Graph of Verification: Structured Verification of LLM Reasoning with Directed Acyclic Graphs

Jiwei Fang[1], **Bin Zhang**[2], **Changwei Wang**[3], **Jin Wan**[3], **Zhiwei Xu**[1*]
[1] Shandong University
[2] The Key Laboratory of Cognition and Decision Intelligence for Complex Systems, Institute of Automation, Chinese Academy of Sciences
[3] Qilu University of Technology

Abstract

Verifying the complex and multi-step reasoning of Large Language Models (LLMs) is a critical challenge, as holistic methods often overlook localized flaws. Step-by-step validation is a promising alternative, yet existing methods are often rigid. They struggle to adapt to diverse reasoning structures, from formal proofs to informal natural language narratives. To address this adaptability gap, we propose the Graph of Verification (GoV), a novel framework for adaptable and multi-granular verification. GoV's core innovation is its flexible *node block* architecture. This mechanism allows GoV to adaptively adjust its verification granularity—from atomic steps for formal tasks to entire paragraphs for natural language—to match the native structure of the reasoning process. This flexibility allows GoV to resolve the fundamental trade-off between verification precision and robustness. Experiments on both well-structured and loosely-structured benchmarks demonstrate GoV's versatility. The results show that GoV's adaptive approach significantly outperforms both holistic baselines and other state-of-the-art decomposition-based methods, establishing a new standard for training-free reasoning verification.

Code — https://github.com/Frevor/Graph-of-Verification
Extended version — https://arxiv.org/abs/2506.12509

Introduction

Despite the remarkable progress of Large Language Models (LLMs) in complex reasoning tasks (Achiam et al. 2023; Shao et al. 2024; Yang et al. 2024; Grattafiori et al. 2024), verifying the fidelity of their reasoning processes remains a fundamental challenge, especially in domains involving intricate mathematics and logic. Beyond occasional computational or logical mistakes that lead to incorrect outcomes, a more concerning issue is that LLMs frequently produce reasoning steps that appear plausible yet contain subtle logical flaws, even when the final answer happens to be correct (He et al. 2025; Luo et al. 2024). Consequently, there is a critical need for methods that can rigorously and scalably assess the internal validity of LLM-generated reasoning, ensuring the dual correctness of both the process and the outcome.

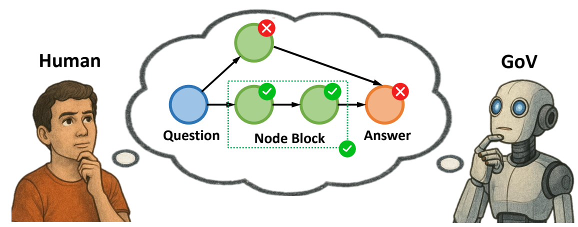

Figure 1: Both humans and GoV approach reasoning validation by decomposing the process into a directed acyclic graph. The granularity of these validation units is highly flexible, allowing for adaptation to different tasks.

To tackle this verification challenge, we draw inspiration from a fundamental aspect of human cognition. When faced with a complex argument, humans instinctively decompose it into a series of smaller, dependent steps, verifying the logic sequentially. This process can be naturally modeled as a directed acyclic graph (DAG) (Wavering 1989), as illustrated in Figure 1. This intuitive "divide and conquer" strategy stands in stark contrast to the prevailing approach in LLM verification: **holistic verification**, where the entire reasoning process is evaluated in a single pass. Such a monolithic approach often overwhelms the verifier LLM with excessive cognitive load, hindering its ability to detect localized flaws, a vulnerability that becomes particularly pronounced as the reasoning chain lengthens and complexity increases. We argue that embracing a structured, decomposition-based approach is the indispensable path toward reliable reasoning evaluation.

However, this divide-and-conquer principle reveals a deeper, more fundamental challenge: how should one effectively decompose the ubiquitous, loosely-structured reasoning expressed in natural language? While formal proofs can be straightforwardly decomposed along its explicit logical structure, the field currently lacks a unified framework capable of navigating the entire spectrum of reasoning—from the highly structured to the highly informal. This adaptability gap is a critical barrier to achieving truly reliable reasoning evaluation.

To address this challenge, we propose **G**raph **o**f **V**erification (**GoV**), a novel framework designed specifically for **adaptability** and **multi-granular verification**. The core

*Corresponding author: zhiwei_xu@sdu.edu.cn

innovation of GoV is its flexible **node block** architecture. This mechanism allows the verification granularity to be configured according to the native structure of the reasoning process. For well-structured tasks, a node block can be a single atomic statement, enabling maximum precision. In contrast, for loosely-structured natural language narratives, it can be a semantically coherent paragraph, ensuring robustness. By dynamically selecting the most appropriate granularity, GoV resolves the tension between precision and robustness, offering a unified and effective verification solution for diverse reasoning tasks.

Our primary contributions are as follows:

- To the best of our knowledge, we are the first to formalize decomposed verification as a **two-dimensional design space** characterized by *Verification Granularity* and *Contextual Scope*, offering a principled perspective for verifying both well-structured and loosely-structured text.
- We propose **GoV**, a novel and flexible multi-granular verification framework. Its core node block mechanism enables the verification strategy to be adapted to the structural characteristics of the task.
- We demonstrate the effectiveness and generality of GoV through experiments on both well-structured and loosely-structured tasks, showcasing its powerful ability to solve diverse verification challenges in different configurations.

