/, 2024.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive APIs. *arXiv preprint arXiv:2305.15334*, 2023. URL https://arxiv.org/abs/2305.15334.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023. URL https://arxiv.org/abs/2302.04761.

Zhi Rui Tam, Cheng-Kuang Wu, Yi-Lin Tsai, Chieh-Yen Lin, Hung-yi Lee, and Yun-Nung Chen. Let me speak freely? A study on the impact of format restrictions on performance of large language models. *arXiv preprint arXiv:2408.02442*, 2024. URL https://arxiv.org/abs/2408.02442.

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

Brandon T. Willard and Rémi Louf. Efficient guided generation for large language models. *arXiv preprint arXiv:2307.09702*, 2023. URL https://arxiv.org/abs/2307.09702.

Brandon T. Willard and Rémi Louf. Say what you mean: A response to "let me speak freely". https://blog.dottxt.ai/say-what-you-mean.html, 2024. dottxt blog.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927. doi: 10.1080/01621459.1927.10502953.