Mapping the Minds of LLMs: A Graph-Based Analysis of Reasoning LLMs

Anonymous ACL submission

Abstract

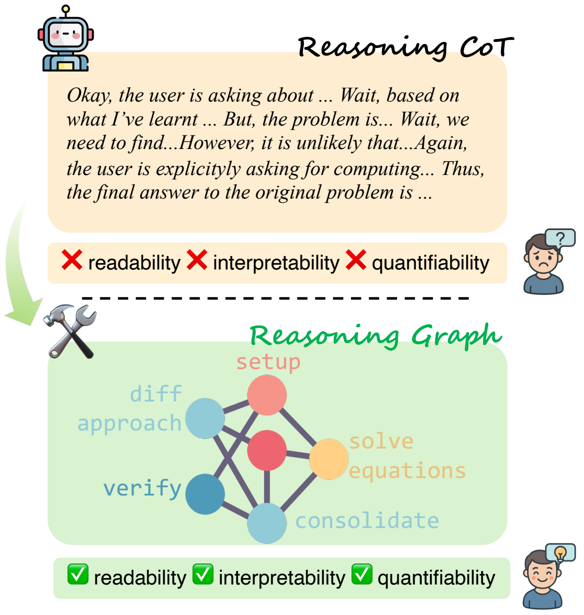
Recent advances in test-time scaling have enabled Large Language Models (LLMs) to display sophisticated reasoning abilities via extended Chain-of-Thought (CoT) generation. Despite their impressive reasoning abilities, Reasoning LLMs (RLMs) frequently display unstable behaviors, e.g., hallucinating unsupported premises, overthinking simple tasks, and displaying higher sensitivity to prompt variations. This raises a deeper research question: *How can we represent the reasoning process of RLMs to map their minds?* To address this, we propose a unified graph-based analytical framework for fine-grained modeling and quantitative analysis of RLM reasoning dynamics. Our method first clusters long, verbose CoT outputs into semantically coherent reasoning steps, then constructs directed reasoning graphs to capture contextual and logical dependencies among these steps. Through a comprehensive analysis of derived reasoning graphs, we also reveal that key structural properties, such as exploration density, branching, and convergence ratios, strongly correlate with models' performance. The proposed framework enables quantitative evaluation of internal reasoning structure and quality beyond conventional metrics and also provides practical insights for prompt engineering and cognitive analysis of LLMs. Code and resources will be released to facilitate future research in this direction.

1 Introduction

Recent LLMs equipped with test-time scaling capabilities, such as OpenAI's o-series (OpenAI et al., 2024; OpenAI, 2025), DeepSeek-R1 (DeepSeek-AI et al., 2025), and Gemini-2.5 (Kavukcuoglu, 2025), employ a system II, *think-slow-before-answer*, pipeline that transforms how these models approach complex problems during test time. Rather than producing outputs directly after the input with normally limited token length, these reasoning models engage in explicit and free extended reasoning through Chain-of-Thought (Wei et al., 2022) mechanisms. This innovation enables reasoning models to decompose intricate challenges in various domains, explore multiple possible solutions, and self-assess intermediate conclusions before synthesizing final responses during their extended inference time. In general, these reasoning models currently outperform conventional LLMs on various types of benchmarks, which require advanced math (Patel et al., 2024) and coding (Jimenez et al., 2024) capability.

Despite these promising advancements, reasoning models exhibit undesire (Chen et al., 2024) and unstable (Yang et al., 2025b) behaviors that challenge the established understanding of large language models. One of the particularly striking phenomena is the performance degradation associated with few-shot learning, which in most cases improves the performance of conventional LLMs. Recent technique reports also documented that these RLMs are somehow more sensitive to prompts (DeepSeek-AI et al., 2025). We believe these existing unclear behaviors of RLM call for deeper investigations into how RLMs operate and reason.

Our research proposes a novel framework to trace the reasoning processes from a graph perspective. While some work has previously examined the correlation between the quantity of reasoning tokens and RLM's accuracy (Sui et al., 2025; Ballon et al., 2025; Yang et al., 2025b), our approach goes beyond the token perspective and focuses on the semantic organization of the model's reasoning processes. Specifically, our analytical frameworks first cluster raw and verbose reasoning tokens into coherent logic steps and then map their inter-dependencies as a graph, enabling globally semantical insights into how reasoning models reason at a higher level (Figure 1). After a comprehensive analysis of derived reasoning graphs, we identify specific quantifiable features that are associated


Reasoning CoT
✗ readability ✗ interpretability ✗ quantifiability
Reasoning Graph
setup
diff approach
solve equations
verify
consolidate
✓ readability ✓ interpretability ✓ quantifiability

Okay, the user is asking about ... Wait, based on what I've learnt ... But, the problem is... Wait, we need to find...However, it is unlikely that...Again, the user is explicityly asking for computing... Thus, the final answer to the original problem is ...

Figure 1: A conceptual overview of our framework for modeling the long reasoning CoT with a graph structure. This graph-based representation enables stronger readability for human researchers, systematic interpretability of the global structure, and quantifiable graph metrics for in-depth analysis.

with advanced reasoning behavior, which is often linked to higher problem-solving performance.

To summarize, our contributions in this paper include:

- a novel reasoning-graph toolkit that converts natural language long Chain-of-Thought into analyzable graph structures, enabling quantification of reasoning through topological and semantic metrics.
- comprehensive analysis of how different prompting strategies may influence reasoning LLMs, establishing quantitative boundaries for prompt engineering optimization.
- quantifiable indicators of reasoning quality beyond task accuracy, providing a higher-level cognitive understanding of reasoning in LLMs.