Related Work

Reasoning in Large Language Models

Complex reasoning is a hallmark of advanced LLMs, a capability primarily validated on mathematical and scientific reasoning benchmarks that have become the principal litmus test for their power. A key advancement in this area is Chain-of-Thought prompting (Wei et al. 2022), which elicits a step-by-step reasoning process from the model, significantly improving its performance on tasks requiring complex planning and computation. Building on this, subsequent research has explored more sophisticated reasoning structures, such as exploring multiple reasoning paths with Tree-of-Thoughts (Yao et al. 2023) or modeling non-linear dependencies with Graph-of-Thoughts (Besta et al. 2023). This body of work has established a clear consensus: a structured reasoning process is crucial for enhancing the quality of solutions generated by LLMs. Our work builds upon this principle, shifting the focus from **generating** structured reasoning to **validating** it.

Verification of LLM Reasoning

Although generating more elaborate reasoning processes is beneficial, systematically verifying their correctness remains a significant challenge (Hong et al. 2023; Stechly, Valmeekam, and Kambhampati 2024; Gu et al. 2024; Zheng et al. 2023). To address this issue, existing verification approaches can be broadly categorized as follows.

Holistic Verification and Training-Based Verifiers A common approach is holistic verification, where an LLM is prompted to evaluate an entire reasoning chain at once. However, this method often falters when dealing with long and complex reasoning chains, as it imposes a heavy cognitive load on the verification model. Another paradigm involves training-based verifiers, most notably process-based reward models (Lightman et al. 2023; Wang et al. 2023; Zhang et al. 2025), which are fine-tuned on human-annotated data to score or classify intermediate reasoning steps. While effective, these models require substantial labeled data, are costly to train, and may struggle to keep pace with the rapidly evolving capabilities of frontier LLMs, necessitating expensive retraining cycles. In contrast, our Graph-of-Verification (GoV) framework is **training-free**, making it a more agile and resource-efficient solution.

Decomposition-Based Verifiers To overcome the limitations of holistic evaluation, an emerging and powerful trend is decomposition-based verification, which breaks down the reasoning chain into smaller, more manageable units for stepwise validation. Recent works, such as PARC (Mukherjee et al. 2025) and Deductive Verification (Ling et al. 2023), have demonstrated the potential of this approach. These methods typically aim for highly granular verification, focusing on validating atomic reasoning steps against the minimal necessary premises. This is achieved either through post-hoc premise extraction or by constraining the generation process to a ”natural program” format that explicitly declares premises.

However, the pursuit of atomic granularity and minimal premises, while theoretically ideal, relies on fragile premise extraction or declaration steps that are often unreliable in loosely structured natural language contexts. In contrast, GoV is the first framework explicitly designed to address and navigate this challenge. By introducing a multi-granularity architecture with node blocks, GoV can adapt its strategy—from fine-grained verification for well-structured tasks to a more robust, coarse-grained approach for natural language—thereby offering a more unified and practical solution to reasoning verification.

Graph of Verification

In this section, we first establish the conceptual foundation of our work by defining the design space for decomposed verification. This framing clarifies the core trade-offs that any such verification strategy must navigate. We then formally present the Graph of Verification (GoV), a framework whose components and mechanisms are specifically designed to operate flexibly within this space. The overall framework of GoV is illustrated in Figure 2.

The Design Space of Decomposed Verification

While the divide-and-conquer principle is central to verifying complex reasoning, the critical question of how to best decompose a reasoning chain remains underexplored. We posit that any decomposition strategy can be characterized within a novel two-dimensional design space: Verification Granularity and Contextual Scope.

Dimension 1: Verification Granularity The first dimension, verification granularity, defines the scale of the reasoning unit being validated. This dimension spans a spectrum:

