Externalized CPDAG Summaries Improve LLM Causal Deduction

Wentao Sun
Nokia Bell Labs
wentao.sun@nokia.com

Joao-Paulo Fontoura Nogueira
Institut Polytechnique de Paris
joao-paulo.fontoura_nogueira@telecom-sudparis.eu

Dominique Verchere
Nokia Bell Labs
dominique.verchere@nokia-bell-labs.com

Mathieu Acher
IRISA/Inria
mathieu.acher@irisa.fr

Alonso Silva
Nokia Bell Labs
alonso.silva@nokia-bell-labs.com

Abstract

Observational data rarely identify a single causal structure. Such structures are typically modeled as directed acyclic graphs (DAGs) whose arrows point from direct causes to their effects, and observed correlations and conditional independencies usually leave a whole family of DAGs that fit the data equally well. A causal claim follows from the data only if it holds in every one of them. Classical causal discovery methods such as the Peter–Clark (PC) algorithm summarize this family as a completed partially directed acyclic graph (CPDAG): an edge is directed when every compatible graph orients it the same way, and undirected when the data leave its direction open. On the Corr2Cause benchmark, large language models (LLMs) that reason in free text often lose track of this family and decide from a few local edges. We propose *Structured Thinking*, a two-turn method in which the model first summarizes the CPDAG in a constrained schema and then checks the claim against that summary. Without fine-tuning or an external solver, it raises Qwen3.5-27B from 77.5 to 86.5 F_1 (mean of three seeds) over a baseline that receives explicit PC-algorithm instructions, whereas a second turn of free-text analysis does worse than that baseline. The gain holds on test splits with renamed variables or paraphrased hypotheses and on other models. Given only the graph and the claim, the model's answer flips when the graph is replaced by a valid CPDAG under which the claim has the opposite truth value. Moreover, replacing the model's graphs with reference CPDAGs on a fixed subset shows that most remaining errors arise in building the graph rather than in reading it. Making causal structure explicit can help LLMs reason more reliably, provided the intermediate representation is itself trustworthy.

1 Introduction

Figure 1 shows a test item from Corr2Cause [Jin et al., 2024]. The premise describes a closed system of five variables through the correlations and conditional independencies among them, and the hypothesis asks whether A and C have at least one common effect. Many causal graphs can produce the same statistics, so the hypothesis counts as true only if it holds in every one of them. Given explicit instructions for causal discovery, Qwen3.5-27B reasons for 47k characters of free text,

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Input Corr2Cause test item 590 · ground truth: **Yes**

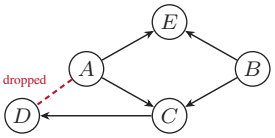
Premise. Suppose there is a closed system of 5 variables, A, B, C, D and E. [. . .] A correlates with C. [. . .] However, A is independent of B. B and D are independent given A and C. [. . .] D and E are independent given A and C.

Hypothesis. There exists at least one collider (i.e., common effect) of A and C.

(a) BaselineEnhanced one turn

free-text PC reasoning (excerpt; 47k characters in full)

"If edge (A,D) exists, consider the path B-E-A-D. From above, $B \to E \leftarrow A$. Conditioning on A opens this path (A is a collider). [. . .] Therefore, the edge (A,D) cannot exist [. . .] D: $A \to D$? No (edge ruled out). [. . .] Thus, there is no variable Z that is a common effect of A and C."



structure stated in its reasoning

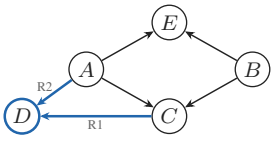
Answer: **No** ✗

(b) Structured Thinking two turns

Turn 1 schema-constrained `CausalAnalysis` tool call

```
CausalAnalysis(
  variables    = [A, B, C, D, E],
  skeleton     = [A-C, A-D, A-E, B-C, B-E, C-D],
  non_adjacent = [A-B|, B-D|A,C, C-E|A,B, D-E|A,B],
  v_structures = [A->C<-B, A->E<-B],
  directed     = [A->C, B->C, A->E, B->E, C->D, A->D],
  undirected   = [])
```

Turn 2 the summary is returned with a graph view; the model queries it



returned CPDAG

"The edges are $A \to D$ and $C \to D$. [. . .] Since the directions [. . .] are fixed in the CPDAG (determined by v-structures and Meek's rules), this collider exists in all DAGs in the Markov equivalence class."

Answer: **Yes** ✓

Figure 1: **Free-text reasoning versus Structured Thinking on Corr2Cause test item 590** (Qwen3.5-27B, abridged verbatim outputs). With the PC-algorithm prompt alone (left), the model misapplies d-separation (red), drops the edge A–D, and finds no common effect of A and C. Structured Thinking (right) writes the CPDAG into a schema-constrained `CausalAnalysis` call; Turn 2 reads off $A \to D \leftarrow C$ and checks that this collider holds in every DAG of the equivalence class. R1 and R2 mark orientations forced by Meek's rules.

misapplies d-separation (the graphical test for conditional independence), drops the edge A–D, and answers NO. Asked first to write down the graph, the same model finds the collider $A \to D \leftarrow C$ and answers YES.

The difficulty is that the label depends on a hidden object that the premise never states. Causal structure is modeled as a directed acyclic graph (DAG), whose arrows point from direct causes to their effects. Under standard Markov and faithfulness assumptions, the premise determines a Markov equivalence class: the set of DAGs that encode exactly the stated independencies. Classical causal discovery procedures, such as the Peter–Clark (PC) algorithm [Spirtes et al., 2001, Kalisch and Bühlmann, 2007], summarize this class as a completed partially directed acyclic graph (CPDAG). An edge is directed in the CPDAG when every graph in the class orients it the same way, and undirected when the class contains both orientations. A hypothesis is labeled YES only if it holds in every graph of the class, so each Corr2Cause question is a query on this CPDAG. Answering it in prose requires the model to hold many facts at once: which pairs of variables are adjacent, which triples form colliders, and which edge directions are forced. Free text gives these facts no fixed place, and a long trace such as the one in Figure 1 loses some of them along the way.

In earlier work, we found that asking a model to build a knowledge-graph style intermediate improves causal generalization on Corr2Cause [Sun et al., 2025]. However, that study left open what drives the gain: the structure itself, the causal instructions that come with it, or simply the extra turn spent before answering.

To separate these factors, we introduce *Structured Thinking*, a two-turn method that splits building the causal graph from reading it (Figure 1, right). In Turn 1, the model follows PC-algorithm instructions and writes a `CausalAnalysis` object listing the variables, the skeleton, the non-adjacent pairs with their separating sets, the v-structures (colliders), and the directed and undirected edges. For Qwen-family models, decoding is constrained to this schema; API models receive the same schema through tool calling. In Turn 2, the object is returned to the model, which checks the hypothesis against the graph. The model builds and reads its own graph: nothing is fine-tuned and no solver is called.

Our main baseline, BASELINEENHANCED, receives a detailed PC-algorithm walk-through and answers in one turn. On the full Corr2Cause test set, Structured Thinking raises Qwen3.5-27B from 77.5 to 86.5 F_1(YES), averaged over three seeds, and our controls trace this gain to the graph rather than to the extra turn or the graph vocabulary. When Turn 1 writes the same PC analysis as free text, the model falls below the baseline, mostly by adding false positives. A single-turn prompt with the structured instructions but no emitted graph stays near the baseline, and removing the hard schema constraint costs much of the gain, largely because Turn 2 then often receives no usable graph. Turn 2 also relies on what the graph says. When the premise is hidden and the graph is replaced by a valid counterfactual CPDAG that reverses the true answer, Turn 2 switches to the new answer in 90% of cases. Substituting reference CPDAGs for the model's own graphs removes most remaining errors on a fixed subset, mainly by supplying graphs the model failed to produce, so the main bottleneck is graph construction rather than graph reading.

The gain carries over to Qwen3.6-27B and Qwen3.5-9B and to test splits with renamed variables or paraphrased hypotheses; it is smaller for GPT-5.4-mini and Gemma-4-31B, whose baselines are already strong. Beyond Corr2Cause, a typed graph intermediate beats a matched prose intermediate on StepGame spatial deduction [Shi et al., 2022, Li et al., 2024]. It fails on CLUTRR kinship reasoning [Sinha et al., 2019], where even reference graphs give low accuracy because composing kinship relations over the graph is itself hard for the model.

Contributions. We treat causal deduction on Corr2Cause as building and querying a CPDAG, and show that a language model can do both on its own. Using an instruction-rich baseline, a two-turn prose control, and ablations, we show that the gain cannot be explained by causal instructions or an extra turn alone. Using counterfactual graphs, reference-graph substitution, a validity audit, and two transfer tasks, we show that Turn 2 uses the content of the graph, locate most remaining errors in graph construction, and identify where the approach stops working.

2 Background and Related Work

CPDAGs and causal deduction. The PC algorithm [Kalisch and Bühlmann, 2007, Spirtes et al., 2001] removes edges using conditional independencies, orients v-structures, and propagates Meek rules [Meek, 1995]. Its output is a CPDAG: directed edges are forced across the equivalence class, while undirected edges encode genuine ambiguity [Pearl, 2009]. Corr2Cause uses this CPDAG as the hidden object behind each label. This is why the task is a better stress test than asking whether two variables are merely associated: an answer can be wrong even when the model finds a plausible DAG, if another DAG in the same equivalence class falsifies the hypothesis.