2 Related Works

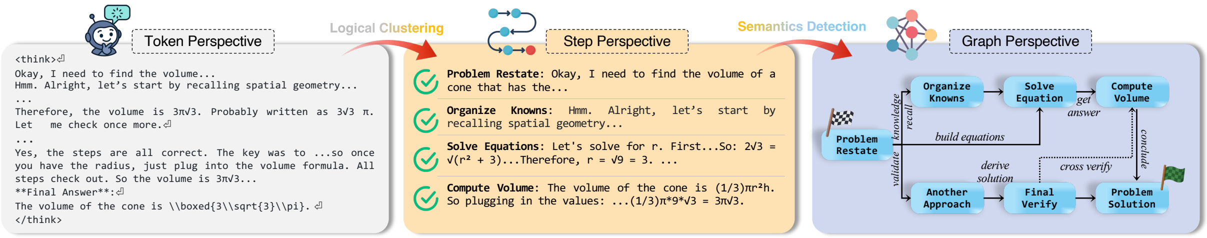
Test-Time Scaling Similar to human dual-processing hypothesis of the mind (Da Silva, 2023), augmenting the computational budget at test-time has been shown to substantially enhance the reasoning capabilities of large language models (LLMs) (OpenAI et al., 2024; OpenAI, 2025; Kavukcuoglu, 2025; Anthropic, 2025). These reasoning LLMs (RLMs) show highly advanced self-reflection, backtracking, and cross-validation behavior during the extended chain-of-thought (CoT) responses, enabling them to tackle intricate reasoning challenges and outperform previous conventional base LLMs (Li et al., 2025; Chen et al., 2025).

Few-Shot Learning Few-shot prompting once emerged as a crucial technique for enhancing the performance and adaptability of large language models (LLMs) by providing limited yet highly informative demonstrations (Song et al., 2023). In detail, it leverages a minimal number of illustrative examples embedded directly into the input context, enabling models to rapidly generalize across diverse tasks without explicit parameter updates (Brown et al., 2020). However, many researchers and practitioners have reported that few-shot prompting could instead degrade the model's performance (DeepSeek-AI et al., 2025), signaling the instability of current reasoning LLMs. In this paper, we will examine the impact of zero/few-shot prompting on RLM's reasoning, assessing both the quality of internal reasoning and overall performance in in-context learning scenarios. Provide more valuable insights for future prompt engineering and model optimization.

Long CoT Analysis Some previous studies have identified a negative relationship between an RLM's accuracy and the number of reasoning tokens it generates (Ballon et al., 2025; Yang et al., 2025b). However, their analyses of RLMs mainly relied on a one-dimensional metric: the length of CoT token sequences. It still remains unclear and counterintuitive why even longer system II thinking could lead to performance degradation, suggesting a gap in our understanding of how RLMs work in general. In this work, we introduce a comprehensive structured framework to formulate the chain-of-thought process, offering deeper insights into the underlying reasoning behavior.

3 Constructing Reasoning Graph from Raw Reasoning Tokens

Given more computational resources at inference time, Reasoning Language Models (RLMs) can autonomously explore feasible solutions, perform cross-validations, actively access intermediate steps, and synthesize consolidations through



```
<think>⏎
Okay, I need to find the volume...
Hmm. Alright, let’s start by recalling spatial geometry...
...
Therefore, the volume is 3π√3. Probably written as 3√3 π.
Let   me check once more.⏎
...
Yes, the steps are all correct. The key was to ...so once
you have the radius, just plug into the volume formula. All
steps check out. So the volume is 3π√3...
**Final Answer**:⏎
The volume of the cone is \\boxed{3\\sqrt{3}\\pi}. ⏎
</think>
```

```
Problem Restate: Okay, I need to find the volume of a cone that has the...
Organize Knowns: Hmm. Alright, let’s start by recalling spatial geometry...
Solve Equations: Let's solve for r. First...So: 2√3 = √(r² + 3)...Therefore, r = √9 = 3. ...
Compute Volume: The volume of the cone is (1/3)πr²h. So plugging in the values: ...(1/3)π*9*√3 = 3π√3.
```

Figure 2: Our pipeline for building the graph structure from reasoning large language models’ output. Starting from raw **token perspective**, we first use "\n\n" as natural delimiters to split the raw reasoning tokens into an ordered list of reasoning units. Then we perform logical clustering to combine logically cohesive reasoning units into a reasoning step (*node*), shifting into intermediate **step perspective**. Lastly, we detect semantical relationships (*edge*) between steps (*node*) to reveal the high-level **graph perspective** from reasoning LLM’s output.

extended chain-of-thought tokens. This critical feature allows RLMs to fully release their internal reasoning potential under sophisticated challenges. However, current test-time scaling is also a double-edged sword: as models are encouraged to elaborate and reflect, their behavior often becomes unreliable and less predictable.

Counterintuitively, the “thinking out loud” style of RLM should, in theory, offer richer data for LLM interpretability research and help us understand how LLM actually reasons. Yet, to the best of our knowledge, there is still a lack of effective methods to systematically analyze and model the semantic content of RLM-generated reasoning tokens. To fill this gap, we propose a novel, structured approach for representing and dissecting the reasoning process of RLMs.

It is widely acknowledged that RLMs tend to generate complex, branching chains of thought. This pattern closely mirrors the way humans reason: rather than following a strictly linear path, our thinking often leaps between ideas, drawing on contextual cues, connecting prior knowledge and memories, searching for potential solutions, and constantly checking for errors along the way. It is precisely this interplay of multiple analytical paths that allows us to synthesize a coherent conclusion. Inspired by this convoluted human *mind map*, we propose a unified, graph-based framework (Figure 2) to model the structure of RLM outputs.

3.1 Graph Formalization

We can formally define the reasoning graph $G = (V, A)$ with the following components:

- $V = \{s_1, s_2, ..., s_n\}$: An ordered list of vertices representing semantically clustered reasoning steps.
- $A \in \{-1, 0, 1\}^{n \times n}$: An adjacency matrix representing the ternary logical relationship between reasoning steps.

In the remaining part of this section, we will first introduce a method for clustering long and verbose reasoning traces into discrete, semantically coherent reasoning steps, each of which will serve as a **node** in a reasoning graph (Section 3.2). We then describe how to extract semantic dependencies between these steps to form the **edges** of the reasoning graph ((Section 3.3)). This reasoning graph construction method will be used in subsequent sections as a key tool for quantitatively analyzing RLM’s behavior.