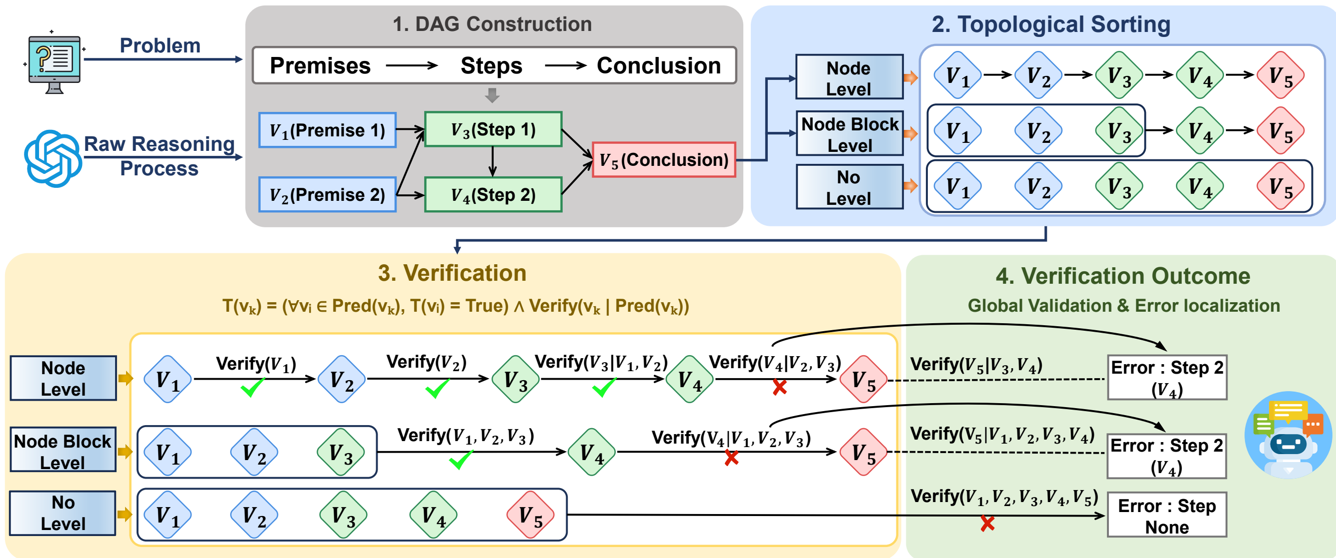

Figure 2: The GoV Four-Stage Verification Pipeline. GoV operationalizes structured validation by first modeling reasoning as a (1) Directed Acyclic Graph (DAG). A (2) topological sort then enforces causal consistency, ensuring premises are verified before conclusions. Following this order, (3) sequential verification is performed by an LLM which assesses each unit based on previously validated antecedents. The process (4) terminates at the first detected error, enabling precise fault localization and guaranteeing the logical soundness of any fully verified reasoning chain.

- **Atomic Granularity:** On the fine-grained end, the reasoning is decomposed into the smallest possible, indivisible logical or computational units (e.g., a single arithmetic operation '`a + b = c`'). This approach offers the highest precision and makes error localization exact, but it imposes strict requirements on the structural clarity of the reasoning process.
- **Block Granularity:** On the coarse-grained end, multiple related atomic steps are aggregated into a single, semantically coherent unit for verification (e.g., a full paragraph explaining a particular stage of a solution). This approach is more robust and adaptable to the native structure of natural language reasoning.

Dimension 2: Contextual Scope Contextual scope defines how much historical information is provided as the premise for verifying the current unit. This also exists on a spectrum:

- **Minimal Context:** This approach provides only the theoretically necessary direct premises required to validate the current unit. Its advantage is minimizing cognitive load and reducing distraction, but it relies on a premise extraction process that can itself be fallible.
- **Inclusive Context:** This approach provides a broader set of all previously verified information as context. It is a safer and more robust strategy but risks introducing redundant information.

The optimal configuration within this design space is **NOT** universal; rather, it is contingent on the inherent structure of the reasoning process itself. For *well-structured reasoning*, such as formal proofs or programmatic calculations, a strategy of *Atomic Granularity* and *Minimal Context* is often viable and offers the highest precision. However, for the more common case of *loosely-structured reasoning* found in natural language narratives, this strategy becomes brittle. The difficulty in reliably performing atomic decomposition and premise extraction necessitates a more robust approach, typically favoring *Block Granularity* and *Inclusive Context*. The lack of a framework that can flexibly navigate these tradeoffs and adapt to this structural spectrum is a critical gap in current research. The GoV framework, which we introduce next, is designed precisely to fill this gap.

Reasoning with Directed Acyclic Graphs

The GoV framework is built upon the foundational principle of modeling any reasoning process as a Directed Acyclic Graph (DAG). We adopt this structure because its alignment with the directional and non-circular nature of logical deduction provides the ideal backbone for systematic, decomposed verification. Reasoning progresses naturally from foundational premises to derived conclusions, a characteristic effectively represented by the directed edges of a DAG. This *directionality* ensures that each inference builds upon previously established truths, thereby eliminating ambiguity. Moreover, the *acyclic* property of the DAG guarantees well-founded reasoning by removing circular dependencies, thus preventing situations where the validity of a statement relies, whether directly or indirectly, on itself.

Therefore, GoV posits that any reasoning process $\mathcal{S}$ can be formally modeled as a DAG $\mathcal{G} = (V, E)$, where $\mathcal{G}$ serves as the backbone for systematic verification. We now provide a detailed description of the two components that constitute the graph.

Node The set of nodes $V = \{v_1, v_2, \ldots, v_n\}$ in $\mathcal{G}$ comprises the fundamental verification units within the reasoning process. Each node $v_i \in V$ typically corresponds to a distinct statement, such as an initial premise, an axiom, an intermediate derived conclusion, or a final assertion. These statements can be broadly categorized:

- **Foundational Elements (Premises, Axioms, Facts):** These are the exogenous inputs or fundamental truths forming the starting points of reasoning, often extracted from the problem statement or established domain knowledge.
- **Inferred Statements (Intermediate Conclusions):** These represent new propositions derived from foundational elements or previously validated inferred statements through the application of logical rules, mathematical operations, or other inferential steps.
- **Terminal Statements (Final Conclusions, Goals):** These signify the ultimate outputs or target states the reasoning process aims to establish.

Edge The set of directed edges $E \subseteq V \times V$ encodes the direct logical dependencies or inferential links between these statements. An edge $(v_i, v_j) \in E$ signifies that statement v_i serves as a *direct* premise or justification for the derivation or assertion of statement v_j. Consequently, the epistemic validity of v_j is contingent upon the validity of v_i (and potentially other co-premises) and the soundness of the inferential step (v_i, v_j) connecting them.

In summary, by modeling reasoning as a DAG, the GoV framework ensures a transparent, structured, and non-circular verification process, where each statement is grounded in a traceable chain of valid inferences.