LLMs and structural causality. Surveys and benchmarks consistently find structural causal reasoning harder for LLMs than commonsense causal association [Kiciman et al., 2024, Ma, 2025, Zečević et al., 2023, Willig et al., 2022, Long et al., 2023, Chi et al., 2024]. Yang et al. [2024] argue that many causal benchmarks can be partially solved by retrieving familiar domain knowledge; Corr2Cause is useful precisely because its labels are generated from independence constraints and PC-algorithm semantics. Complementary benchmarks such as CLadder, CLEAR-3K, and CausalARC evaluate questions across Pearl's ladder of causation, natural-language causal explanation, and SCM-world reasoning rather than CPDAG deduction [Jin et al., 2023, Liu et al., 2026, Maasch et al., 2025]. Prompting strategies improve performance [Sgouritsa et al., 2024, Liu et al., 2025, Bagheri et al., 2024], but still leave a gap to algorithmic ground truth, especially as the number of variables grows.

Structured intermediates. The closest precursor to this work builds a knowledge-graph style intermediate for Corr2Cause and shows that explicit structure can improve causal generalization [Sun et al., 2025]. We make the intermediate more specific: it is a CPDAG summary, emitted under regex- or JSON-schema-constrained decoding [Willard and Louf, 2023, Dong et al., 2024], and compared against a strong direct-answer PC baseline. Concurrent modular prompting decomposes Corr2Cause into prose sub-answers [Kadziolka and Salehkaleybar, 2025]; C2P similarly uses causal prompting steps but is not a constrained CPDAG-state method [Bagheri et al., 2024]. A prose sub-answer can be locally plausible but hard to audit; a typed graph has explicit fields whose content can be removed, scrambled, or queried.

Faithfulness of reasoning traces. Recent work questions whether free-form CoT is causally used by the final answer, using pre-CoT probes, causal-bypass diagnostics, causal sufficiency/necessity tests, and judge benchmarks for process faithfulness [Cox et al., 2026, Sathyanarayanan et al., 2026, Yu et al., 2025, Mittal and Arike, 2026, Fu et al., 2025]. Our probes are behavioral rather than activation-level: after Turn 1 emits a graph, we intervene on graph fields and measure Turn 2 degradation. They therefore test content dependence of the emitted state, not full mechanistic faithfulness of every token.

Tool use, programs, and positioning. Tool-augmented language models [Parisi et al., 2022, Yao et al., 2023, Schick et al., 2023] usually call external functions, while program-aided prompting [Gao et al., 2023, Chen et al., 2023] offloads computation to code. Structured Thinking instead uses the tool interface to record the model's own schema-constrained analysis; no symbolic causal solver supplies the answer. Our contribution is a constrained CPDAG-summary intermediate, evaluated against direct-answer and two-turn prose PC baselines and tested through interventions on graph content.

3 Method

3.1 Task

An instance has a premise P listing marginal dependencies and conditional independencies $X_i \perp X_j \mid Z$, plus a hypothesis H such as "X_a is a parent of X_b." Under the faithfulness and Markov assumptions, P identifies a CPDAG $\mathcal{G}$ and its Markov equivalence class $\mathrm{MEC}(\mathcal{G})$. The label is

$$y(P, H) \;=\; \textsc{Yes} \iff \forall D \in \mathrm{MEC}(\mathcal{G}) : H \text{ holds in } D,$$

i.e. the hypothesis must survive *all* valid DAG completions. We report F_1 on the positive class as the primary metric because the ID split is imbalanced toward NO ($180/1162$ positives), and report accuracy as a secondary metric.

3.2 Structured Thinking pipeline

Structured Thinking is a two-turn conversation. In Turn 1, the system prompt describes the PC algorithm and requires a `CausalAnalysis` object with fields `variables`, `skeleton`, `non_adjacent`, `v_structures`, `directed`, and `undirected`. For Qwen-family open-weight models, decoding is constrained by a schema-derived regex; for API models, the same schema is enforced through JSON-schema tool calling. Gemma uses the same structured interface without hard constrained decoding because the available backend was unreliable for that tokenizer. Thinking tokens, when used, appear before the constrained region. In Turn 2, the emitted object g is returned as a tool-result message with a compact DOT view, and the model answers H by checking adjacency, forced orientations, and directed reachability over all valid completions. The final YES/NO is extracted by regex.

The schema guarantees parseable fields; acyclicity, Meek closure, and CPDAG consistency are left to the model. We therefore distinguish three objects: a *candidate CPDAG summary* is what the model emits; a *graph-validated CPDAG* is a candidate that passes our validator (§7.1); and a *reference CPDAG* is built from the premise by a symbolic solver. The design pins both the intermediate *type* (a CPDAG summary rather than prose) and the Turn 2 *target* (a graph query rather than a restatement of P). We ablate the PC prompt, constrained decoding, and thinking tokens, and probe premise visibility and graph content, in §5–§6; full prompt and schema text appears in Appendix H–I.

Why this intermediate is useful. A CPDAG separates three decisions that free text tends to entangle: which pairs are adjacent, which triples are compelled colliders, and which orientations are forced across the equivalence class. Turn 2 can inspect this structure directly rather than rediscover it from the premise (Figure 1); the advantage is a better state representation, not more verbosity.

Constrained decoding. The constraint is applied to the *analysis object*, not to the free reasoning tokens that precede it. For Qwen-family models, the regex wraps a `<think>` block followed by a `<tool_call>` envelope. Removing the hard constraint keeps the tool-call template as a soft instruction, so the ablation asks whether syntactic enforcement itself matters beyond reminding the model to be structured.

3.3 Baselines and controls

The principal baseline, BASELINEENHANCED, is a strong direct-answer PC baseline. It receives an explicit PC-algorithm walk-through in a single turn and answers directly. The prompt covers skeleton removal from conditional independencies, collider orientation, Meek-rule propagation, closed-world treatment of unstated independencies, and the requirement that a positive hypothesis hold in every DAG in the Markov equivalence class. This comparison controls for giving the direct-answer model an explicit PC-algorithm scaffold.

As a prose-state control, we add TWOTURN-PROSE-PC. Turn 1 uses the same detailed PC scaffold as BASELINEENHANCED but asks for free-text analysis while withholding the final yes/no answer; Turn 2 returns that analysis and requests the answer. This control uses no tool call and no schema-constrained typed graph object. It tests whether the same PC scaffold plus a two-turn prose intermediate suffices.

We also include a single-turn structured-prompt diagnostic for Qwen3.6-27B, SINGLETURN-STRUCTPROMPT. This condition reuses the structured CPDAG-oriented instructions but does not require a tool call or schema-constrained graph output before answering. It tests whether merely mentioning the right graph vocabulary can replace writing out the graph itself.

4 Main Results

Our experiments answer four questions. *Effectiveness*: does a typed intermediate improve over a strong direct PC-instruction baseline and over a PC-scaffolded prose intermediate? *Robustness*: does the gain hold across seeds, models, perturbed wording, and graph size? *Validity and use*: how often is the emitted state a valid CPDAG, and does Turn 2 depend on its content? *Scope*: where does the recipe transfer, and where does it fail?

4.1 Setup

Models and decoding. Unless noted, open-weight models (`Qwen3.5-27B`, `Qwen3.6-27B`, `Qwen3.5-9B`, `Gemma-4-31B-it`) run under vLLM [Kwon et al., 2023] with T=0.6, top-p=0.95, `max_model_len`= 32768, and fixed seed; `gpt-5.4-mini` uses the OpenAI Responses API with `reasoning_effort="high"`; model identifiers are listed in Appendix K. Qwen-family Structured Thinking runs use schema-derived regex constraints via `xgrammar` [Dong et al., 2024]; direct baselines and TWOTURN-PROSE-PC do not emit a constrained CPDAG object. Gemma Structured is reported without constrained decoding because available backends were unreliable for that tokenizer.

Data and protocol. ID is the n=1162 Corr2Cause test split. We also use Corr2Cause's two perturbed test splits (n=2246 each): *Refactor-OOD* renames the variables, and *Paraphrase-OOD* rewrites the hypotheses. The headline Qwen3.5-27B ID comparison is a seed-matched rerun on seeds 42, 52, and 62 with the dataset hash, model revision, and decoding settings pinned; for each seed we report an exact McNemar test, and across seeds the mean and sample standard deviation. All remaining rows are single runs, with the original Qwen3.5-27B run retained for the controls, ablations, and probes.

4.2 Results across seeds, splits, and models

Across three matched seeds, Structured Thinking raises Qwen3.5-27B from 77.48 to 86.48 F_1(YES) ($+9.00 \pm 4.56$ pp) and from 93.55% to 95.81% accuracy (Table 1). All three gains are positive: +12.84, +3.96, and +10.20 pp on seeds 42, 52, and 62. Across seeds, Structured Thinking fixes 141 baseline errors and introduces 62 new ones; fixes outnumber new errors on every seed. The exact McNemar test is significant on seeds 42 and 62 ($p = 4.2 \times 10^{-6}$ and 1.1×10^{-3}) but not on seed 52 ($p = 0.22$), which has the strongest baseline. The direction of the gain is stable; its size varies from run to run.