3.2 Clustering Raw Tokens into Discrete Reasoning Steps

Long chain-of-thought (CoT) sequences generated by RLMs often span thousands of tokens. While these detailed traces offer rich insight into the model’s reasoning process, their length and fragmented nature present challenges for systematic analysis. A common pre-processing strategy is to segment the output based on explicit delimiters: RLMs frequently insert the token “\n\n” to denote boundaries between successive thoughts. Let $T = (t_1, t_2, \dots, t_N)$ represent the generated token sequence and use $\mathcal{D} =$ "\n\n" as the delimiter. We thus obtain an initial partition into *reasoning units*:

$$U = (u_1, u_2, \dots, u_M),$$

where each u_i is a contiguous subsequence bounded by delimiters, i.e., $u_i = (t_{s_i}, \dots, t_{e_i})$ and $t_{e_i+1} = \mathcal{D}$.

Despite its simplicity, delimiter-based segmentation has two fundamental limitations. For complex tasks such as advanced mathematical reasoning or

$\mathcal{P}_{\text{clu}}$ Context-aware Logical Units Clustering

Instruction:
You are given a sequence of reasoning units...
[*Logical Units Template*]

Your task is to cluster consecutive reasoning units that are semantically connected...

Expected Output Format: [*Output Guideline*]

Output: {"s_0": {"title":..., "content":...},...}

Figure 3: Our abbreviated prompt template to guide LLM to cluster reasoning units into logical cohesive reasoning steps. For detailed *Logical Units Template* & *Output Guideline* see Appendix A.

code generation, M can be excessively large, resulting in an unwieldy number of fragmented units that hinder semantic analysis and dependency extraction. Moreover, the model's stylistic tendency to insert delimiters frequently can lead to reasoning units that are too fine-grained, often lacking coherent context for standalone analysis.

Context-aware Logical Units Clustering. To address these challenges, we introduce a context-aware logical units clustering procedure that aggregates semantically related reasoning units into higher-level *reasoning steps*. Specifically, we leverage a large language model (LLM) to sample possible clusterings under decoding temperatures τ_r conditioned on a carefully designed prompt template $\mathcal{P}_{\text{clu}}$ (see Figure 3) $\mathcal{P}_{\text{clu}}$:

$$S = (s_1, s_2, \dots, s_K) \sim P_{\text{LLM}}(S \mid \mathcal{P}_{\text{clu}}, U; \tau_r),$$

where each s_j is ideally formed by concatenating adjacent u_i meeting a semantic affinity criterion, with $K \ll M$. This aggregation aims to ensure that each reasoning step s_j provides sufficient context for downstream analysis while maintaining a manageable total number of segments.

Yet, given the generative nature of LLMs, repeated invocations of the clustering prompt do not guarantee an identical clustering. Rather than treating this variability as noise, we further harness it through a further ensemble sampling and selection approach to identify the most coherent clustering.

Ensemble Sampling To capture the full range of possible clustering, we generate an ensemble of B candidate instances of clustering by independently sampling $P_{\text{LLM}}(S \mid \mathcal{P}_{\text{clu}}, U)$ from with varied random seeds and decoding temperatures:

$$\mathcal{C} = \{C^{(1)}, C^{(2)}, \dots, C^{(B)}\}$$
$$C^{(b)} = (s_1^{(b)}, \dots, s_{K_b}^{(b)})$$

Each $C^{(b)}$ is a candidate clustering of the original reasoning units.

To objectively compare these candidates, we introduce a weighted average quality score that evaluates key properties of each clustering. In general, we compute

$$F(C^{(b)}) = \sum_{\ell=1}^{L} w_\ell \, \phi_\ell(C^{(b)}),$$

where each ϕ_ℓ is a quantitative metric and the weights w_ℓ are non-negative and sum to one. In our experiments, we consider three intuitive criteria:

Criteria 1 **Intra-step coherence.**

$$\phi_{\text{ic}}(C^{(b)}) = \frac{1}{K_b} \sum_j \frac{2 \sum_{u<v \in s_j^{(b)}} \cos(\mathbf{e}_u, \mathbf{e}_v)}{|s_j^{(b)}|(|s_j^{(b)}| - 1)}$$

which rewards semantic similarity among units within each reasoning step.

Criteria 2 **Step-to-step separation.**

$$\phi_{\text{sep}}(C^{(b)}) = \frac{1}{K_b - 1} \sum_{j=1}^{K_b - 1} \left[1 - \cos(\mathbf{e}_{s_j^{(b)}}, \mathbf{e}_{s_{j+1}^{(b)}}) \right]$$

which encourages semantic distinctiveness between adjacent reasoning steps.

Criteria 3 **Length regularity.**

$$\phi_{\text{len}}(C^{(b)}) = 1 - \left| \frac{\frac{1}{K_b} \sum_j |s_j^{(b)}|}{\mu_{\text{ref}}} - 1 \right|$$

where μ_{ref} is a referenced average step length derived from the original CoT structure (see Appendix E for details), penalizing pathological segmentations that are too short or too long.

Here, $\mathbf{e}_{\cdot}$ denotes a sentence embedding vector derived from a pretrained encoder.

The final segmentation is selected as the candidate with the highest composite score:

$$V := C^* = \arg\max_{C \in \mathcal{C}} F(C).$$

$\mathcal{P}_{\text{sem}}$ Adjacency Matrix Sampling

Instruction:
Given an ordered sequence of K reasoning steps,...
[*Logical Steps Template*]

...Your task is to decide whether step i supports, contradicts, or is independent of step j...

Expected Output Format: [*Output Guideline*]

- -

Output: {(s_0, s_1):..., (s_0, s_2): ..., (s_1, s_0): ...}

Figure 4: Our abbreviated prompt template to detect semantical relationship between two different reasoning steps. For detailed input/output template and intuition behind the instruction, see Appendix A

This ensemble–scoring framework offers a flexible yet principled way to select coherent and analyzable reasoning step from all plausible instances of clustering generated by the LLM. The complete clustering and selection algorithm is summarized in Appendix D.

For all subsequent analyses, each s_j in the selected C^* is treated as a node in our reasoning graph, providing a compact yet semantically rich foundation for structural and dependency analysis.