Multi-Granular Topological Units

To realize the adaptable granularity required to navigate the design space, GoV introduces two primary types of verification units: Atomic Nodes and Node Blocks. This multi-granular architecture is a crucial design feature of GoV. Before detailing these units, we first establish the verification sequence using a topological sort of the graph: for any DAG $\mathcal{G}$, there exists a vertex ordering $\sigma : V \to \{1, 2, \ldots, n\}$ such that $\sigma(v_i) < \sigma(v_j)$ holds for every edge $(v_i, v_j) \in E$. The derived verification sequence $\mathcal{C}_{\text{verif}} = (c_1, c_2, \ldots, c_n)$, where $\sigma(c_k) = k$, ensures each statement is verified strictly after all its logical antecedents, thereby preventing dependency violations.

With the verification order established, we now define the units themselves:

Atomic Nodes: At the finest granularity, atomic node verification treats each vertex $v_k \in V$ as an independent unit. This mode achieves maximum error localization capability through complete stepwise decomposition.

Node Blocks: While atomic decomposition offers maximum precision, it is often brittle and impractical for loosely-structured natural language, as argued in our design space framework. GoV's Node Block abstraction directly addresses this by grouping topologically consecutive nodes into logically coherent units $\mathcal{B} = (B_1, \ldots, B_m)$:

$$\mathcal{C}_{\text{verif}} = (\underbrace{c_1, \ldots, c_{|B_1|}}_{B_1}, \ldots, \underbrace{c_{n-|B_m|+1}, \ldots, c_n}_{B_m}). \quad (1)$$

This coarser granularity reduces verification invocations to $m \leq n$ while enforcing two critical constraints: (i) Topological consistency: for any B_j, B_K with $j < k$, $\max_{v \in B_j} \sigma(v) < \min_{v \in B_k} \sigma(v)$, preserving macro-level dependency relationships; (ii) Semantic cohesion: each block must correspond to complete logical units (e.g., lemmas in proofs or paragraphs in narratives) to prevent arbitrary fragmentation. Notably, when $m = 1$ (i.e., the entire reasoning DAG is treated as a single verification block), the framework degenerates to conventional monolithic verification approaches.

This adaptable architecture of Atomic Nodes and Node Blocks is the core mechanism that allows GoV to be flexibly configured. By choosing the appropriate granularity, users can position their verification strategy at the optimal point within the design space described, balancing precision against robustness to suit the specific reasoning format.

GoV Verification Mechanism

Having defined the multi-granular units of GoV, we now detail the verification mechanism itself. The verification of each unit (atomic node or node block) within the GoV framework is performed by an LLM, guided by two core principles: (i) respecting the topological order imposed by $\mathcal{C}_{\text{verif}}$ or its block-wise partitioning $\mathcal{B}$, and (ii) ensuring the availability of all requisite premises that have been previously verified. The verification function, applied to a target unit, either a node c_k or a block B_j, is defined as $\text{Verify}(\cdot,\ Pred_{\text{prov}})$, which determines the validity of the unit based on the provided premises $Pred_{\text{prov}}$, and returns a Boolean outcome (True or False). The implementation of GoV involves verifying each individual node or node block, and further extends this mechanism to support holistic reasoning across the entire inference process.

Verifying an Atomic Node c_k The set of *direct logical predecessors* (parents) for an atomic node c_k is $Pred(c_k) = \{c_i \mid (c_i, c_k) \in E\}$. The premises provided to the LLM, $Pred_{\text{prov}}(c_k)$, must at least include all nodes in $Pred(c_k)$. Moreover, every node in $Pred_{\text{prov}}(c_k)$ must have been previously verified as True and must appear before c_k in the topological ordering $\mathcal{C}_{\text{verif}}$. That is, $Pred_{\text{prov}}(c_k) \subseteq \{c_1, \ldots, c_{k-1}\}$ where $T(c_i) = \text{True}$ for all $c_i \in Pred_{\text{prov}}(c_k)$.

- If $Pred(c_k) = \emptyset$ (i.e., c_k is a foundational element), its validity $T(c_k)$ is evaluated directly as $T(c_k) = \text{Verify}(c_k, \emptyset)$, typically by validating it against the problem statement or established knowledge.
- If $Pred(c_k) \neq \emptyset$, then c_k is a derived statement. Its validity $T(c_k)$ depends both on the validity of all its immediate predecessors and the soundness of the inference process itself, formally expressed as:

$$\begin{aligned} T(c_k) = &\big(\forall c_i \in Pred(c_k), T(c_i) = \text{True}\big) \\ &\wedge \text{Verify}\big(c_k, Pred_{\text{prov}}(c_k)\big). \end{aligned} \quad (2)$$

Verifying a Node Block B_j For a node block B_j, the minimal set of *external* direct prerequisite nodes is defined as:

$$Pred_{\text{ext}}(B_j) = \Big(\bigcup_{v \in V(B_j)} Pred(v) \Big) \setminus V(B_j). \quad (3)$$

The set of premises provided to the LLM, denoted as $Pred_{\text{prov}}(B_j)$, must cover all information corresponding to $Pred_{\text{ext}}(B_j)$ and consist solely of previously verified blocks that precede B_j in the topological ordering. In practical scenarios where node blocks correspond to natural language paragraphs (such as in the ProcessBench experiments), $Pred_{\text{prov}}(B_j)$ typically includes the full content of all preceding validated blocks $(B_1, \ldots, B_{j-1})$. The LLM then performs Verify$(B_j, Pred_{\text{prov}}(B_j))$ to assess the internal coherence of B_j and validate all statements c_k within B_j based on both the provided premises and any intermediate conclusions derived within the block.