The gain comes mainly from recall. Per seed, recall rises from 70.0, 75.6, and 69.4% to 86.7, 84.4, and 88.3%, while precision moves from 79.8, 87.2, and 86.2% to 88.1, 85.4, and 86.0%. On average, recall rises by 14.8 pp and precision by 2.1 pp. Structured Thinking is also more stable across seeds:

Table 1: **Main results on Corr2Cause**. F_1(YES) and accuracy (%). Top: seed-matched rerun, mean $\pm$ sample SD over three seeds (per seed in Table 6). Bottom: single runs. Δ is the F_1 change relative to the one-turn PC-instruction baseline for the same model and split. CD marks hard schema-constrained decoding of the Turn-1 graph.

Model	Method	Split	CD	F_1(Yes)	Acc.	ΔF_1
Qwen3.5-27B (3 seeds)	BaselineEnhanced	ID	–	77.48 ± 3.23	93.55 ± 0.95	—
	Structured Thinking	ID	✓	$\mathbf{86.48} \pm 1.36$	$\mathbf{95.81} \pm 0.41$	+9.00
Qwen3.5-27B	BaselineEnhanced	ID	–	73.01	92.43	—
	TwoTurn-Prose-PC	ID	–	67.55	89.41	−5.46
	Structured Thinking	ID	✓	**86.36**	**95.87**	+13.35
	BaselineEnhanced	Refactor-OOD	–	72.33	92.61	—
	Structured Thinking	Refactor-OOD	✓	**82.79**	**95.06**	+10.46
	BaselineEnhanced	Paraphrase-OOD	–	75.43	92.92	—
	Structured Thinking	Paraphrase-OOD	✓	**84.21**	**94.92**	+8.79
Qwen3.6-27B	BaselineEnhanced	ID	–	75.92	92.69	—
	SingleTurn-StructPrompt	ID	–	74.53	91.82	−1.39
	Structured Thinking	ID	✓	**85.71**	**95.52**	+9.79
Qwen3.5-9B	BaselineEnhanced	ID	–	39.18	84.77	—
	Structured Thinking	ID	✓	**59.56**	**88.90**	+20.38
GPT-5.4-mini	BaselineEnhanced	ID	–	85.39	95.61	—
	Structured Thinking	ID	✓	**88.58**	**96.47**	+3.19
Additional observation: no hard constrained decoding						
Gemma-4-31B	BaselineEnhanced	ID	–	85.38	95.70	—
	Structured Thinking	ID	×	**87.39**	**96.13**	+2.01

48 of 1162 items change prediction across the three seeds, against 111 for the baseline, whose unstable items all have five or more variables and are mostly positives.

TWOTURN-PROSE-PC reaches only 67.55 F_1, below both the one-turn baseline and Structured Thinking. It has higher recall than the one-turn baseline, but at the cost of many more false positives (71 versus 27; Appendix Table 5): with the same PC scaffold, a schema-free prose intermediate mostly shifts answers toward YES. In the same run, Structured Thinking reduces both false negatives (61→28) and false positives (27→20).

The same pattern holds beyond one Qwen release. On Qwen3.6-27B, Structured Thinking improves from 75.92 to 85.71 F_1 (+9.79 pp). A single-turn structured-prompt diagnostic reaches 74.53, so merely giving CPDAG-oriented wording without requiring an external graph object does not explain the gain.

The gain also persists when the variables are renamed (+10.46 pp) and when the hypotheses are paraphrased (+8.79 pp; exact McNemar $p = 3.5 \times 10^{-6}$ and 7.7×10^{-4}). The remaining rows compare each model with its own baseline. Structured Thinking improves Qwen3.5-9B by +20.38 pp and GPT-5.4-mini by +3.19 pp. Gemma-4-31B, which runs without hard constrained decoding, gains +2.01 pp. Gains are smaller when the direct baseline is already strong, as it is for GPT-5.4-mini and Gemma (about 85 F_1).

4.3 Gains concentrate on larger graphs

Figure 2 shows that the gain is larger in the hardest bin: on ID, the three-seed gain grows from +5.2 pp at n=5 to +11.9 pp at n=6, and the perturbed splits show the same pattern. At n=6, Structured Thinking is also nearly seed-invariant (83.2–84.7 F_1), while the baseline ranges from 69.2 to 76.2.

Pilot stress tests on larger graphs from other graph families (n=7 and 8; Appendix C) show that the advantage extends beyond the original $n \leq 6$ regime. On Qwen3.5-27B, Structured Thinking improves Random n=7 from 28.42 to 91.87 F_1, ALARM-derived n=7 from 15.38 to 88.73, and ASIA-derived n=8 from 9.30 to 60.00. In all three settings the direct baseline's recall is low (9–18%); for named-network-derived rows, source names and semantic variable names are withheld from the model.

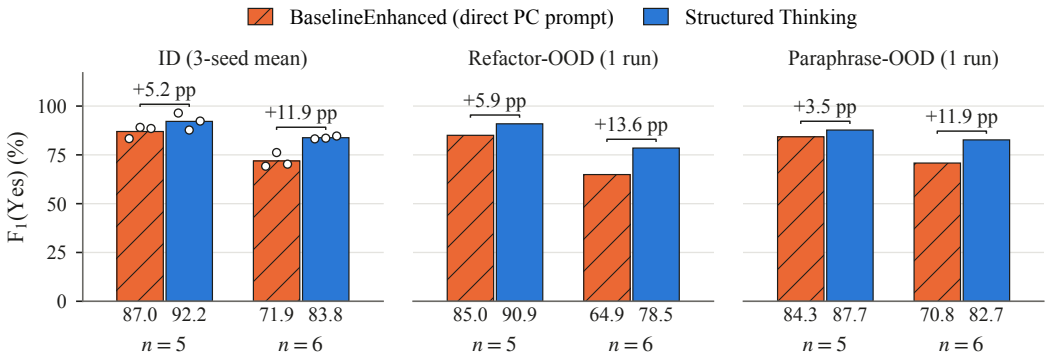


Figure 2: **F_1(YES) by variable count (Qwen3.5-27B).** ID bars are three-seed means, with dots for seeds 42, 52, and 62 from left to right; the perturbed splits are single runs. Brackets give the gain in pp. Bins with $n \leq 4$ contain at most three positives and are omitted.

Table 2: **Ablations on Qwen3.5-27B**. n=1162 ID split. Removing the PC-algorithm prompt collapses the method almost entirely; removing thinking or constrained decoding each causes a substantial drop.

Configuration	F_1(Yes)	Acc.	Δ vs. full
Structured Thinking (full)	**86.36**	95.87	—
–thinking	71.95	92.08	−14.41
–constrained decoding	76.88	93.37	−9.48
–PC-algorithm prompt	36.21	87.26	−50.15
BaselineEnhanced	73.01	92.43	−13.35

5 Ablations

We stress-test the Qwen3.5-27B gain by removing one design choice at a time (Table 2). The PC-algorithm prompt matters most: replacing it with a minimal "produce a causal analysis" instruction drops F_1 to 36.2, far below baseline. On top of that prompt, removing constrained decoding costs 9.5 pp and removing thinking costs 14.4 pp at 27 B.

In the constrained-decoding ablation the model still sees the tool-call syntax; only the hard regular-language constraint is removed. Our validity audit shows what the constraint buys. In matched full-split runs, it raises the share of instances with a usable graph from 74.5% to 97.2%, while the share of valid CPDAGs among usable graphs stays similar (81.1% without and 79.0% with the constraint). The clearest effect of the constraint is thus that Turn 2 reliably receives a graph.

A smaller-model factorial is reported in Appendix D: at 9 B, thinking becomes harmful while constrained decoding remains helpful, suggesting that explicit reasoning traces should be calibrated by model size.

6 Probes: is the emitted graph actually used?

We test whether Turn 2 uses the content of the graph with validity-preserving counterfactuals, then substitute reference CPDAGs to separate building the graph from reading it (Table 3; Qwen3.5-27B, ID split).

Validity-preserving counterfactuals. For 192 frozen ID instances, we pair the reference CPDAG with a counterfactual: another graph-validated CPDAG over the same variables under which the hypothesis has the opposite truth value. A serialization-only control keeps the reference graph and only permutes the order of its edges and field entries. All three graphs are valid CPDAGs, so this probe avoids the confound of graph scrambles, which can leave the space of valid CPDAGs (Appendix F). Given only the graph and the hypothesis, Turn 2 follows the counterfactual: the answer

Table 3: **Turn-2 probes on Qwen3.5-27B, ID split.** (a) Graph-only counterfactual probe: Turn 2 receives one of three graph-validated CPDAGs and the hypothesis, without the premise; flips are changes from the answer given with the reference CPDAG (all values in %). (b) Accuracy (%) on 128 frozen instances (seed 62) when Turn 2 receives the model's own candidate graph or the reference CPDAG, with or without the premise; a missing graph counts as an error.

(a) Counterfactual probe, 192 triplets

Graph given to Turn 2	Answer flips	Flips toward graph truth	Acc. vs. graph truth
Reference CPDAG	—	—	96.4
Serialization-only control	1.0	—	95.3
Valid counterfactual CPDAG	90.1	90.1	93.8

(b) Graph construction versus graph reading, n=128

Graph given to Turn 2	Without premise	With premise
Candidate graph (model's own)	89.06	92.97
Reference CPDAG	94.53	96.88

flips toward the new graph's truth in 173 of 192 triplets (90.1%), against 2 of 192 (1.0%) for the serialization control (exact McNemar $p = 6.7 \times 10^{-52}$). When the original premise is also shown, the counterfactual graph contradicts it, and the effect shrinks but stays well above control (28.1% target-consistent flips versus 3.1% control changes). Turn 2 thus weighs the premise and the graph together, and the graph's content, not its formatting, moves the answer.

