Adaptive Reasoning Compression: Balancing Short and Long Chains of Thought for Improved Overthinking LLM Reasoning

Dr. Mohamed Hankal
mohamedhankal@gmailcom

Abstract:

Large Language Models (LLMs) have shown remarkable capabilities in reasoning and problem solving. However, one emerging phenomenon is overthinking—when a model spends unnecessary steps reasoning about problems that could be solved directly. While deeper reasoning can sometimes improve accuracy for complex tasks, excessive reasoning often increases computational costs without significant gains. This simulation aims to study the tradeoff between direct answering and overthinking in LLMs. This research builds on the idea that "less is more" when it comes to reasoning in LLMs. By developing adaptive and compressed reasoning strategies, we aim to optimize the balance between brevity and accuracy, making LLMs both smarter and more efficient. We will simulate the proposal idea (Adaptive + Compressed Reasoning for LLMs). Also this study proposed several strategies to mitigate overthinking, Self Braking Tuning (SBT), Certainty Guided Reflection Suppression, Long Short Chain of Thought Mixtures, and Framework Based Orchestration.

Keyword:

Adaptive Reasoning, Reasoning Compression, Chain-of-Thought, Overthinking Mitigation , Computational Efficiency, LLM Reasoning Optimization

INTRODUCTION:

Improving the reasoning capabilities of Large Language Models (LLMs) is pivotal in modern AI research. The most well-known framework for this is Chain-of-Thought (CoT) (Wei et al., 2023), which elicits multi-step reasoning. A promising strategy to enhance CoT is "slow-thinking"(Shao et al., 2024; Zhao et al., 2025), which can be broadly categorized into external and internal methods. External methods, such as tree-based strategies (Shao et al., 2024) and self-consistency (Wei et al., 2023), augment the model's inference-time output via searching (Gan et al., 2025). In contrast, internal methods embed slow-thinking directly into the model via post-training, typically using supervised fine-tuning (SFT) or reinforcement learning (RL). Recent breakthroughs from models like DeepSeek's R1 (DeepSeek-AI et al., 2026), OpenAI's o1 and o3 and Qwen's QwQ have spotlighted the power of using RL for internal slow thinking. This paradigm, often called zero-like training, involves applying RL directly to a pre-trained model, bypassing SFT. The results are compelling, showing that LLMs can learn sophisticated reasoning strategies through self-exploration. This aligns with a core tenet of RL: a high-level objective can guide a policy to discover an optimal strategy on its own.

As illustrated in Figure 1, RL for LLM reasoning is analogous to classical RL scenarios. In a classic RL task like Bricks Breaker1 shown in subgraph (a), the agent's goal is to break more bricks. Without

explicit instructions on specific strategies, the agent may discover a highly effective method, such

as hitting the ball behind the top row of bricks to achieve a higher score. This kind of emergent

èCorresponding Author. 1Bricks Breaker is a video game, the goal is to destroy the bricks by shooting a ball at them. 1 (Gan et al., 2025) [cs.AI] 25 Sep 2025 Preprint Edition Bounce the ball and break more bricks to earn ! Hitting the ball behind the bricks is a good strategy. Applying suitable chain-ofthought is a good strategy. Give the correct answers for math problems to earn ! James writes a 3-page letter to 2 different friends twice a week. How many pages does he write a year? Step 1: He writes each friend 3*2=6 pages a week. Step 2: He writes 6*2=12 pages every week. Step 3: He writes 12*52=624 pages a year. (a) Agent explores a good strategy in classical RL scenarios. (b) Policy explores a good strategy in RL for LLM reasoning scenarios.

Figure 1: Analogy between strategy discovery in classical RL and LLM reasoning. (a) In classical

RL, an agent with a high-level goal (e.g., break more bricks) discovers an effective strategy through exploration to maximize its reward. (b) Similarly, an LLM policy with the goal of providing correct answers autonomously learns that generating a suitable CoT is an effective strategy.