Overall Verification and Error Handling The verification process proceeds sequentially along $\mathcal{C}_{\text{verif}}$ (or $\mathcal{B}$). If any unit c_k (or block B_j) is found to be False, it is identified as the earliest point of failure, and the GoV process typically halts, recording this failure. The entire reasoning trajectory $\mathcal{S}$ is considered valid if and only if all its constituent units (nodes or blocks) are successfully verified as True.

In summary, GoV allows the verification to be conducted on fine-grained atomic nodes or on coarser-grained node blocks (formed from these nodes), depending on the user's definition tailored to the specific task. In addition, GoV permits a degree of flexibility in selecting $Pred_{\text{prov}}$, allowing the inclusion of a validated superset of direct prerequisites, provided they precede the current unit, when such extended context is expected to facilitate the LLM's assessment. This structured, iterative, and premise-driven scheme makes GoV's reasoning validation highly reliable.

Experiments

To evaluate the practical effectiveness of GoV, we conduct empirical studies on two contrasting reasoning scenarios: one with well-structured, formally defined arithmetic dependencies, and another with loosely-structured, natural language explanations. These two case studies serve complementary purposes: the first highlights GoV's capability for fine-grained, high-precision error localization in mathematically rigid tasks, while the second demonstrates its robustness and adaptability in the more challenging setting of free-form reasoning. Together, they showcase GoV's versatility and superiority in both deterministic and open-ended verification environments.

To mitigate the influence of model randomness, all models are configured with a temperature of *zero*. Furthermore, to comprehensively assess verification performance, we employ the following evaluation metrics:

- **Correct Accuracy** The proportion of fully correct reasoning processes that are judged as "correct" by the model.
- **Error Accuracy** The proportion of reasoning processes containing errors in which the model successfully identifies the first erroneous step.

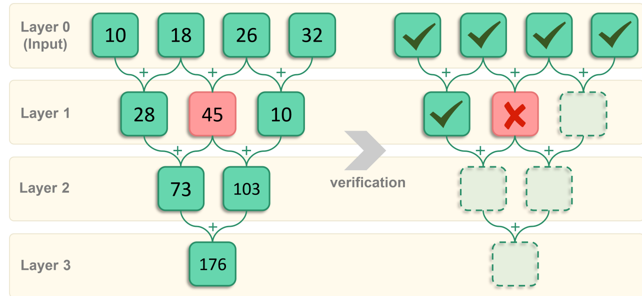

Figure 3: Illustration of the numerical validation procedure for Number Triangle Summation.

- **F1** The harmonic mean of correct accuracy and error accuracy, serving as the primary metric for evaluating overall model performance.

Number Triangle Summation

To validate GoV's effectiveness in a well-structured reasoning environment, we first introduce the Number Triangle Summation benchmark. This task is designed to assess verification accuracy on complex, multi-step reasoning where the dependency graph is explicit and unambiguous. An illustration of the corresponding verification task is shown in Figure 3. The reasoning process is defined as follows: starting from N initial numbers at the zeroth layer, each subsequent layer is formed by summing every pair of adjacent numbers from the previous layer. This process continues iteratively until a single number remains at the final layer. Notably, the verification order inherent in this task can be naturally represented as a DAG. Verifying the entire reasoning process is inherently long and complex, as the correctness of each intermediate result relies heavily on preceding inference steps. Existing methods often struggle to reliably verify such multi-step reasoning chains, or fail to accurately localize the specific point at which the reasoning breaks down.

To validate GoV's effectiveness and scalability in a well-structured reasoning environment, we introduce the Number Triangle Summation benchmark. To analyze how verification performance changes with problem complexity, we systematically vary the number of initial inputs (N), which determines the length and depth of the reasoning chain. We generated datasets for $N \in \{2, 4, 6, 8\}$. For each value of N, we programmatically generated 500 instances. The base layer (Layer 0) of each instance consists of N randomly sampled positive integers between 1000 and 9999. The solution steps for each subsequent layer were automatically derived by the program. To simulate real-world errors, each reasoning process had a 50% chance of being corrupted by a single-digit perturbation in one of the addition results. This design allows us to directly measure the performance of verification methods as the cognitive load increases.

Given the task's explicit mathematical structure, we configured GoV to operate at the optimal point in our defined design space for such problems: Atomic Granularity for maximum precision and Minimal Context to reduce cognitive load on the verifier LLM. Specifically:

- **Atomic Granularity:** Each individual addition equation is treated as an atomic node in the verification graph. This allows for the most fine-grained error localization possible.
- **Minimal Context:** During the verification of each node, we provide only its direct prerequisites—the two operands from the previous layer—as the contextual input ($Pred_{\text{prov}}(v_k)$). This minimizes the cognitive load on the verifier LLM by excluding all irrelevant information.

The verification process then proceeds along the natural top-down hierarchy of the triangle, which corresponds to a topological sort of the graph, halting at the first detected error. We conduct experiments using different LLMs to compare the performance of the proposed GoV framework with that of baseline approaches.