Building the graph versus reading it. Table 3b substitutes solver-built reference CPDAGs for the model's own graphs. With the premise present, reference graphs raise accuracy from 92.97% to 96.88%. The gain comes from the 6 instances where the model emitted no usable graph: on the 122 instances with a usable candidate graph, both arms give the same answer on every instance. Replications on seeds 42 and 52 agree: reference CPDAGs reduce the errors from 15 to 5 and from 18 to 4, chiefly by supplying missing graphs. On this subset, most errors therefore arise in Turn 1, chiefly as missing graphs, while Turn 2 reads a usable graph reliably but not perfectly: with the reference CPDAG and the premise, it still errs on 4 of 128 instances. A full-split analysis agrees: among the 945 graph-validated CPDAGs of one full run (81.3% of instances), the graph itself implies the correct label in 98.4% of cases, and Turn 2 agrees with its graph in 98.7%.

7 Analysis and Practitioner Findings

7.1 Graph validity and error localization

Graph validity. Our validator checks the graph-theoretic conditions of a CPDAG: consistent fields without duplicates, a DAG extension whose colliders are exactly the emitted v-structures, and agreement of the emitted directed and undirected edges with the CPDAG of that equivalence class, obtained by orienting the colliders and closing under Meek's rules (Appendix E). It does not check separating-set tokens or consistency with the premise. On two full ID runs, 76.8% and 81.3% of test instances yield a graph-validated CPDAG; without hard constraints the rate falls to 60.4%, mostly because fewer graphs are produced (§5).

Automatic CPDAG-field audit. The emitted fields agree closely with the reference CPDAG (Table 4): on the full ID and Refactor-OOD splits, skeleton F_1 is 0.96, directed-edge F_1 is 0.88–0.90, and 75.9% and 74.4% of graphs match the reference exactly. The audit also shows the remaining gap between schema validity and graph-semantic validity: graph-consistency violations occur in 18.4% of ID outputs and 20.7% of Refactor-OOD outputs, with Meek-closure violations in 5.1% and 6.0%, respectively. These failures are not uniformly catastrophic: many are local or irrelevant to the queried hypothesis, and Turn 2 still sees the premise and hypothesis. Nevertheless, graph quality predicts downstream correctness. ID accuracy drops from 97.2% when the emitted CPDAG exactly matches the reference to 93.9% for consistent but non-exact graphs and 91.1% for summaries with graph-consistency violations; Refactor-OOD shows the same ordering (96.6%, 92.8%, 90.1%).

Table 4: **Automatic CPDAG-field audit** for Structured Thinking on full ID and Refactor-OOD splits. Graph-consistency violations are graph-semantic failures among schema-valid outputs, including field conflicts, v-structure inconsistencies, missing closure, or no valid DAG extension.

Audit metric	ID	Refactor-OOD
Skeleton F_1	0.960	0.959
Directed-edge F_1	0.896	0.884
V-structure F_1	0.923	0.920
Exact CPDAG match	75.9%	74.4%
Graph-consistency violations	18.4%	20.7%
Acc. when graph exact	97.2%	96.6%
Acc. with consistency violation	91.1%	90.1%

A 32-case manual diagnostic in Appendix L gives the same qualitative picture: many residual errors are already visible in the emitted graph, while others come from Turn-2 readout or MEC-ambiguity handling.

7.2 What the controls establish

None of these controls reproduces the gain. BASELINEENHANCED supplies explicit PC instructions, TWOTURN-PROSE-PC adds a prose intermediate, and SINGLETURN-STRUCTPROMPT adds CPDAG-oriented wording without an emitted graph; all score well below Structured Thinking (§4). The probes show that Turn 2 follows the content of the graph and is largely insensitive to its serialization (§6).

7.3 Transfer beyond Corr2Cause

Beyond CPDAG queries, we compare a typed graph intermediate with a matched two-turn prose intermediate on two GPT-5.6 snapshots (Appendix G). On StepGame spatial deduction [Shi et al., 2022, Li et al., 2024], the typed graph wins on both snapshots: 83.1% versus 70.6% accuracy on `gpt-5.6-terra` and 71.4% versus 19.4% on `gpt-5.6-luna` (n=360 each; exact McNemar $p \leq 1.9 \times 10^{-6}$). Answering directly from the story is near ceiling on this task (99.7% and 99.4%) and remains stronger, so StepGame tests typed against prose externalization, not the value of an intermediate over direct answering. On CLUTRR kinship reasoning [Sinha et al., 2019], the recipe fails: typed graphs fall well below the prose intermediate (17.8% and 22.2% versus 70.6% and 55.0%), and even reference graphs reach only 34%, so our graph representation and readout do not fit that task.

7.4 Cost, reproducibility, and deployment

Structured Thinking requires two model calls and, for Qwen-family local runs, a schema-constrained decoder, but no fine-tuning, learned verifier, or symbolic causal solver at inference time. The token footprint is roughly 2× the one-turn baseline and wall-clock is about 1.5× in batched vLLM runs. API-backed reasoning models require care: on GPT-5.4-mini, strict tool forcing compressed reasoning before the tool call and `tool_choice="auto"` restored performance, whereas on a current GPT-5.6 snapshot forcing made no detectable difference (Appendix A.1). The headline comparison uses the open-weight Qwen3.5-27B; code, data, prompts, schemas, and metadata are publicly released (Appendix M).

8 Limitations

Our causal claims are scoped to CPDAG-style deduction from conditional independence statements. StepGame shows that typed externalization can beat a prose intermediate on another structured-deduction task, and CLUTRR shows a task where our graph-plus-readout pipeline fails. Transfer to SCM, interventional, and counterfactual queries, as in CLadder [Jin et al., 2023], or to real-world causal claims is not established.

The headline Qwen3.5-27B comparison rests on three seeds. The gain is positive on each, but its size varies from $+4.0$ to $+12.8$ pp and is not individually significant on seed 52; with three seed-level observations, we report the spread descriptively. The perturbed splits, larger-graph stress tests, other models, and controls are single runs and do not establish robustness across seeds.

The evidence is uneven across the two turns. For graph construction, the validity audit covers full test splits, but 19–23% of test instances do not yield a graph-validated CPDAG, and the validator does not check separating-set tokens or consistency with the premise. For graph reading, the reference-CPDAG and counterfactual probes cover frozen subsets of 128 and 192 instances.

Finally, schema constraints, graph-field names, tool-result formatting, and Turn-2 query instructions are bundled in the method, and our controls do not separate their contributions; the baseline's collider rule also omits the separating-set condition stated in the structured prompt (Appendix H.3). Lightly structured prose intermediates would sharpen this decomposition.

9 Conclusion

Structured Thinking asks a language model to write down the causal graph behind a Corr2Cause question before answering it, and on Qwen3.5-27B it raises F_1 by 9.0 points on average over a baseline that already receives the PC algorithm in its prompt. Our controls show where this gain comes from. Causal instructions and an extra turn are not enough on their own: the instructions are already in the baseline, CPDAG vocabulary without an emitted graph does not improve on it, and a second turn of free-text analysis makes the model worse. The improvement appears once the intermediate is a graph in a fixed schema, and replacing that graph with a different valid one changes the answer accordingly.

Separating graph construction from graph reading also shows where to improve next. Given a valid CPDAG, the model nearly always reads it correctly, so most of its errors start earlier: a sizable share of its own graphs are missing or fail our validity check, and missing graphs account for most errors on our fixed subset. Pairing constrained decoding with a graph validator, or letting the model repair its graph before Turn 2 reads it, are natural next steps.

The same separation marks where the approach stops working. It helps when the answer can be read off the graph; on CLUTRR, composing kinship relations over the graph remains hard for the model even with reference graphs, and a prose intermediate does better. More generally, an explicit causal structure makes a model's reasoning more reliable when the model can build that structure correctly and the answer follows from it directly.

Acknowledgments and Disclosure of Funding

We thank the reviewers and the area chair for their constructive feedback. This work was funded by Nokia Bell Labs. Wentao Sun's doctoral research is supported by an ANRT CIFRE fellowship with Nokia Bell Labs. Competing interests: Wentao Sun, Dominique Verchere, and Alonso Silva are employees of Nokia Bell Labs; the authors declare no other competing interests.

References

Abdolmahdi Bagheri, Matin Alinejad, Mahdi Dehshiri, Kevin Bello, and Alireza Akhondi-Asl. C^2P: Featuring large language models with causal reasoning. *arXiv preprint arXiv:2407.18069*, 2024.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research (TMLR)*, 2023. arXiv:2211.12588.

Haoang Chi, He Li, Wenjing Yang, Feng Liu, Long Lan, Xiaoguang Ren, Tongliang Liu, and Bo Han. Unveiling causal reasoning in large language models: Reality or mirage? In *Advances in Neural Information Processing Systems*, volume 37, pages 96640–96670, 2024. doi: 10.52202/079017-3064. URL `https://openreview.net/forum?id=1IU3P8VDbn`.