3.3 Extracting Inter-Dependencies between Reasoning Steps

our next objective is to construct a directed semantic graph $G = (V, E)$, where each edge $(i, j) \in E$ represents an inferred relationship—such as support or contradiction—between step s_i and step s_j. To achieve this, we propose a rejection sampling-based semantic detection procedure that fuses global predictions from multiple LLM samplings.

Adjacency Matrix Sampling. We first obtain diverse global views of step-wise dependencies by repeatedly prompting the LLM with a structured template $\mathcal{P}_{\text{sem}}$ (see Figure 3). Each prompt presents the entire ordered set of reasoning steps and requests predictions for every ordered pair (i, j), $i < j$. The model outputs a full adjacency matrix:

$$A^{(r)} \sim P_{\text{LLM}}\left(A \mid \mathcal{P}_{\text{sem}}, V; \tau_r\right)$$
$$A^{(r)} \in \{-1, 0, 1\}^{K\times K},$$

where $A^{(r)}_{ij} = 1$ (support), -1 (contradict), or 0 (independent). We repeat this sampling R times, varying the decoding temperature τ_r to enhance diversity. This strategy ensures the LLM can leverage the full context for globally consistent predictions while capturing uncertainty.

Adaptive Edge-wise Probability Estimation. For each possible edge (i, j), we aggregate predictions across the R samples to estimate the empirical probability of each relation:

$$\hat{p}_{ij}(c) = \frac{1}{R}\sum_{r=1}^{R} \mathbf{1}\left[A^{(r)}_{ij} = c\right], \qquad c \in \{-1, 0, 1\}.$$

These aggregated probabilities provide a measure of confidence for the existence and type of each possible semantic relation.

Rather than relying on a fixed number R of sampled adjacencies, we fuse information across all samples for a robust final graph construction. Sampling continues until the estimated probabilities for all edges reach a specified confidence level. Specifically, for each edge, we compute the pooled standard error:

$$\mathrm{SE}_{ij} = \sqrt{\frac{1}{R}\sum_{l\in\{-1,+1\}} \hat{p}_{ij}(l)\big(1 - \hat{p}_{ij}(l)\big)}$$

Here, we explicitly omitted ($l = 0$) case to simplify the estimator while preserving the accuracy required for the adaptive stopping criterion (see Appendix F). The process halts once $\max_{i<j} \mathrm{SE}_{ij} \le \varepsilon$ [1], or a hard cap $R_{\max}$ is reached. This guarantees that our edge probability estimations are statistically reliable before moving to the next phase.

With reliable probability estimates, we next construct the final adjacency matrix via a consensus rule. For each pair (i, j), we define the signed confidence:

$$w_{ij} := \hat{p}_{ij}(+1) - \hat{p}_{ij}(-1), \quad w_{ij} \in [-1, 1].$$

We then apply a dual-threshold criterion:

$$A_{ij} = \begin{cases} +1, & \text{if } w_{ij} \ge \tau_{\text{pos}} \\ -1, & \text{if } w_{ij} \le -\tau_{\text{neg}} \\ 0, & \text{otherwise,} \end{cases} \qquad A_{ji} = -A_{ij},$$

where τ_{pos} and τ_{neg} can be tuned independently[2].

The resulting weighted adjacency $W = \left[w_{ij}\right]_{i,j}$ is preserved for future analysis, while we pay attention to the hard-thresholded $A = \mathrm{sign}(W) \odot \mathbf{1}[|W| \ge \tau]$, which serves as the binary backbone for structural analysis. We also include a complete

[1] in practice $\varepsilon = 0.05$ suffices

[2] We typically set $\tau_{\text{pos}} = 0.4$, $\tau_{\text{neg}} = 0.3$ to reflect the empirical imbalance between supporting and contradicting links.

adjacency matrix sampling and adaptive edge estimation algorithm in Appendix D.

To sum up, the pipeline described in this section provides a principled and systematic framework for converting raw reasoning traces from RLMs into interpretable graph structures. This unified graph representation serves as a powerful analytical tool, enabling fine-grained examination of how RLMs organize, connect, and validate intermediate inferences. In the following sections, we will leverage this reasoning graph formalism to quantitatively analyze the internal reasoning dynamics and decision-making behaviors of advanced RLMs.

4 Reveal Cognitive Behavior of RLM with Reasoning Graph

Existing analysis of Reasoning LLMs (RLMs) have primarily relied on performance-based metrics such as accuracy or token-level statistics like reasoning length. While these measures offer a coarse understanding of model behavior, they fail to capture the complex and dynamic structure exposed by RLM's output.

In this section, we propose to move beyond token-level perspectives and instead leverage the reasoning graph constructed in Section 3 as an effective medium for cognitive analysis. By representing the model's chain-of-thought as a graph of semantically coherent reasoning steps (nodes) and their directed relationships (edges), we can systematically quantify the structure, flexibility, and effectiveness of model reasoning. This shift enables us to answer deeper questions about how RLMs organize, explore, and consolidate information during problem problem-solving process. Figure 2 provides a concrete example of RLM's reasoning process for solving a spatial geometric problem.

All implementation details are included in Appendix C.

4.1 Graph-Based Metrics for Quantifying Model Reasoning

To systematically analyze the cognitive organization of RLM reasoning, we introduce several graph-based metrics, each designed to capture a distinct structural aspect of the reasoning process.

Exploration Density (ρ_E): Measures the overall connectivity among reasoning steps, reflecting the breadth of the model's exploration.

$$\rho_E(G) = \frac{|E|}{|V|(|V|-1)}$$

Higher values indicate denser intra-reasoning-step correlations.

Branching Ratio (γ_B): Quantifies the diversity of alternative reasoning paths, capturing the model's capacity for exploring parallel ideas and diverse solutions.

$$\gamma_B(G) = \frac{|\{s \in V \mid \mathrm{d}^-(s) > 1\}|}{|V|}$$

where $d^-(s)$ is the out-degree of node s.

Convergence Ratio (γ_C) : Captures the extent to which the model integrates multiple reasoning threads into unified conclusions, indicating its ability to synthesize disparate ideas.

$$\gamma_C(G) = \frac{|\{s \in V \mid \mathrm{d}^+(s) > 1\}|}{|V|} \tag{1}$$

where $d^+(s)$ is the in-degree of node s.