Model	Method	F1 Score by Problem Size (N)			
		N=2	**N=4**	**N=6**	**N=8**
Qwen2.5-7B -Instruct	Holistic Verification	82.3	89.5	72.8	51.5
	GoV (Ours)	**97.1**	**89.6**	**84.8**	**70.7**
Qwen2.5-14B -Instruct	Holistic Verification	97.1	94.2	70.8	62.9
	GoV (Ours)	**99.0**	**95.1**	**93.9**	**86.8**
Qwen2.5-32B -Instruct	Holistic Verification	**100.0**	97.0	80.6	56.3
	GoV (Ours)	99.8	**99.4**	**95.8**	**91.4**
Qwen2.5-72B -Instruct	Holistic Verification	**99.6**	93.3	46.3	49.5
	GoV (Ours)	98.7	**97.9**	**98.1**	**98.1**

Table 1: F1 scores on the Number Triangle Summation task with varying problem sizes (N).

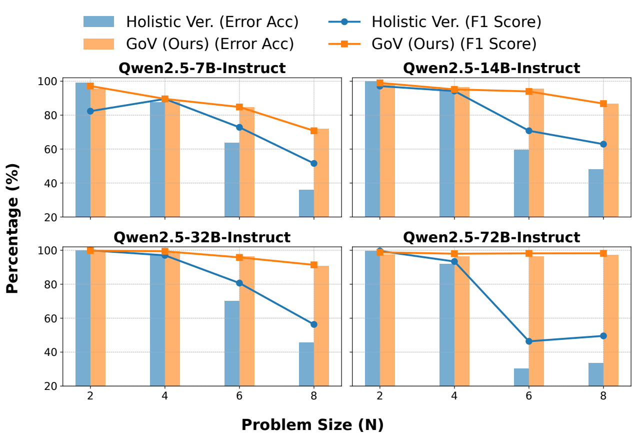

Figure 4: The performance of GoV and holistic verification on Number Triangle Summation.

As shown in Table 1 and Figure 4, all models based on the GoV framework demonstrate substantial improvements in F1 scores. Under the baseline setting (CoT-based holistic verification), although the prompt instructs the LLM to examine each equation individually and the generated reasoning chains seem to reflect such step-by-step checking, the verification remains unfaithful. This observation is consistent with the conclusions reported in the previous literature (Radhakrishnan et al. 2023; Lindsey et al. 2025).

In contrast, GoV's success can be directly attributed to its configured strategy, which is perfectly matched to the task's structure. By operating at an atomic granularity, it decomposes the complex task into a series of simple, independently verifiable checks. Concurrently, by providing only a minimal context, it eliminates distracting information and allows the verifier LLM to function with maximum reliability. This experiment serves as a powerful validation of our framework's core principle: when a task's structure permits, a high-precision configuration of GoV can achieve outstanding verification performance.

ProcessBench

Having validated GoV's high-precision capabilities in a well-structured environment, we now evaluate its adaptability in the more common and challenging domain of loosely-structured natural language reasoning. To this end, we utilize the ProcessBench benchmark (Zheng et al. 2024), a dataset designed to evaluate models on a variety of reasoning errors. Each test instance consists of a question, a step-by-step solution generated by a LLM, and a human-annotated label indicating the index of the first incorrect step. If the solution is entirely correct, the label is assigned as -1. We use the full test sets from three ProcessBench subsets: *GSM8K* (elementary-level math problems), *MATH* (more advanced math problems), and *OlympiadBench* (Olympiad-level math problems), comprising 400, 1000, and 1000 samples, respectively. These subsets collectively span a broad range of reasoning difficulty, offering representative coverage of typical mathematical tasks.

To comprehensively evaluate GoV's performance, we compare it against two distinct verification paradigms: **Holistic Verification:** This represents the standard practice, where the entire reasoning process is provided to an LLM verifier in a single pass. We use the official results from the ProcessBench leaderboard as our baseline. **Decomposition with Premise Extraction (PARC):** As a state-of-the-art decomposition-based method, PARC first attempts to identify the minimal premises for each reasoning step before verification. This allows for a direct comparison of GoV's pragmatic approach against strategies that pursue premise minimality.

To address the challenges of loosely-structured reasoning, where atomic decomposition is brittle and impractical, we configured GoV for its high-robustness mode. This strategy, guided by our design space, entails:

- **Block Granularity:** We treat each natural language paragraph as a node block (B_j). This choice leverages the text's inherent semantic structure, aligning our verification units with the author's logical stages.
- **Inclusive Context:** During the verification of a given block, we provide an inclusive context ($Pred_{\text{prov}}$), which consists of the full text of all previously validated paragraphs. This pragmatic approach ensures all potentially relevant information is available, critically avoiding the risks associated with a fragile premise extraction step.

As we will demonstrate, this robust configuration is key to GoV's superior performance on natural language tasks.