Kyle Cox, Darius Kianersi, and Adrià Garriga-Alonso. Decoding answers before chain-of-thought: Evidence from pre-cot probes and activation steering. *arXiv preprint arXiv:2603.01437*, 2026. URL https://arxiv.org/abs/2603.01437.

Yixin Dong, Charlie F. Ruan, Yaxing Cai, Ruihang Lai, Ziyi Xu, Yilong Zhao, and Tianqi Chen. XGrammar: Flexible and efficient structured generation engine for large language models. *arXiv preprint arXiv:2411.15100*, 2024. MLSys 2025.

Jiarun Fu, Lizhong Ding, Hao Li, Pengqi Li, Qiuning Wei, Xu Chen, Yurong Cheng, and Ye Yuan. Is it necessary to inject causality into chain-of-thought reasoning? OpenReview submission, 2025. URL https://openreview.net/forum?id=pY7NPetIsZ.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. In *International Conference on Machine Learning (ICML)*, 2023. arXiv:2211.10435.

Zhijing Jin, Yuen Chen, Felix Leeb, Luigi Gresele, Ojasv Kamal, Zhiheng Lyu, Kevin Blin, Fernando Gonzalez Adauto, Max Kleiman-Weiner, Mrinmaya Sachan, and Bernhard Schölkopf. CLadder: Assessing causal reasoning in language models. In *Advances in Neural Information Processing Systems*, volume 36, pages 31038–31065, 2023.

Zhijing Jin, Jiarui Liu, Zhiheng Lyu, Spencer Poff, Mrinmaya Sachan, Rada Mihalcea, Mona T. Diab, and Bernhard Schölkopf. Can large language models infer causation from correlation? In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=vqIH0ObdqL. arXiv:2306.05836.

Kacper Kadziolka and Saber Salehkaleybar. Causal reasoning in pieces: Modular in-context learning for causal discovery. *arXiv preprint arXiv:2507.23488*, 2025.

Markus Kalisch and Peter Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8:613–636, 2007.

Emre Kiciman, Robert Ness, Amit Sharma, and Chenhao Tan. Causal reasoning and large language models: Opening a new frontier for causality. *Transactions on Machine Learning Research (TMLR)*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=mqoxLkX210. Selected for presentation at ICLR 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*, 2023. arXiv:2309.06180.

Fangjun Li, David C. Hogg, and Anthony G. Cohn. Advancing spatial reasoning in large language models: An in-depth evaluation and enhancement using the StepGame benchmark. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18500–18507, 2024. doi: 10.1609/aaai.v38i17.29811.

Naiming Liu, Richard Baraniuk, and Shashank Sonkar. CLEAR-3K: Assessing causal explanatory capabilities in language models. In *Findings of the Association for Computational Linguistics: EACL 2026*, pages 3121–3136, Rabat, Morocco, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.findings-eacl.163. URL https://aclanthology.org/2026.findings-eacl.163/.

Xiao Liu, Da Yin, Chen Zhang, Dongyan Zhao, and Yansong Feng. Eliciting and improving the causal reasoning abilities of large language models with conditional statements. *Computational Linguistics*, 51(2):467–504, 2025.

Stephanie Long, Tibor Schuster, and Alexandre Piché. Can large language models build causal graphs? *arXiv preprint arXiv:2303.05279*, 2023. Presented at the NeurIPS 2022 Workshop on Causal Machine Learning for Real-World Impact.

Jing Ma. Causal inference with large language model: A survey. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5901–5913, Albuquerque, New Mexico, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-naacl.327. URL https://aclanthology.org/2025.findings-naacl.327/.

Jacqueline R. M. A. Maasch, John Kalantari, and Kia Khezeli. CausalARC: Abstract reasoning with causal world models. *arXiv preprint arXiv:2509.03636*, 2025. NeurIPS 2025 Workshop: Bridging Language, Agent, and World Models (LAW).

Christopher Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 403–410. Morgan Kaufmann, 1995.

Avni Mittal and Rauno Arike. C2-Faith: Benchmarking LLM judges for causal and coverage faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2603.05167*, 2026.

Aaron Parisi, Yao Zhao, and Noah Fiedel. TALM: Tool augmented language models. *arXiv preprint arXiv:2205.12255*, 2022.

Judea Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2nd edition, 2009. doi: 10.1017/CBO9780511803161.

Anish Sathyanarayanan, Aditya Nagarsekar, and Aarush Rathore. Bypassing the rationale: Causal auditing of implicit reasoning in language models. *arXiv preprint arXiv:2602.03994*, 2026.

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. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2302.04761.

Eleni Sgouritsa, Virginia Aglietti, Yee Whye Teh, Arnaud Doucet, Arthur Gretton, and Silvia Chiappa. Prompting strategies for enabling large language models to infer causation from correlation. *arXiv preprint arXiv:2412.13952*, 2024.

Zhengxiang Shi, Qiang Zhang, and Aldo Lipani. StepGame: A new benchmark for robust multi-hop spatial reasoning in texts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11321–11329, 2022. doi: 10.1609/aaai.v36i10.21383.

Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L. Hamilton. CLUTRR: A diagnostic benchmark for inductive reasoning from text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4505–4514, 2019. doi: 10.18653/v1/D19-1458.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. Adaptive Computation and Machine Learning. MIT Press, 2nd edition, 2001. ISBN 978-0-262-19440-2. doi: 10.7551/mitpress/1754.001.0001. URL https://direct.mit.edu/books/monograph/2057/Causation-Prediction-and-Search.

Wentao Sun, João Paulo Nogueira, and Alonso Silva. Structured thinking matters: Improving LLMs generalization in causal inference tasks. *arXiv preprint arXiv:2505.18034*, 2025.

Brandon T. Willard and Rémi Louf. Efficient guided generation for large language models. *arXiv preprint arXiv:2307.09702*, 2023.

Moritz Willig, Matej Zečević, Devendra Singh Dhami, and Kristian Kersting. Can foundation models talk causality? In *UAI 2022 Workshop on Causal Representation Learning*, 2022. arXiv:2206.10591.

Linying Yang, Vik Shirvaikar, Oscar Clivio, and Fabian Falck. A critical review of causal reasoning benchmarks for large language models. *AAAI 2024 Workshop on "Are Large Language Models Simply Causal Parrots?"*, 2024. arXiv:2407.08029.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023. URL `https://openreview.net/forum?id=WE_vluYUL-X`. arXiv:2210.03629.

Xiangning Yu, Zhuohan Wang, Linyi Yang, Haoxuan Li, Anjie Liu, Xiao Xue, Jun Wang, and Mengyue Yang. Causal sufficiency and necessity improves chain-of-thought reasoning. *arXiv preprint arXiv:2506.09853*, 2025.

Matej Zečević, Moritz Willig, Devendra Singh Dhami, and Kristian Kersting. Causal parrots: Large language models may talk causality but are not causal. *Transactions on Machine Learning Research (TMLR)*, 2023. arXiv:2308.13067.

A Additional diagnostics

Table 5: **Confusion matrices, Qwen3.5-27B ID split (n=1162), single run of Table 1.** Structured Thinking reduces both false negatives (61 → 28) and false positives (27 → 20). The PC-scaffolded prose control increases recall but introduces many false positives.

		Predicted		TP	FN	FP
Method	**Actual**	NO	YES			
BaselineEnhanced	NO	955	27	119	61	27
	YES	61	119			
TwoTurn-Prose-PC	NO	911	71	128	52	71
	YES	52	128			
Structured Thinking	NO	962	20	**152**	**28**	**20**
	YES	28	152			

A.1 API configuration sensitivity

On API-backed reasoning models, strict `tool_choice="required"` can change the reasoning regime. In our initial GPT-5.4-mini run, name-forcing the `CausalAnalysis` tool reduced F_1 to 74.7 (-10.7 pp vs. the 85.4 baseline): median reasoning tokens on n=6 fell from roughly 5000 to roughly 800, and 54% of emitted CPDAGs had zero directed edges. Switching to `tool_choice="auto"` plus a prompt-level instruction to call the tool preserved the same schema while restoring the reasoning budget (roughly 5600 tokens on n=6), raising the directed-edge rate from 42% to 96% and recovering the reported 88.6 F_1. The two configurations differ only in this API flag and prompt-level instruction.

GPT-5.4-mini has since been retired, so we repeated the comparison on a current snapshot (`gpt-5.6-terra`) with an identical schema, interface, and instance set (n=1162 per arm). Fail-closed accuracy is 90.79% with `tool_choice="auto"` and 91.05% with name-forcing (-0.26 pp for auto minus forcing; paired bootstrap 95% CI $[-2.32, +1.72]$; exact McNemar $p = 0.867$), with the same Turn-2 coverage in both arms (1098/1162). Forcing the tool has no detectable effect on this snapshot.

B Full-ID seed robustness

Because the local Qwen runs use stochastic decoding, we rerun the principal Qwen3.5-27B ID comparison on three seeds with the dataset hash, model revision (`fc05daec`), and decoding settings pinned, and with both methods run on the same seeds. Table 6 reports the full ID split (n=1162). On seed 42, it reproduces the gap of the single run in Table 1 (+12.84 versus +13.35 pp).