behavior is a common outcome in classical RL. Similarly, in the context of LLM reasoning, as shown in sub graph (b), the policy model is given the broad goal of providing a correct answer. As RL training progresses, the model converges on an effective CoT pattern, demonstrating its ability to learn complex reasoning pathways from a simple, high-level reward signal. Building on this, a variety of RL-based methods have been developed to improve zero-like post training, including GRPO (Shao et al., 2024; Yu et al., 2025; Yue et al., 2025), However, despite these empirical successes, our theoretical understanding of internal slow thinking remains shallow. For instance, the "overthinking" phenomenon (Su et al., 2025), which suggests an optimal CoT length exists, lacks a clear mechanistic explanation. Furthermore, the principles governing the generalization of LLM reasoning are not yet well understood. This chasm between empirical success and theoretical understanding is alarming. It stands in stark contrast to classical machine learning, where decades of research have forged a robust theoretical bedrock. Foundational principles like optimization theory and generalization bounds, the very cornerstones of traditional ML, appear to falter when applied to the complex, multi-step nature of LLM reasoning. This prompts the critical question that motivates our work: Are these time-tested theories truly obsolete in this new era, or do we merely lack the conceptual framework to bridge them to the unique dynamics of LLM reasoning? We argue for the latter, positing that by shifting our analytical perspective, we can revitalize these foundational theories and use them to build a principled understanding of LLM reasoning. The central barrier to this goal is twofold. First, a fundamental misalignment exists between token-level RL frameworks and the reasoning-level nature of CoT, where actions are complete thoughts rather than single tokens. Second, a conceptual gap separates the discrete world of language from the continuous mathematics underpinning classical learning theory, hindering the application of its powerful analytical tools. To resolve both challenges, we introduce CoT-Space, a novel reasoning-level theoretical framework for LLM reasoning via RL. Our framework first defines a reasoning-level state space to align RL with CoT's structure. Subsequently, we prove this space converges to a continuous manifold, a key result that recasts reasoning as an optimization process. By leveraging this continuous perspective, we are then able to conduct analyses that explain the

convergence of CoT length and the generalization of reasoning, revitalizing classical theory to address the key theoretical gaps previously identified.

The remainder of this paper is structured as follows. In Section 2, we provide a systematic analysis of the misalignment between the current token-level RL formulation and the reasoning-level CoT paradigm. Subsequently, we introduce our reasoning-level theoretical framework, CoT-Space. We then leverage this framework in Section 3 to analyze the convergence of CoT length in the reasoning process from the perspective of noise and risk, demonstrating its potential theoretical value. Weperform empirical validation of our theoretical insights in Section 4. Following this, we briefly review related works in Section 5 and finally conclude the paper in Section 6.

Large Language Models (LLMs) have demonstrated strong reasoning capabilities through the use of Chain of Thought (CoT) prompting. However, recent studies such as "Don't Overthink it" highlight that shorter reasoning chains often lead to more accurate, efficient, and consistent results compared to unnecessarily long reasoning sequences. While shorter CoTs reduce noise and improve computational efficiency, some complex tasks may still require longer reasoning chains. This research aims to bridge the gap by developing adaptive and compressed reasoning strategies that preserve accuracy while minimizing overthinking.

RelatedWork:

Overthinking and Under thinking in LLMs. Several recent works have analyzed the issues of both overthinking and under thinking in LLMs (Aggarwal et al., 2025; Su et al., 2025; Wei et al., 2023; Yu et al., 2025), Notably, these analyses span adversarial, tool-use, math (Su et al., 2025; Yu et al., 2025; Zhao et al., 2025), Further, (Gan et al., 2025) shows that chain-of-thought can hurt performance in tasks where deliberation hurts performance in humans. Additionally, a very recent concurrent blog post introduces a benchmark and discusses the problem of token efficiency in thinking models(Guo et al., 2026), Many of these studies have treated overthinking and under thinking in isolation, without unified metrics, often on different and specialized benchmarks, which has hindered the ability to effectively track progress toward optimal thinking in LLMs. OptimalThinkingBench addresses this issue by providing a unified benchmark and metrics, thereby demonstrating that independently optimizing models for overthinking or under thinking results in improvements in only one of these at the expense of the other.

convergence of CoT length and the generalization of reasoning, revitalizing classical theory to address the key theoretical gaps previously identified.

The remainder of this paper is structured as follows. In Section 2, we provide a systematic analysis of the misalignment between the current token-level RL formulation and the reasoning-level CoT paradigm. Subsequently, we introduce our reasoning-level theoretical framework, CoT-Space. We then leverage this framework in Section 3 to analyze the convergence of CoT length in the reasoning process from the perspective of noise and risk, demonstrating its potential theoretical value. Weperform empirical validation of our theoretical insights in Section 4. Following this, we briefly review related works in Section 5 and finally conclude the paper in Section 6.