Model	Method	GSM8K			MATH			OlympiadBench		
		Correct Acc	Wrong Acc	F1	Correct Acc	Wrong Acc	F1	Correct Acc	Wrong Acc	F1
Qwen2.5-7B -Instruct	Holistic Verification	66.3	36.7	47.3	63.8	23.7	34.6	46.0	25.4	32.7
	PARC	60.1	38.6	47.0	45.6	41.2	43.3	-	-	-
	GoV (Ours)	89.1	43.0	**58.0**	80.7	41.7	**55.0**	74.9	30.1	**42.9**
Qwen2.5-14B -Instruct	Holistic Verification	93.8	47.8	63.3	86.9	40.4	55.2	76.4	30.9	44.0
	GoV (Ours)	92.2	60.3	**72.9**	90.6	56.0	**69.2**	83.4	42.3	**56.2**
Qwen2.5-32B -Instruct	Holistic Verification	97.9	43.0	59.8	95.6	33.3	49.4	90.0	22.4	35.9
	PARC	95.9	55.1	70.0	86.9	53.9	66.5	-	-	-
	GoV (Ours)	94.3	66.6	**78.1**	89.1	61.4	**72.7**	79.9	55.0	**65.2**
Qwen2.5-72B -Instruct	Holistic Verification	98.4	61.4	75.6	91.9	45.3	60.7	88.5	33.7	48.9
	PARC	97.8	59.7	74.1	86.7	53.9	66.5	-	-	-
	GoV (Ours)	98.9	67.6	**80.3**	90.3	64.4	**75.2**	84.0	49.4	**62.2**
Llama3.3-70B -Instruct	Holistic Verification	96.9	66.2	**78.6**	93.1	38.4	54.4	90.0	30.9	46.0
	GoV (Ours)	92.2	68.6	**78.6**	79.3	65.1	**71.5**	64.6	57.0	**60.5**
GPT4.1-Nano	Holistic Verification	89.1	44.9	59.7	75.8	55.2	63.9	71.1	47.3	56.8
	GoV (Ours)	92.2	62.3	**74.3**	79.3	67.0	**72.6**	67.2	57.6	**62.0**

Table 2: Performance comparison on the ProcessBench benchmark. GoV achieves superior F1 scores compared to holistic and PARC baselines consistently across all models and datasets. Results for PARC are sourced from the authors' public rebuttal, as its code was not available at the time of our experiments

A cornerstone of GoV's power is its practical flexibility, which allows verification strategies to be customized to the native structure of the reasoning process. Crucially, this showcases that GoV circumvents the need for a rigid, predefined atomic-level DAG. Instead, its adaptable node block system is engineered to leverage the inherent, macroscopic logical structure found in various reasoning formats. The superiority of this flexible design is confirmed by our experimental results, where GoV consistently outperforms more rigid approaches.

As shown in Table 2, GoV achieves state-of-the-art performance, consistently and significantly outperforming all comparative methods across the various models and datasets.

Superiority over Holistic Verification: Compared to the holistic baseline, GoV's success confirms the fundamental benefit of decomposition, reducing cognitive load and enabling more focused, accurate verification. The pronounced gains on complex datasets like *OlympiadBench* (e.g., F1 score from 35.9% to 62.2% for *Qwen2.5-32B-Instruct*) highlight this advantage.

Superiority over PARC: More importantly, GoV consistently outperforms PARC, a fellow decomposition-based framework. We attribute this significant advantage directly to GoV's adaptive strategy. While PARC relies on a fragile, LLM-based step to extract minimal premises, GoV's Block Granularity and Inclusive Context strategy proves far more robust in the noisy context of natural language. This result provides powerful empirical evidence for our central thesis: for loosely-structured tasks, a verification strategy that prioritizes decompositional robustness over contextual minimality is superior.

In summary, these results demonstrate that GoV's adaptive, pragmatic approach to decomposition enables a new state of the art in training-free reasoning verification.

Conclusion

In this work, we addressed the critical challenge of verifying complex LLM reasoning, where existing methods often struggle with the trade-off between precision and robustness across diverse reasoning structures. We introduced the Graph of Verification (GoV), a novel, training-free framework that models reasoning as a Directed Acyclic Graph and leverages a flexible node block architecture for adaptable, multi-granular verification.

Our primary contribution is the formalization of decomposed verification within a two-dimensional design space of Verification Granularity and Contextual Scope. Building on this, we developed GoV, a practical framework that can navigate this space, tailoring its verification strategy—from fine-grained atomic checks for highly-structured tasks to robust paragraph-level blocks for loosely-structured natural language narratives. Our empirical evaluation on contrasting benchmarks demonstrated GoV's superiority over both holistic baselines and other decomposition-based methods. The results validate our central hypothesis: adaptability is key to achieving reliable verification, and GoV's pragmatic approach of prioritizing robustness over contextual minimality is crucial for narrative reasoning.

While GoV establishes a new standard for training-free verification, we recognize avenues for future work. The current segmentation of reasoning into nodes or blocks relies on the task's inherent structure; developing automated methods to parse unstructured reasoning into optimal GoV graphs is a compelling research direction. Furthermore, extending GoV from a pure verifier to an interactive "Graph of Correction" framework, where it not only identifies but also proposes and validates fixes for flawed reasoning steps, presents an exciting frontier.