Linearity (ℓ): Represents the prevalence of strictly sequential reasoning, measuring the fraction of nodes with degree greater than one.

$$\ell(G) = 1 - \frac{|\{s \in V \mid d(s) > 2\}|}{|V|} \tag{2}$$

where $d(s)$ is the total degree of node s.

Collectively, these metrics offer a comprehensive high-level view of the reasoning graph structure. They allow us to quantify not only how much the model reasons, but how it organizes its thinking through branching exploration, convergence, or rigid linearity. In the following analyses, we will show how these quantities are directly related to and influence model performance.

4.2 Impact of Prompting Paradigms on Reasoning Structure

Having established our graph-based metrics, we next investigate how different prompting styles shape the internal reasoning structure of RLMs. We focus on three few-shot demonstration styles: `Minimal`, `Concise`, and `Explanatory`. (see Appendix B for detailed definitions). Each style provides a distinct richness of context, ranging from bare problem–answer pairs to human-like concise reasoning and extended, self-reflective chains generated by the model itself.

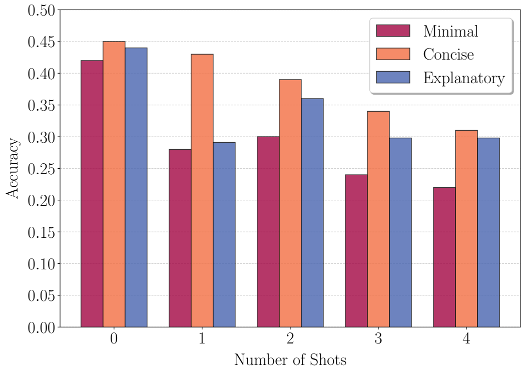


Figure 5: Few-shot prompting accuracy on GPQA-Diamond* dataset using reasoning Qwen-7B (distilled from DeepSeek-R1). The accuracy drops dramatically with respect to the increasing number of examples within the prompt.

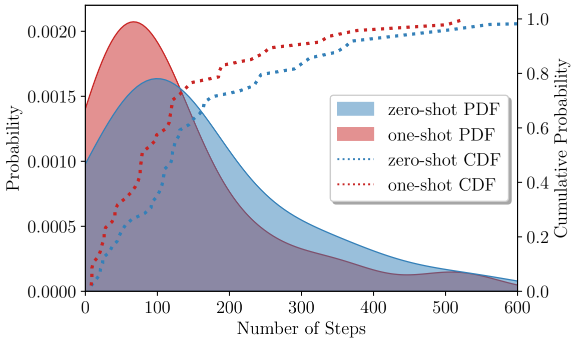
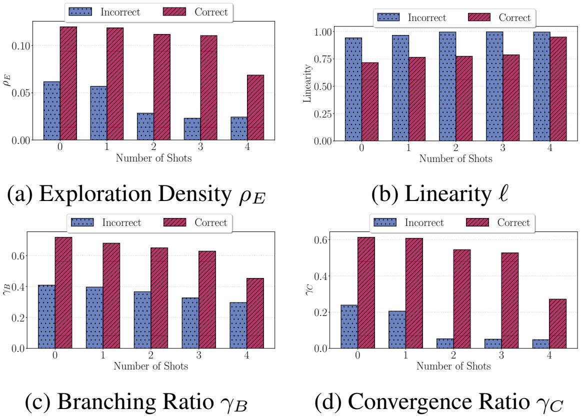


Figure 7: Distribution of reasoning step counts under zero-shot and one-shot prompting. The inclusion of a single demonstration in one-shot settings causes a pronounced shift in the distribution compared to zero-shot, highlighting the sensitivity of RLM to prompt design.



(a) Exploration Density ρ_E

(b) Linearity ℓ

(c) Branching Ratio γ_B

(d) Convergence Ratio γ_C

Figure 8: Different metrics of reasoning graph given different numbers of few-shot examples.

Prompting Style Modulates RLM Performance. Our results reveal a consistent and striking trend: increasing the number of in-context examples leads to a monotonic decline in accuracy, regardless of demonstration style (Figure 5). However, the severity of this decline is strongly related to the structure and verbosity of the provided exemplars, with `Minial` being the most RLM-unfriendly prompting style.

While there is a hypothesis explaining that few-shot prompting leads to a reduction in total length of I/O tokens (Figure 6), raw length alone does not fully explain the loss of reasoning effectiveness. Instead, these phenomena call for deeper understanding and explanations for the observed degradation in model performance.

Structural Shifts Triggered by Prompting Styles To better understand these performance variations, we examine the corresponding changes in reasoning graph topology across prompting conditions (Figure 8). It turns out that zero-shot prompting induces richer, more complex graph structures: graphs feature higher exploration density, greater branching and convergence, and a more diverse distribution of reasoning step counts (Figure 7). This suggests that, when not being influenced by extra demonstrations, the model engages in more adaptive and active exploration, revisiting, branching, and synthesizing reasoning steps.

In contrast, increasing the number of few-shot exemplars, especially in more verbose forms, systematically reduces both branching and convergence ratios, resulting in more linear graph architectures. The model appears to mimic the structure of the provided examples, limiting its capacity for active online reasoning. Notably, even a single demonstration can trigger a significant distributional shift toward shorter, more stereotyped

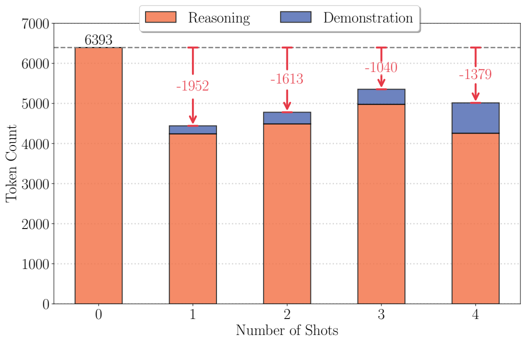


Figure 6: Average number of tokens under different numbers of shots with `explanatary` few-shot style. Few-shot prompting leads to significantly fewer reasoning tokens compared with zero-shot prompting.