Recall rises on every seed, by 9 to 19 pp. Precision rises by 8.4 pp on seed 42 and falls by 1.8 and 0.3 pp on seeds 52 and 62. The baseline's run-to-run variation is concentrated on large positive instances: 111 items receive different baseline predictions across the three seeds, all with five or more variables (79 with six) and 67 of them positive, against 48 items for Structured Thinking. These are also the instances where Structured Thinking helps most (Figure 2).

Table 6: **Seed-matched rerun for Qwen3.5-27B on Corr2Cause ID (n=1162).** Values are percentages. "Only S / only B" counts items answered correctly by Structured Thinking only and by the baseline only; p is the exact McNemar test on these discordant pairs. Means are followed by sample SDs over seeds.

Seed	Method	F_1(Yes)	Acc.	Prec.	Rec.	Only S / only B	p
42	Baseline	74.56	92.60	79.75	70.00	60 / 19	4.2×10^{-6}
42	Structured	87.39	96.13	88.14	86.67		
52	Baseline	80.95	94.49	87.18	75.56	32 / 22	0.22
52	Structured	84.92	95.35	85.39	84.44		
62	Baseline	76.92	93.55	86.21	69.44	49 / 21	1.1×10^{-3}
62	Structured	87.12	95.96	85.95	88.33		
Baseline, mean		77.48 ± 3.23	93.55 ± 0.95	84.38 ± 4.04	71.67 ± 3.38		
Structured, mean		86.48 ± 1.36	95.81 ± 0.41	86.49 ± 1.45	86.48 ± 1.95		
Paired difference		$+9.00 \pm 4.56$	$+2.27 \pm 1.34$	$+2.11 \pm 5.49$	$+14.81 \pm 5.25$		

C Larger-graph stress tests

Table 7: **Pilot larger-graph stress tests.** Values are percentages except N and #Yes. These single-seed rows use graph families outside the original Corr2Cause $n \leq 6$ range.

Setting	Model	Method	N	#Yes	Prec.	Rec.	Acc.	F_1
Random n=7 pilot	Qwen3.5-27B	Baseline	300	150	67.50	18.00	54.67	28.42
Random n=7 pilot	Qwen3.5-27B	Structured	300	150	97.74	86.67	92.33	**91.87**
Random n=7 pilot	Qwen3.5-9B	Baseline	300	150	75.76	16.67	55.67	27.32
Random n=7 pilot	Qwen3.5-9B	Structured	300	150	70.37	25.33	57.33	**37.25**
ALARM-derived n=7	Qwen3.5-27B	Baseline	300	150	43.75	9.33	48.67	15.38
ALARM-derived n=7	Qwen3.5-27B	Structured	300	150	97.60	81.33	89.67	**88.73**
ALARM-derived n=7	Qwen3.5-9B	Baseline	300	150	47.83	14.67	49.33	22.45
ALARM-derived n=7	Qwen3.5-9B	Structured	300	150	65.38	22.67	55.33	**33.66**
ASIA-derived n=8	Qwen3.5-27B	Baseline	168	20	8.70	10.00	76.79	9.30
ASIA-derived n=8	Qwen3.5-27B	Structured	168	20	60.00	60.00	90.48	**60.00**
ASIA-derived n=8	Qwen3.5-9B	Baseline	40	20	100.00	5.00	52.50	9.52
ASIA-derived n=8	Qwen3.5-9B	Structured	40	20	100.00	20.00	60.00	**33.33**

The 27B gains are large on all three stress settings, while the 9B rows remain positive but capacity-limited. The ASIA-derived 9B row is small (N=40). The source graph families are used to generate larger CPDAG-query instances; source-network names and semantic variable names are withheld from the model, and prompts use anonymized variable labels. Seed metadata and confusion matrices are included in the released evaluation files.

D Scale interaction diagnostic

Table 8: **Qwen3.5-9B factorial** (n=1162, ID split). Thinking is net-harmful to Structured Thinking at 9 B, reversing its Qwen3.5-27B behaviour (Table 2).

Configuration	F_1(Yes)	Acc.
Structured Thinking (full)	59.56	88.90
–thinking	**64.46**	**89.85**
–constrained decoding	45.85	85.97
BaselineEnhanced	39.18	84.77
–thinking	45.58	86.23

Thinking helps at 27 B but hurts at 9 B: disabling it improves Structured Thinking by $+4.9$ pp and the baseline by $+6.4$ pp. Constrained decoding shows the opposite pattern, helping more at 9 B (13.7 pp) than at 27 B (9.5 pp). This suggests a practical calibration rule: enable thinking only when a validation split shows that the model's trace is reliable at the target scale.

E Graph validity audit

The validator works offline on each candidate summary. It rejects duplicate or contradictory entries and checks that the directed and undirected edges partition the skeleton. It then requires a DAG extension: an acyclic orientation of the undirected edges whose colliders are exactly the emitted v-structures. Finally, it orients those colliders on the skeleton, closes the result under Meek's rules R1–R3, which yields the CPDAG of the equivalence class, and requires this CPDAG to match the emitted directed and undirected edges exactly; this rejects both missing and superfluous orientations. A missing or unparsable summary counts as invalid (fail-closed). The validator does not interpret separating-set tokens and does not check the graph against the premise. All 1162 reference CPDAGs and all 576 graphs of the counterfactual probe pass it. A weaker check that keeps the emitted orientations fixed would accept over-oriented graphs (for example $A \to B \to C$ in place of $A - B - C$) and raise each validity rate by 2.5 to 3.3 points, without changing the comparison between runs.

Table 9: **Graph-theoretic validity of candidate summaries** on the full ID split (n=1162, Qwen3.5-27B). "Usable" means the summary parses and passes the schema; "valid" means it is a graph-validated CPDAG.

Run	Usable	Valid (all records)	Valid | usable
Structured Thinking, constrained decoding	97.2	76.8	79.0
Structured Thinking, no hard constraint	74.5	60.4	81.1
Structured Thinking, independent run	94.9	81.3	85.7

The first two rows form the matched comparison discussed in §5. On the 945 graph-validated summaries of the independent run, the graph implies the official label in 930 cases, and Turn 2 agrees with its graph in 933.

F Probe details

Counterfactual construction. From the 1162 ID instances, 384 admit a counterfactual CPDAG with the same variable count under which the hypothesis flips; we froze a selection of 192 before running the probe. For every triplet, the reference, counterfactual, and serialization-control graphs are recomputed and checked by the validator of Appendix E, together with the truth value of the hypothesis on each graph. Turn 2 uses the standard tool-result template and decoding settings. Of the 173 flips, 96 go from NO to YES and 77 from YES to NO; every flip moves toward the counterfactual graph's truth. The premise-present variant reuses the 192 triplets from an earlier run and was analyzed post hoc.

Reference-CPDAG decomposition. The reference CPDAG is built by a symbolic solver that parses the premise, runs the PC algorithm, enumerates the Markov equivalence class, and evaluates the hypothesis. On the same frozen 128 instances, with the premise present, candidate graphs are usable for 117, 118, and 122 instances on seeds 42, 52, and 62. On these paired instances, accuracy with the candidate versus the reference graph is 96.58 versus 96.58%, 93.22 versus 97.46%, and 97.54 versus 97.54%. Only seed 62 has the full 2×2 design of Table 3b.

Destructive scrambles. Table 10 reports full-split probes that corrupt fields of the emitted graph in Turn 2. They show strong dependence on graph content, but a scrambled graph need not be a valid CPDAG, which is why the counterfactual probe above is our primary evidence.

G Cross-task transfer

Both studies use `gpt-5.6-terra` and `gpt-5.6-luna` with high reasoning effort, identical frozen item sets across arms and snapshots, and a fail-closed denominator in which any protocol failure counts as an error. Turn 1 sees the story but not the query, and Turn 2 sees the emitted intermediate and the query but not the story. *Direct* answers from the story in one call; *Prose* restates the explicit facts as text; *Structured* emits a typed relation graph through a schema-enforced tool call; *Gold* gives Turn 2 the reference graph.

Table 10: **Destructive Turn-2 probes on Qwen3.5-27B ID** (single run of Table 1).

Turn-2 condition	F_1(Yes)	Acc.	ΔF_1
Full Structured (unmodified)	86.36	95.87	—
no-premise (drop P in Turn 2)	82.56	94.84	−3.80
scrambled-skeleton only	80.11	93.98	−6.25
scrambled-v-structures only	80.00	94.06	−6.36
scrambled-directed only	80.70	94.32	−5.66
full-scramble (all of skeleton, directed, undirected)	74.32	92.69	−12.04

StepGame. We use the corrected-noise release of StepGame [Li et al., 2024] (MIT license) and select 360 test items by hash, stratified over $k \in \{4, 6, 8, 10\}$ reasoning hops and the nine relation labels. We selected StepGame after the CLUTRR result below was known and fixed its data, arms, primary comparison (Structured minus Prose), and statistical thresholds before the first full run.

CLUTRR. We use 320 frozen CLUTRR [Sinha et al., 2019] items with the same four arms. Structured falls below Prose on both snapshots, and even Gold reaches only 34.06%: most wrong Gold answers predict no relation, so composing kinship relations from the graph alone is itself a bottleneck.

Table 11: **Cross-task accuracy (%)** on two GPT-5.6 snapshots. p is the exact McNemar test for Structured versus Prose.