Ultimately, Graph of Verification is a significant step toward transparent, scrutable, and reliable AI. By providing a structured lens to inspect LLM reasoning, we pave the way for trust in not only the final answers but the processes that produce them.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62506210), Taishan Scholars Program No.TSQN202507241, Key R&D Program of Shandong Province, China, No.2025KJHZ013, Shandong Provincial University Youth Innovation and Technology Support Program No.2022KJ291, Shandong Provincial Natural Science Foundation for Young Scholars Program No.ZR2025QC1627 and Qilu University of Technology (Shandong Academy of Sciences) Youth Outstanding Talent Program No. 2024QZJH02.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Besta, M.; Blach, N.; Kub'ivcek, A.; Gerstenberger, R.; Gianinazzi, L.; Gajda, J.; Lehmann, T.; Podstawski, M.; Niewiadomski, H.; Nyczyk, P.; and Hoefler, T. 2023. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. In *AAAI Conference on Artificial Intelligence*.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gu, J.; Jiang, X.; Shi, Z.; Tan, H.; Zhai, X.; Xu, C.; Li, W.; Shen, Y.; Ma, S.; Liu, H.; Wang, Y.; and Guo, J. 2024. A Survey on LLM-as-a-Judge. *ArXiv*, abs/2411.15594.

He, Y.; Li, S.; Liu, J.; Wang, W.; Bu, X.; Zhang, G.; Peng, Z.; Zhang, Z.; Zheng, Z.; Su, W.; and Zheng, B. 2025. Can Large Language Models Detect Errors in Long Chain-of-Thought Reasoning? *ArXiv*, abs/2502.19361.

Hong, R.; Zhang, H.; Pang, X.; Yu, D.; and Zhang, C. 2023. A Closer Look at the Self-Verification Abilities of Large Language Models in Logical Reasoning. *ArXiv*, abs/2311.07954.

Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; and Cobbe, K. 2023. Let's Verify Step by Step. *ArXiv*, abs/2305.20050.

Lindsey, J.; Gurnee, W.; Ameisen, E.; Chen, B.; Pearce, A.; Turner, N. L.; Citro, C.; Abrahams, D.; Carter, S.; Hosmer, B.; Marcus, J.; Sklar, M.; Templeton, A.; Bricken, T.; McDougall, C.; Cunningham, H.; Henighan, T.; Jermyn, A.; Jones, A.; Persic, A.; Qi, Z.; Thompson, T. B.; Zimmerman, S.; Rivoire, K.; Conerly, T.; Olah, C.; and Batson, J. 2025. On the Biology of a Large Language Model. *Transformer Circuits Thread*.

Ling, Z.; Fang, Y.; Li, X.; Huang, Z.; Lee, M.; Memisevic, R.; and Su, H. 2023. Deductive Verification of Chain-of-Thought Reasoning. *ArXiv*, abs/2306.03872.

Luo, L.; Liu, Y.; Liu, R.; Phatale, S.; Guo, M.; Lara, H.; Li, Y.; Shu, L.; Zhu, Y.; Meng, L.; et al. 2024. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*.

Mukherjee, S.; Chinta, A.; Kim, T.; Sharma, T. A.; and Tur, D. H. 2025. Premise-Augmented Reasoning Chains Improve Error Identification in Math reasoning with LLMs. In *Forty-second International Conference on Machine Learning*.

Radhakrishnan, A.; Nguyen, K.; Chen, A.; Chen, C.; Denison, C.; Hernandez, D.; Durmus, E.; Hubinger, E.; Kernion, J.; Lukosiute, K.; Cheng, N.; Joseph, N.; Schiefer, N.; Rausch, O.; McCandlish, S.; Showk, S. E.; Lanham, T.; Maxwell, T.; Chandrasekaran, V.; Hatfield-Dodds, Z.; Kaplan, J.; Brauner, J.; Bowman, S. R.; and Perez, E. 2023. Question Decomposition Improves the Faithfulness of Model-Generated Reasoning. *CoRR*, abs/2307.11768.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Stechly, K.; Valmeekam, K.; and Kambhampati, S. 2024. On the Self-Verification Limitations of Large Language Models on Reasoning and Planning Tasks. *ArXiv*, abs/2402.08115.

Wang, P.; Li, L.; Shao, Z.; Xu, R.; Dai, D.; Li, Y.; Chen, D.; Y.Wu; and Sui, Z. 2023. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. *ArXiv*, abs/2312.08935.

Wavering, M. J. 1989. Logical reasoning necessary to make line graphs. *Journal of Research in Science Teaching*, 26(5): 373–379.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Yang, Q. A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Dong, G.; Wei, H.; Lin, H.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Lin, J.; Dang, K.; Lu, K.; Bao, K.; Yang, K.; Yu, L.; Li, M.; Xue, M.; Zhang, P.; Zhu, Q.; Men, R.; Lin, R.; Li, T.; Xia, T.; Ren, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.-C.; Wan, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; Qiu, Z.; Quan, S.; and Wang, Z. 2024. Qwen2.5 Technical Report. *ArXiv*, abs/2412.15115.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *ArXiv*, abs/2305.10601.

Zhang, Z.; Zheng, C.; Wu, Y.; Zhang, B.; Lin, R.; Yu, B.; Liu, D.; Zhou, J.; and Lin, J. 2025. The Lessons of Developing Process Reward Models in Mathematical Reasoning. *ArXiv*, abs/2501.07301.

Zheng, C.; Zhang, Z.; Zhang, B.; Lin, R.; Lu, K.; Yu, B.; Liu, D.; Zhou, J.; and Lin, J. 2024. ProcessBench: Identifying Process Errors in Mathematical Reasoning. *ArXiv*, abs/2412.06559.

Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E. P.; Zhang, H.; Gonzalez, J. E.; and Stoica, I. 2023. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *ArXiv*, abs/2306.05685.