Prompt Type	Acc (%)	ρ_E	γ_B	γ_C	ℓ	Mean Steps
Llama-8B*						
Zero-shot	44.5	0.117	0.564	0.676	0.744	11.6
Concise	41.2	0.065	0.252	0.426	0.947	8.6
Explanatory	32.7	0.057	0.238	0.392	0.926	10.8
Qwen-14B*						
Zero-shot	51.8	0.122	0.612	0.719	0.716	15.8
Concise	48.5	0.069	0.264	0.453	0.931	12.2
Explanatory	45.3	0.061	0.243	0.420	0.946	14.2
Qwen3-32B						
Zero-shot	56.2	0.188	0.835	0.760	0.444	19.0
Concise	53.7	0.110	0.529	0.634	0.766	15.6
Explanatory	50.1	0.069	0.283	0.449	0.931	18.0

Table 1: Comparison of model performance and reasoning graph metrics across different model sizes and prompting paradigms. * denotes reasoning models that are officially distilled from DeepSeek-R1.

reasoning chains.

These findings underscore the critical role of few-shot prompting styles in shaping RLM reasoning. Including no or a few extra demonstrations encourages flexible exploration and integration, which fosters more RLM's innate sophisticated reasoning graphs and leads to higher task performance. This motivates a shift in prompt engineering: effective demonstrations should balance informativeness with structural diversity, avoiding excessive *unnecessary* context that suppresses the model's reasoning potential.

4.3 Structural Signatures of Effective Reasoning

To comprehensively reveal the relationship between reasoning structure and task performance, we present in Table 1 a systematic comparison of graph-based metrics across multiple model scales and prompting paradigms. Several key patterns emerge that robustly distinguish effective reasoning.

Sophisticated Reasoning Graph Structure Drive Success. Across all models and prompt types, higher accuracy is consistently associated with richer reasoning graph structure: increased exploration density (ρ_E), higher branching ratio (γ_B), and greater convergence ratio (γ_C). Notably, larger models (e.g., Qwen3-32B) exhibit both the highest accuracy and the most complex graph structures, particularly under zero-shot prompting. This indicates that effective reasoning is achieved through a harder exploration (multiple attempts generation) and integrative convergence (synthesizing reasoning threads).

Prompt Constraints Induce Linearity and Impair Performance. Prompt types that impose stronger structural constraints consistently yield lower branching and convergence ratios, along with increased linearity (ℓ). This shift toward more linear graph topologies is directly correlated with performance degradation. The effect is most pronounced in smaller models but persists even for larger architectures. These results again highlight the double-edged nature of few-shot demonstrations for RLM.

Quantitative Correlations. We extend the Pearson correlation analysis to all four graph-based metrics and observe that exploration density ($r = 0.68$), branching ratio ($r = 0.67$), and convergence ratio ($r = 0.68$) each exhibit a strong positive association with accuracy with all significant at the 0.05 level. These results indicate that denser, more exploratory and convergent reasoning paths are closely linked to model accuracy. Notably, these trends persist across all model scales and prompting regimes, highlighting the robustness and explanatory value of our structural framework.

In summary, these results provide compelling evidence demonstrating that reasoning graph analysis provides highly correlated and deep insights into the internal cognitive dynamics of reasoning language models.

5 Conclusion

This paper introduces a novel graph-based framework for analyzing reasoning output produced by reasoning large language models, offering quantifiable findings about how models *organize* their thought flow under various factors. We first propose a reasoning graph toolkit that efficiently converts raw Chain-of-Thought tokens into analyzable graph structures. We then offer discovery regarding how various prompting styles may cast significant influence on RLM's internal reasoning structure and thus final performance. We also provide strong evidence supporting that graph-level predictors strongly correlate with RLM problem-solving performance. These findings not only establish quantitative insights for future prompt engineering for reasoning models, but also provide a new structural perspective for evaluating reasoning quality beyond traditional metrics and interpreting how LLMs reason at a higher level.

6 Limitations & Future Work

While our current graph-based analysis of RLMs focuses primarily on mathematical and coding tasks, extending this framework to broader domains, such as multi-modal or open-domain reasoning, may yield deeper insights into model behavior across varied scenarios and further inform our understanding of test-time scaling. In addition, the quantifiable structural metrics we propose provide a foundation for future research to explore more localized patterns and relational dynamics within reasoning graphs. We believe that ongoing work along these lines can contribute to a more comprehensive understanding and interpretability of large language models in practice.

References

Anthropic. 2025. Claude's extended thinking. https://www.anthropic.com/news/visible-extended-thinking. Accessed: 2025-05-18.

Marthe Ballon, Andres Algaba, and Vincent Ginis. 2025. The relationship between reasoning and performance in large language models – o3 (mini) thinks harder, not longer. *ArXiv*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. https://arxiv.org/abs/2503.09567. ArXiv preprint arXiv:2503.09567.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2024. Do not think that much for 2+3=? on the overthinking of o1-like llms. https://arxiv.org/abs/2412.21187. ArXiv preprint arXiv:2412.21187.

Sergio Da Silva. 2023. System 1 vs. system 2 thinking. *Psych*, 5(4):1057–1076.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z.F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2024. Deepseek-v3 technical report. https://arxiv.org/abs/2412.19437. ArXiv preprint arXiv:2412.19437.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. Swe-bench: Can language models resolve real-world github issues? In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*.

Koray Kavukcuoglu. 2025. Gemini 2.5: Our most intelligent ai model. https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/#gemini-2-5-pro. Accessed: 2025-05-18.

Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhiwei Li, Bao-Long Bi, Ling-Rui Mei, Junfeng Fang, Zhijiang Guo, Le Song, and Cheng-Lin Liu. 2025. From system 1 to system 2: A survey of reasoning large language models. https://arxiv.org/abs/2502.17419. ArXiv preprint arXiv:2502.17419.

OpenAI. 2025. Openai o3 and o4-mini system card. https://openai.com/index/o3-o4-mini-system-card/. Accessed: 2025-05-18.