Task	Snapshot	Direct	Prose	Structured	Gold	p
StepGame (n=360)	terra	99.72	70.56	**83.06**	94.17	1.9×10^{-6}
	luna	99.44	19.44	**71.39**	98.06	3.3×10^{-46}
CLUTRR (n=320)	terra	69.06	**70.63**	17.81	34.06	6.8×10^{-42}
	luna	68.44	**55.00**	22.19	34.06	1.9×10^{-19}

H Prompt templates

We reproduce the main prompts for Structured Thinking and the direct PC baseline, as found in `src/prompts.py`. Each prompt is a Python string constant; the Python-level newlines below correspond exactly to the newlines passed to the model. Additional control variants, including the run reported here as TWOTURN-PROSE-PC and SINGLETURN-STRUCTPROMPT, are recorded in the released experiment metadata.

H.1 Structured Thinking system prompt (Sstruct, Turn 1)

Source: `src/prompts.py:131-162` (constant `SYSTEM_STRUCTURED_V2`).

```
You are an expert in causal inference and the PC algorithm for
causal discovery from observational data.

Given correlational premises (marginal dependencies and conditional
independencies) and a causal hypothesis, analyze the causal structure
using the PC algorithm:

## Step 1: Determine the Skeleton
- Start with a complete undirected graph over all variables.
- For each conditional independence X _||_ Y | S, REMOVE edge X-Y.
  Record S as the separating set sep(X,Y).
- CRITICAL: If two variables are adjacent in a faithful DAG, they
  CANNOT be conditionally independent given ANY subset. So any stated
  CI X _||_ Y | S proves there is NO direct edge X-Y.
- Marginal correlation does NOT imply adjacency -- variables can
  correlate through indirect paths.

## Step 2: Identify V-Structures (Colliders)
```

```
- For each non-adjacent pair (X, Y) with common neighbor Z in
  the skeleton:
  - If Z is NOT in sep(X,Y) -> orient as X -> Z <- Y (v-structure)
  - If Z IS in sep(X,Y) -> NOT a v-structure, do not orient

## Step 3: Apply Meek's Rules (iterate until no changes)
- R1: X -> Y -- Z, X and Z non-adjacent => orient Y -> Z
- R2: X -> Y -> Z, X -- Z => orient X -> Z
- R3: X -- Z, Y -> Z, W -> Z, X adj Y, X adj W, Y and W
  non-adjacent => orient X -> Z

## Step 4: Record your CPDAG using the CausalAnalysis tool

Important principles:
- Closed-world assumption: only the stated independencies hold.
  Any independence not listed is assumed to NOT hold.
- A hypothesis is deducible iff it holds in ALL DAGs of the
  Markov equivalence class (all valid completions of the CPDAG).

End your final answer with exactly "Answer: Yes" or "Answer: No".
```

Although this legacy Turn-1 system prompt contains an answer-format sentence, the constrained decoder accepts only the `CausalAnalysis` tool call in Turn 1; final answers are generated and extracted only in Turn 2.

H.2 Turn-2 tool-result template

Source: `src/prompts.py:166-187` (function `make_tool_result_v2`). After Turn 1 emits a `CausalAnalysis` object g, the tool-result message fed back in Turn 2 is formatted as:

```
Causal analysis recorded. Your CPDAG summary:
  Variables: <variables>
  Skeleton: <skeleton>
  Non-adjacent: <non_adjacent>
  V-structures: <v_structures>
  Directed: <directed>
  Undirected: <undirected>

<DOT visualization>

Now evaluate the hypothesis against this CPDAG:
1. Check if the relevant edge exists in the skeleton.
2. Check if its direction is determined in the directed edges.
3. For undetermined edges: enumerate ALL valid DAG completions
   and verify the hypothesis holds in EVERY one.
4. For indirect causation claims: check if a directed path
   exists in ALL valid completions.
5. Use systematic case analysis -- do NOT simply conclude
   'direction is unknown, therefore No'.
```

H.3 BaselineEnhanced system prompt (Sbase)

Source: `src/prompts.py:115-118` (constant `SYSTEM_BASELINE_ENHANCED`), which concatenates the `ENHANCED_REASONING` block (`src/prompts.py:12-39`) with a terminal answer-format instruction.

```
You are an expert in causal inference, graphical causal models, and
causal discovery from observational data.

When given correlational premises (marginal and conditional
independencies) and a causal hypothesis, apply the following
reasoning framework:

1. IDENTIFY the skeleton: which pairs of variables are adjacent
   (connected) based on marginal and conditional dependencies.
2. ORIENT v-structures (colliders): if X and Y are both adjacent
```

```
   to Z but not to each other, orient as X -> Z <- Y.
3. APPLY orientation rules (Meek's rules) to propagate edge
   directions where forced by acyclicity and existing orientations.
4. DETERMINE the Markov equivalence class: identify which edges
   have determined directions and which remain ambiguous.
5. EVALUATE the hypothesis: a causal claim can be DEDUCED if and
   only if it holds in ALL DAGs within the Markov equivalence class
   consistent with the observed independencies.

Important principles:
- Assume the stated statistical relations are exhaustive (the
  closed-world assumption): any independence not listed is a dependence.
- Conditional independence X _||_ Y | Z constrains the graph structure.
- Some causal directions CAN be determined from observational data
  (especially via v-structures and orientation propagation).
- A hypothesis is deducible if every faithful DAG consistent with
  the stated independencies entails that hypothesis.
- A hypothesis is NOT deducible if there exists at least one
  faithful DAG consistent with the premises where it does not hold.

End your response with exactly "Answer: Yes" or "Answer: No".
```

The baseline is a strong PC-instruction baseline rather than a plain chain-of-thought baseline: it covers skeleton construction, v-structures, Meek's rules, Markov equivalence classes, and the closed-world assumption. It is not token-identical to S_{struct}; in particular, its collider rule omits the condition that Z is not in the separating set of X and Y.

H.4 Control prompt variants

TWOTURN-PROSE-PC is a two-turn prose control with the same PC scaffold as BASELINEENHANCED. Turn 1 uses the baseline's skeleton, v-structure, Meek-rule, closed-world, and MEC-universal-quantifier instructions, but asks for free-text analysis and explicitly withholds the final yes/no answer. Turn 2 returns that free-text reasoning and asks for the final answer. This control uses no tool call and no schema-constrained typed graph object. Its prompt template is:

```
Turn 1:
<BaselineEnhanced system prompt above, with the final
answer-format instruction replaced by:>

Do not give the final yes/no answer yet. Write your causal analysis in
free text, following the PC-algorithm scaffold above.

Turn 2:
Here is your previous analysis:
<turn1_analysis>

Now answer the original causal hypothesis. End your response with
exactly "Answer: Yes" or "Answer: No".
```

SINGLETURN-STRUCTPROMPT reuses the structured CPDAG-oriented instructions in a single-turn direct-answer setting, but disables the tool call and schema-constrained graph output. These controls help distinguish graph-object externalization from a PC-scaffolded prose extra turn and graph vocabulary alone.

I Schema and constrained-decoding grammar

I.1 `CausalAnalysis` schema

Source: `src/schema.py:194-241`. The schema is defined with Pydantic and converted to JSON Schema via `CausalAnalysis.model_json_schema()`.

```
class CausalAnalysis(BaseModel):
    """Structured causal analysis following the PC algorithm.
```

```
    Determine the causal skeleton, identify v-structures, apply Meek's
    rules, and record the resulting CPDAG.
    """
    variables:    list[str]   # e.g. ['A', 'B', 'C', 'D']
    skeleton:     list[str]   # adjacent pairs, e.g. ['A-B', 'C-D']
    non_adjacent: list[str]   # non-adj pairs w/ sep set, e.g. ['A-C|B']
    v_structures: list[str]   # e.g. ['A->E<-C']
    directed:     list[str]   # oriented edges, e.g. ['A->E', 'E->F']
    undirected:   list[str]   # e.g. ['D-E', 'A-F']
```

I.2 Regex construction (Qwen / Mistral)

Source: `src/schema.py:248-286`. The regex derived from the JSON schema via `outlines_core.json_schema.build_regex_from_schema` is wrapped in a model-specific envelope.

```
# Qwen family (Qwen3.5-27B, Qwen3.6-27B, Qwen3.5-9B) -- chat template uses
# <think>...</think> for reasoning and <tool_call>...</tool_call>
# for tool invocation.
def build_analysis_regex(enable_thinking: bool = True) -> str:
    schema_str = json.dumps(CausalAnalysis.model_json_schema())
    args_regex = build_regex_from_schema(schema_str)
    body = (
        r'\{"name":\s*"CausalAnalysis",\s*"arguments":\s*'
        + args_regex + r"\}"
    )
    tool_call = r"<tool_call>\s*" + body + r"\s*</tool_call>"
    if enable_thinking:
        return r"<think>\s*[\s\S]+?</think>\s*" + tool_call
    return tool_call

# Magistral / Mistral family -- chat template uses [THINK]...[/THINK]
# and emits the raw JSON object without a tool_call envelope.
def build_analysis_regex_mistral(enable_thinking: bool = True) -> str:
    schema_str = json.dumps(CausalAnalysis.model_json_schema())
    args_regex = build_regex_from_schema(schema_str)
    if enable_thinking:
        return r"\[THINK\][\s\S]+?\[/THINK\]\s*" + args_regex
    return args_regex
```

For API models (`gpt-5.4-mini`) the same schema is enforced through OpenAI tool-calling with `strict=true`; the reported main run uses `tool_choice="auto"` plus a prompt-level instruction to call the tool, because name-forcing the tool compressed reasoning (Appendix A.1).