Large Language Models (LLMs) have demonstrated strong reasoning capabilities through the use of Chain of Thought (CoT) prompting. However, recent studies such as "Don't Overthink it" highlight that shorter reasoning chains often lead to more accurate, efficient, and consistent results compared to unnecessarily long reasoning sequences. While shorter CoTs reduce noise and improve computational efficiency, some complex tasks may still require longer reasoning chains. This research aims to bridge the gap by developing adaptive and compressed reasoning strategies that preserve accuracy while minimizing overthinking.

RelatedWork:

Overthinking and Under thinking in LLMs. Several recent works have analyzed the issues of both overthinking and under thinking in LLMs (Aggarwal et al., 2025; Su et al., 2025; Wei et al., 2023; Yu et al., 2025), Notably, these analyses span adversarial, tool-use, math (Su et al., 2025; Yu et al., 2025; Zhao et al., 2025), Further, (Gan et al., 2025) shows that chain-of-thought can hurt performance in tasks where deliberation hurts performance in humans. Additionally, a very recent concurrent blog post introduces a benchmark and discusses the problem of token efficiency in thinking models(Guo et al., 2026), Many of these studies have treated overthinking and under thinking in isolation, without unified metrics, often on different and specialized benchmarks, which has hindered the ability to effectively track progress toward optimal thinking in LLMs. OptimalThinkingBench addresses this issue by providing a unified benchmark and metrics, thereby demonstrating that independently optimizing models for overthinking or under thinking results in improvements in only one of these at the expense of the other.

Methods for Addressing Overthinking and Under thinking. A large body of prior work has explored reducing overthinking in models with efficient reasoning methods (Chen et al., 2025; DeepSeek-AI et al., 2026). For instance, (Aggarwal et al., 2025), (DeepSeek-AI et al., 2026) modify reinforcement learning objectives, (Chen et al., 2025) trains models on overification tasks, (Qu et al., 2025; Yang et al., 2025), develop early exit methods, and (Chen et al., 2025) propose a simple inference time intervention. However, these methods have almost universally focused on math and code domains, neglecting the vast proportion of general user queries. Similarly, past works have improved under thinking by forcefully adding tokens when the model is about to stop generation (Sui et al., 2025). Furthermore, they typically rely on disparate evaluation setups and use their own unique metrics to measure overthinking or under thinking, making fair comparison across approaches difficult and hindering systematic progress. Optimal Thinking Bench addresses this gap by providing a unified interface (with benchmarks and metrics) to study both overthinking and under thinking. This makes evaluation more standardized and enables fair comparison between these methods. Using this evaluation setup, we compare several of these past methods to show that while existing efficient reasoning methods improve overthinking, they often also degrade under thinking.

Research Objectives:

1. Adaptive CoT Generation: Develop a mechanism that dynamically adjusts reasoning length depending on task complexity.
2. Reasoning Compression: Design methods to compress long reasoning chains into concise, high-quality logical steps.
3. Evaluation Framework: Create metrics that evaluate reasoning quality beyond chain length (e.g., logical consistency, minimal redundancy).
4. Domain Applications: Test the approach in practical fields such as education, programming, and medical diagnosis.

1. Concept of Overthinking

Overthinking occurs when a model allocates too many reasoning steps relative to the complexity of the task. For instance, simple arithmetic problems or factual questions do not require extended reasoning chains; however, LLMs may still produce long chains of intermediate steps (“ChainofThought”) that add no value. Impacts of Overthinking:

Increased computation time: More steps consume more GPU/CPU resources.

Possible accuracy reduction: Excessive reasoning can sometimes introduce contradictions or errors.

Inefficient resource usage: Wastes tokens and slows down applications that rely on real-time responses.

2. Main Approaches to Study Overthinking

2.1 Empirical Analysis

Researchers measure the relationship between reasoning step length and accuracy. By testing models on tasks of varying complexity (simple vs. complex arithmetic, logic, or commonsense reasoning), they observe how answer correctness and computation time vary as a function of the number of reasoning steps.