OpenAI, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Bhrij Patel, Souradip Chakraborty, Wesley A. Suttle, Mengdi Wang, Amrit Singh Bedi, and Dinesh Manocha. 2024. Aime: Ai system optimization via multiple llm evaluators. https://arxiv.org/abs/2410.03131. ArXiv preprint arXiv:2410.03131.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yisheng Song, Ting Wang, Puyu Cai, Subrota K. Mondal, and Jyoti Prakash Sahoo. 2023. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. *ACM Comput. Surv.*, 55(13s).

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, and Xia Hu. 2025. Stop overthinking: A survey on efficient reasoning for large language models. https://arxiv.org/abs/2503.16419. ArXiv preprint arXiv:2503.16419.

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

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. 2025b. Towards thinking-optimal scaling of test-time compute for llm reasoning. https://arxiv.org/abs/2502.18080. ArXiv preprint arXiv:2502.18080.

A Experimental Prompts

To facilitate robust and reproducible semantic analysis of RLM-generated reasoning traces, we design two explicit prompting templates, detailed documented in Template 9 and Template 10.

Template 9 guides the model to cluster reasoning units into coherent, higher-level reasoning steps. This step provides context-aware candidate instances of clustering of long chain-of-thought tokens.

Template 10 is used to extract the semantic relationship between every pair of reasoning steps. By explicitly labeling each pair as `support`, `contradict`, or `independent`, this template enables the later probabilistic estimation of interdependencies within the model's chain-of-thought.

B Few-Shot Prompting Styles

Most existing research works reporting the performance degradation of RLM given few-shot prompting have not explicitly analyzed the structure of few-shot examples, while the concrete formulation of few-shot demonstrations could play a significant role on the behavior of language model. To isolate potential structural factors contributing to this performance degradation, in this paper, we introduce and analyze three distinct few-shot prompting style:
i) `Minimal:` Minimal exemplars containing only problem statements and final answers, without intermediate reasoning steps or explanatory content.
ii) `Concise:` Human-authored concise reasoning traces characterized by *short*, linear progression from problem formulation to solution with minimal exploration.
iii) `Explanatory:` RLM-generated *long* reasoning sequences featuring extensive problem space exploration, iterative verification mechanisms, and explicit self-reflection.

C Implementation Details

Models We evaluate a range of reasoning LLMs, including DeepSeek-R1-distilled-Llama-8B, DeepSeek-R1-distilled-Qwen-14B, and QWEN3-32B (Yang et al., 2025a). For conditional sampling (P_{LLM}), we adopt DEEPSEEK-V3-0324 (DeepSeek-AI et al., 2024).

Hyper-parameters To ensure reproducibility, we set the generation temperature to 0 when producing reasoning chains (CoT) with RLMs. During logical clustering and semantics detection processes, we keep $\tau_r \sim [0.3, 0.7]$ to ensure sampling diversity.

$\mathcal{P}_{\text{clu}}$ Logical Clustering

Instruction:
You are given a sequence of reasoning units, each representing a contiguous fragment from a language model's chain-of-thought (CoT) output. These units have typically been segmented using raw delimiters and may be overly fine-grained or fragmented for downstream analysis.
[*Input Template*]

Your task is to cluster consecutive reasoning units that are semantically connected, producing a concise and coherent set of higher-level reasoning steps. Each reasoning step should: - Combine all units that express a single coherent sub-task, logical inference, or closely related set of thoughts. Aim to group together units that collectively advance the same intermediate goal or logical point. - Ensure that each resulting reasoning step contains enough self-contained context to be analyzed independently, but avoid excessive merging that would result in overly broad or incoherent segments. - Maintain the original sequential order of reasoning. - Avoid splitting apart reasoning units that clearly belong to the same sub-problem or share strong contextual dependency. - Use concise yet informative titles for each reasoning step, reflecting its main logical function or purpose (e.g., "Restate Problem", "Recall Known Facts", "Solve Equation", "Synthesize Solution", etc.).

Expected Output Format:

```
{
  "s0": {"title": "...", "content": "..."},
  "s1": {"title": "...", "content": "..."},
  ...
}
```

where each `"sX"` key indexes an ordered reasoning step, with an appropriate `"title"` summarizing its logical purpose and `"content"` containing the merged, cleaned reasoning text.

Please ensure the output is structured, coherent, and well-suited for subsequent semantic analysis or graph-based modeling of the reasoning process.

Output: {"s_0": {"title":..., "content":...},...}

Figure 9: Complete instruction ($\mathcal{P}_{\text{clu}}$) for clustering reasoning units into logical cohesive reasoning steps.

Datasets All experiments are conducted on the GPQA-Diamond benchmark (Rein et al., 2023). To facilitate robust reasoning analysis and avoid potential training data contamination, we convert multiple-choice items into open-ended questions, requiring models to actively generate reasoning CoT as well as final answer rather than matching existing choices.

D algorithm

This appendix provides the detailed pseudocode for the core algorithmic components of our framework:

$\mathcal{P}_{\text{sem}}$ Semantics Detection

Instruction:
Given an ordered sequence of K reasoning steps, each representing a semantically meaningful stage in a language model's chain-of-thought output, your task is to systematically assess the semantic relationship between each pair of reasoning steps.
[*Logical Steps Template*]

For every ordered pair (i, j) with $1 \leq i < j \leq K$, your task is to decide whether step i: - `supports` step j (i.e., provides information, justification, intermediate results, or logical basis for step j), - `contradicts` step j (i.e., conflicts with, undermines, or provides an incompatible claim or result relative to step j), or - is `independent` of step j (i.e., is neither directly supportive nor contradictory; the steps are unrelated in logical content).
When making your decision, consider both explicit logical connections (e.g., mathematical derivation, use of previous results, direct contradiction) and more implicit semantic dependencies (e.g., fact recall enabling downstream calculations).

Expected Output Format:

```
{
  "(0,1)": "support",
  "(0,2)": "independent",
  "(1,2)": "contradict",
  ...
}
```

where each key `"(i,j)"` denotes an ordered pair of step indices (with $i < j$), and each value is one of `"support"`, `"contradict"`, or `"independent"`.

For each pair, provide only one label reflecting the most salient semantic relationship. If the relationship is unclear or borderline, default to `independent` unless clear evidence suggests support or contradiction.