J Per-experiment tables

Full per-experiment results (F_1, accuracy, precision, recall, per-complexity F_1, confusion matrices) for every reported experiment in the main text and appendix are included in the released artifact as one `evaluation.json` per experiment directory (`experiments/<name>/evaluation.json`), together with audit outputs for Table 4. Token and wall-clock cost per experiment are similarly recorded in `experiments/<name>/usage.json` and summarized in `analysis/cost_table.json`. We omit the verbatim per-cell tables from the appendix to stay within the page budget.

K Assets and licenses

Corr2Cause is used as released by Jin et al. [2024]. The open-weight models are accessed through public Hugging Face releases: `Qwen/Qwen3.5-27B`, `Qwen/Qwen3.6-27B`, `Qwen/Qwen3.5-9B`, and `google/gemma-4-31B-it`; revision hashes are recorded in the run metadata. The corresponding model cards or license files report Apache-2.0 licensing. GPT-5.4-mini and the GPT-5.6 snapshots are accessed through the OpenAI API under the applicable API terms; snapshot identifiers are recorded in the run metadata. StepGame is used in its corrected-noise release under the MIT license [Li et al., 2024], and CLUTRR from its official repository [Sinha et al., 2019]. vLLM,

XGrammar, and the schema-constrained decoding dependencies are cited in the main text; exact package versions are recorded in the released metadata. The code and data artifact is available at `https://github.com/wentaopoly/Structured-Thinking-Matters`; it contains experiment outputs, prompts, schemas, and analysis scripts produced for this paper. No new model weights or human-subject data are released.

L Failure-case gallery

Table 12: **Failure-mode distribution** on 32 hand-labelled Structured Thinking errors (Qwen3.5-27B, ID split).

Failure mode	Count	%
CPDAG_WRONG — missing or spurious graph edges	13	40.6
SEMANTIC_ERROR — correct graph, wrong Turn-2 readout	8	25.0
UNDIRECTED_MEC — MEC ambiguity collapsed incorrectly	6	18.8
KG_INVALID — schema-valid but logically malformed	5	15.6

We hand-label 32 representative Structured Thinking errors selected to cover the observed failure categories; the gallery is diagnostic rather than an unbiased prevalence estimate. The records are included in the released artifact as `analysis/failure_modes.json`, each providing the instance index, variable list, ground-truth label, model prediction, and error category. Two illustrative worked examples follow.

CPDAG_WRONG (idx 321). Premise involves 5 variables. Ground-truth CPDAG has $A-E$ in the skeleton as part of a v-structure; the model emitted a skeleton that omitted $A-E$, so Turn-2 lacks the necessary edge and answered YES to "A is parent of E" via an irrelevant alternative path.

UNDIRECTED_MEC. In these errors, the model treats an undirected CPDAG edge as if one orientation were forced, or checks only one valid completion of the Markov equivalence class. For a parent claim on an undirected edge $X-Y$, the correct universal-quantifier test must reject the claim whenever both $X \to Y$ and $X \leftarrow Y$ remain valid completions. These cases illustrate that schema-valid graph summaries do not guarantee a correct all-completions readout.

M Reproducibility

Every experiment can be reproduced by a single command using the released code at `https://github.com/wentaopoly/Structured-Thinking-Matters`. Chain scripts at the repository root orchestrate batched runs: `run_qwen9b_chain.sh` and `run_qwen9b_ablation_chain.sh` cover the Qwen3.5-9B main and ablation runs; `run_probe_chain.sh` covers the full-split probes of §6; `run_additional_chain.sh` orchestrates additional diagnostic runs; `run_ood_all.sh` produces the Refactor-OOD results; the seed-matched Qwen3.5-27B re-run in Appendix B is recorded with the same per-experiment metadata format. Open-weight runs use $2\times$ GPU tensor-parallel vLLM inference with `gpu_memory_utilization`$=0.88$, and API runs use `max_concurrency`$=5$. All `metadata.json` files record the exact decoding arguments (temperature, top-p, max length, tensor-parallel size, tokenizer mode, constrained-decoding flag), Hugging Face model revision hashes, tokenizer revisions, API model snapshot strings, and package versions for each experiment. The Corr2Cause dataset is used as released at `https://huggingface.co/datasets/causal-nlp/corr2cause`; Refactor-OOD and Paraphrase-OOD are its `perturbation_by_refactorization_test.csv` and `perturbation_by_paraphrasing_test.csv` files.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and §1 state the main claims: gains over a strong PC-instruction baseline across three seeds, cross-model and perturbed-split support, ablation evidence, graph-validity and counterfactual probes, and a reference-CPDAG decomposition; limitations are scoped in §8.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: §8 discusses the CPDAG-deduction scope and the observed cross-task boundary (CLUTRR), the three-seed basis and seed-dependent size of the principal Qwen3.5-27B gain, the remaining single-run perturbed-split, larger-graph, cross-model, and control rows, the share of test instances that do not yield a graph-validated CPDAG and the validator's scope, and bundled method components; backend/tool-choice sensitivity and inference cost are discussed in §7.4.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper is empirical; it makes no new theoretical claims. The PC-algorithm and Markov-equivalence assumptions inherited from Corr2Cause [Jin et al., 2024] (faithfulness, causal Markov) are stated in §3.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: §4.1 reports backend, decoding parameters, exact model IDs, datasets, and significance-test protocol; Appendix H gives the main Structured Thinking and direct PC-baseline prompts verbatim and describes the PC-scaffolded prose and single-turn control variants; Appendix I gives the Pydantic schema and regex construction; Appendix M gives the repository URL, reproduction commands, and released artifacts, with model revisions, API snapshots, larger-graph stress-test metadata, and all control prompts recorded in experiment metadata.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code and data are available at `https://github.com/wentaopoly/Structured-Thinking-Matters`, including prompts, schema, the full `CausalAnalysis` specification, chain scripts (`run_qwen9b_chain.sh`, `run_probe_chain.sh`, `run_additional_chain.sh`), evaluation/usage/metadata JSON artefacts, and the 32-case failure set; the Corr2Cause dataset is public [Jin et al., 2024].

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: No training; all results are inference-only. Decoding hyperparameters (T=0.6, top-p=0.95, max length 32768), backends (vLLM for open-weight, OpenAI Responses API for `gpt-5.4-mini`), constrained-decoding implementations, split sizes, model revision hashes, and API snapshots are reported in §4.1 and in the per-experiment `metadata.json` files.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.

- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The principal Qwen3.5-27B ID comparison is a seed-matched rerun on three seeds: §4 and Appendix B report per-seed exact McNemar tests on the paired n=1162 items and the mean $\pm$ sample standard deviation across seeds, which captures run-to-run variation from stochastic decoding. Probes and cross-task studies report exact McNemar tests on paired items, and the API tool-choice control a paired-bootstrap 95% CI. §8 states that the perturbed splits, larger-graph stress tests, cross-model rows, and controls are single runs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Cost is summarized in §7.4 and derived from the committed `usage.json` files; open-weight runs use 2$\times$ GPU tensor-parallel vLLM inference at `gpu_memory_utilization`= 0.88; API runs use the OpenAI Responses API with `max_concurrency`= 5. The seed-matched rerun, probes, and reference-CPDAG studies ran on NVIDIA H100, A100, and RTX 6000 Ada GPUs, and the cross-task studies used the OpenAI API.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research uses public benchmarks and released model weights or APIs, involves no human subjects, and makes no privacy-sensitive data accesses.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The method is an evaluation/prompting technique for causal-deduction benchmarks. Potential positive impact is better auditing of LLM causal reasoning; potential negative impact is over-trust if structured outputs are mistaken for certified causal discovery. §8 reports how often test instances do not yield a valid CPDAG, and the method should not be used as a substitute for a causal solver.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No new models or data are released; the released artefact is inference-time code using public models and the public Corr2Cause dataset.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Corr2Cause [Jin et al., 2024], StepGame [Li et al., 2024] (MIT), and CLUTRR [Sinha et al., 2019] are used under their released licenses; Qwen3.5, Qwen3.6, Gemma-4, GPT-5.4-mini, and GPT-5.6 are accessed under their respective model licenses / API terms; Outlines [Willard and Louf, 2023], XGrammar [Dong et al., 2024], and vLLM [Kwon et al., 2023] are used under their open-source licenses. Appendix K summarizes assets and licensing.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The released repository contains the `CausalAnalysis` schema, all prompts, the two-turn pipeline implementation (`pipeline.py`, `pipeline_api.py`), evaluation scripts, and significance analysis (`analysis/significance.py`); a README documents usage.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The research involves no crowdsourcing or human subjects; the 32-case failure taxonomy was hand-labelled by the authors.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The research involves no human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are the core subject of study: the method prompts open-weight models (Qwen3.5-9B/27B, Qwen3.6-27B, Gemma-4-31B) and API models (GPT-5.4-mini, GPT-5.6) to emit a structured CPDAG-style summary and answer a causal hypothesis against it. The main prompt templates, schemas, and decoding constraints are reported in §3 and Appendices H–I; control prompt variants are described in Appendix H.4 and recorded in the released experiment metadata.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.

- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.