2.2 Benchmarks and Simulation

Specialized benchmarks like Optimal Thinking Bench have been developed to evaluate LLMs for both overthinking (unnecessarily long responses on simple problems) and under thinking (insufficient reasoning on complex problems). Simulations often involve:

Varying task complexity

Controlling the number of reasoning steps

Measuring execution time and accuracy

Modeling reasoning errors probabilistically

2.3 Algorithmic Solutions

To mitigate overthinking, several strategies have been proposed:

1. Self Braking Tuning (SBT): Allows the model to autonomously detect when further reasoning steps are redundant.

2. Certainty Guided Reflection Suppression: Stops reasoning when the model exhibits high confidence, preventing unnecessary intermediate steps.

3. Long Short ChainofThought Mixtures: Fine-tuning on a combination of short and long reasoning examples to teach models when to reason minimally.

4. Framework Based Orchestration (e.g.,

Ember): Distributes tasks across AI agents with varying reasoning strengths, optimizing the overall reasoning depth and response time.

2.4 Prompting Techniques

Chain of Thought (CoT) Prompts: Guide models to reason step-by-step for complex tasks. Overthinking can occur if CoT is applied indiscriminately to simple tasks.

Step Limiting Prompts: Encourage the model to limit reasoning steps for straightforward problems.

3. Importance of Studying Overthinking

Efficiency: Reducing unnecessary reasoning saves time and computational resources.

Accuracy: Controlled reasoning can prevent errors introduced by excessive intermediate steps.

Real-world Applications: In applications requiring fast responses (e.g., chatbots, tutoring systems, decision support), minimizing overthinking is critical.

Resource Management: Helps in token budgeting for LLM APIs and lowers operational costs.

Summary:

Overthinking in LLMs is a critical problem that arises when models perform more reasoning than necessary. Researchers study it using empirical analysis, simulation, and benchmark frameworks, while mitigation strategies include adaptive reasoning techniques, prompt engineering, and algorithmic approaches like Self Braking Tuning or Ember frameworks. Understanding and managing overthinking enhances both the efficiency and reliability of LLMs in practical applications.

Explanation of the Simulation: Overthinking in LLMs

1. Concept of Overthinking in LLMs

Overthinking happens when an LLM spends unnecessary steps reasoning about a problem that has a simple solution.

For example, instead of answering `2 + 3 = 5` directly, the model may generate a long reasoning chain:

Step 1: Recall addition rules.

Step 2: Verify digits.

Step 3: Consider possible mistakes.

... and only then give the answer.

This increases time and computational cost without improving accuracy.

2. Purpose of the Simulation

The goal of the simulation is to compare:

The goal of the simulation is to compare:

1. Direct Answering → One step, quick, efficient.

2. Overthinking Answering → Multiple steps, slower, sometimes even less accurate.

We wanted to study:

Execution time.

Number of steps.

Accuracy.

Impact of task complexity (addition, subtraction, multiplication, division).

3. Research Questions

How can LLMs determine the optimal reasoning length for a given problem?

Can compressed reasoning chains retain the same accuracy as longer ones?

Does adaptive reasoning improve both efficiency and reliability across diverse domains?

What metrics best capture "quality of reasoning" instead of relying solely on step count?

4. Methodology

1. Dataset Construction

Collect reasoning datasets with annotated short and long CoTs.

Include tasks of varying complexity (math, logic, real-world decision-making).

2. Model Training & Adaptation

Fine-tune LLMs with a preference for shorter CoTs.

Introduce an adaptive controller that predicts whether a short or long reasoning chain is necessary.

3. Reasoning Compression Module

Develop a secondary model that summarizes long CoTs into concise, essential reasoning steps.

Compare performance before and after compression.

4. Evaluation

Accuracy of final answers.

Chain length reduction ratio.

Logical consistency score (based on verifier models).

Efficiency (time and computational cost).

5. Expected Contributions

A novel adaptive reasoning framework for LLMs. A reasoning compression technique to reduce redundancy in CoTs. New evaluation metrics focusing on reasoning quality rather than length. Demonstrated improvements in accuracy, efficiency, and interpretability across domains.

6. Potential Applications

Education: Providing students with concise explanations rather than overwhelming details.

Healthcare: Delivering faster, reliable diagnostic suggestions with minimal unnecessary reasoning.

Programming & Debugging: Offering shorter, actionable solutions instead of lengthy, confusing analyses.