Ensure that all pairs (i, j) with $1 \leq i < j \leq K$ are covered in the output, and that your decisions are consistent and justifiable based on the provided reasoning steps.

Output: {(S_0, S_1):..., (S_0, S_2): ..., (S_1, S_0): ...}

Figure 10: Complete instruction ($\mathcal{P}_{\text{sem}}$) for detecting semantical relationship among all reasoning steps.

ensemble-based clustering of reasoning units 1, and adaptive sampling-based construction of the semantic dependency graph 2. These algorithms operationalize the methods described in the main text, clarifying the iterative processes and statistical aggregation techniques used to ensure robust, uncertainty-aware structure induction from RLM outputs.

E Reference Step Length μ_{ref}

The ideal average step length μ_{ref} serves as a reference for the length-regularity term and is computed by dividing the total number of tokens N by a target number of reasoning steps K_{target}. We set $K_{\text{target}} = \min(\max(3, \lceil\sqrt{M}\rceil), 30)$, where M is the number of initial delimiter-based segments. This square-root heuristic with lower and upper bounds ensures μ_ref adapts to different CoT lengths and discourages both over-segmentation and overly coarse steps, enabling a scale-invariant and task-agnostic regularization.

Algorithm 1: Ensemble-Based Clustering of Reasoning Units

```
Input: Reasoning units $U$, clustering prompt $\mathcal{P}_{\text{cluster}}$, number of samples
       $B$, temperature grid $\{\tau_1, \ldots, \tau_B\}$
Output: Selected segmentation $C^*$
for $b \leftarrow 1$ to $B$ do
    $C^{(b)} \sim P_{\text{LLM}}(S \mid \mathcal{P}_{\text{cluster}}, U;\, \tau_b)$;
    $F^{(b)} \leftarrow F\big(C^{(b)}\big)$;
$C^* \leftarrow \arg\max_b F^{(b)}$;
return $C^*$;
```

F Standard-Error Derivation for the Signed Edge Confidence

Let a single LLM adjacency sample yield a label $c \in \{-1, 0, +1\}$ for the ordered pair (i, j), corresponding to `contradict`, `independent`, and `support`, respectively. Define the random variable

$$Z = \begin{cases} +1, & c = +1, \\ -1, & c = -1, \\ 0, & c = 0. \end{cases}$$

The *signed edge confidence* is the empirical mean of Z over R samples,

$$w_{ij} = \frac{1}{R} \sum_{r=1}^{R} Z^{(r)} = \hat{p}_{ij}(+1) - \hat{p}_{ij}(-1),$$

where $\hat{p}_{ij}(c) = \frac{1}{R} \sum_r \mathbf{1}[c^{(r)} = c]$.

Because $\mathbb{E}[Z] = p(+1) - p(-1)$, only the two informative labels contribute to the signed mean. The variance of Z under the true distribution p is

$$\begin{aligned} &\operatorname{Var}(Z) \\ &= (+1)^2 p(+1) + (-1)^2 p(-1) + 0^2 p(0) \\ &- \big[p(+1) - p(-1)\big]^2 \\ &= p(+1) + p(-1) - \big[p(+1) - p(-1)\big]^2. \end{aligned}$$

Replacing $p(\cdot)$ with their empirical estimates and dividing by R yields an unbiased standard-error estimator:

$$\mathrm{SE}_{ij} = \sqrt{\frac{1}{R} \sum_{l \in \{-1,+1\}} \hat{p}_{ij}(l)\big(1 - \hat{p}_{ij}(l)\big)}$$

The independent label ($l = 0$) contributes neither positive nor negative mass to Z; its influence is expressed implicitly via the complement $1 - p(+1) - p(-1)$. A full multinomial variance expression would add a term $-2\,p(+1)p(-1)$, whose magnitude is $O(p(+1)p(-1))$ and empirically negligible for our edge-sparse setting. Omitting this term simplifies the estimator while preserving the accuracy required for the adaptive stopping criterion

Algorithm 2: Adaptive Sampling-based Semantic Edge Construction

```
Input: Reasoning steps $S = (s_1, \dots, s_K)$, prompt $\mathcal{P}_{\text{sem}}$, confidence threshold $\varepsilon$,
       maximum samples $R_{\max}$, thresholds $\tau_{\text{pos}}, \tau_{\text{neg}}$
Output: Adjacency matrix $A$, edge weights $W$
Initialize counts: $C_{ij}(c) \leftarrow 0$ for all $i < j$ and $c \in \{-1, 0, +1\}$ ;
$r \leftarrow 0$;
repeat
    $r \leftarrow r + 1$;
    Sample $A^{(r)} \sim P_{\text{LLM}}(A \mid \mathcal{P}_{\text{sem}}, V; \tau_r)$ ;
    foreach $i < j$ do
        $c \leftarrow A^{(r)}_{ij}$;
        $C_{ij}(c) \leftarrow C_{ij}(c) + 1$;
    foreach $i < j$ do
        foreach $c \in \{-1, 0, +1\}$ do
            $\hat{p}_{ij}(c) \leftarrow \frac{C_{ij}(c)}{r}$;
        $\mathrm{SE}_{ij} \leftarrow \sqrt{\frac{1}{r} \sum_{l \in \{-1,+1\}} \hat{p}_{ij}(l)(1 - \hat{p}_{ij}(l))}$;
until $\max_{i<j} \mathrm{SE}_{ij} \leq \varepsilon$ or $r \geq R_{\max}$;
foreach $i < j$ do
    $w_{ij} \leftarrow \hat{p}_{ij}(+1) - \hat{p}_{ij}(-1)$;
    if $w_{ij} \geq \tau_{pos}$ then
        $A_{ij} \leftarrow +1$, $A_{ji} \leftarrow -1$;
    else
        if $w_{ij} \leq -\tau_{neg}$ then
            $A_{ij} \leftarrow -1$, $A_{ji} \leftarrow +1$;
        else
            $A_{ij} \leftarrow 0$, $A_{ji} \leftarrow 0$;
    $W_{ij} \leftarrow w_{ij}$, $W_{ji} \leftarrow -w_{ij}$;
return $A$, $W$;
```