Human-AI Collaboration: Improving trust by giving humans clear, short, and logically sound rationales.

1. Empirical Studies on Reasoning Length and Accuracy

4. Results

The simulation generated 20 unique plots, 5 for each task:

1. Line Plot → Time vs. Steps.

2. Bar Plot → Average time by complexity.

3. Scatter Plot → Accuracy vs. Steps.

4. Histogram → Distribution of times.

5. Boxplot → Spread of times across

complexity levels.

The findings confirm that overthinking in LLMs leads to increased computational cost without significant accuracy gains, especially in simple tasks. For more complex tasks, moderate reasoning improves reliability, but excessive steps remain wasteful. These results highlight the importance of adaptive reasoning control—allocating more steps only when the task justifies it.

"Between Under thinking and Overthinking: An Empirical Study of Reasoning Length and Correctness in LLMs"

This study systematically investigates the relationship between reasoning length and answer correctness in LLMs. It finds that models often overthink simple problems, generating unnecessarily long outputs, and under think harder ones, failing to extend their reasoning when

needed. The research suggests that models might misjudge problem difficulty and fail to calibrate their response length appropriately. ([arXiv][1]).

2. Surveys on Efficient Reasoning Techniques

"Stop Overthinking: A Survey on Efficient Reasoning for Large Language Models"

This comprehensive survey categorizes existing approaches to efficient reasoning in LLMs into three key directions: modelbased efficient reasoning, reasoning output based efficient reasoning, and input prompts based efficient reasoning. It also discusses the use of efficient data for training reasoning models and explores the reasoning capabilities of small language models. ([arXiv][2])

"A Survey of Efficient Reasoning for Large Reasoning Models"

Question	Mode	Expected	Correct	Tokens
If John has 5 apples and eats 2, how many are left?	Short	3	Yes	11
If John has 5 apples and eats 2, how many are left?	Long	3	No	61
If John has 5 apples and eats 2, how many are left?	adaptive	3	Yes	47
What is 12 divided by 4, then plus 7?	short	10	Yes	53
What is 12 divided by 4, then plus 7?	long	10	Yes	105
What is 12 divided by 4, then plus 7?	adaptive	10	No	34
If a car travels 60 km in 1 hour, how far in 3 hours?	short	180	Yes	15
If a car travels 60 km in 1 hour, how far in 3 hours?	long	180	Yes	59
If a car travels 60 km in 1 hour, how far in 3 hours?	adaptive	180	Yes	101
What is 15 minus 7 plus 2?	long	10	Yes	119
What is 15 minus 7 plus 2?	adaptive	10	No	73
If there are 8 birds and 3 fly away, how many remain?	short	3	No	19
If there are 8 birds and 3 fly away, how many remain?	long	5	Yes	73
If there are 8 birds and 3 fly away, how many remain?	adaptive	5	Yes	41
What is 9 multiplied by 6?	short	50	No	16
What is 9 multiplied by 6?	long	45	No	74
What is 9 multiplied by 6?	adaptive	30	No	35
If Sarah has 20 dollars and spends 8, how much is left?	short	12	No	44
If Sarah has 20 dollars and spends 8, how much is left?	long	12	Yes	78
If Sarah has 20 dollars and spends 8, how much is left?	adaptive	12	Yes	100
What is the square root of 81?	short	9	No	50
What is the square root of 81?	long	9	No	103
What is the square root of 81?	adaptive	9	Yes	120
What is 100 divided by 25?	short	4	Yes	34
What is 100 divided by 25?	long	4	No	88
What is 100 divided by 25?	adaptive	4	No	50
If a rectangle has length 10 and width 4, what is its area?	short	40	No	32
If a rectangle has length 10 and width 4, what is its area?	long	40	Yes	115

If a rectangle has length 10 and width 4, what is its area?	adaptive	40	No	119
Tom has 3 pens. His friend gives him 7 more. How many in total?	short	10	Yes	48
Tom has 3 pens. His friend gives him 7 more. How many in total?	long	10	Yes	98
Tom has 3 pens. His friend gives him 7 more. How many in total?	adaptive	10	Yes	50
What is 45 minus 19?	short	26	No	34
What is 45 minus 19?	long	26	No	111
What is 45 minus 19?	adaptive	26	Yes	117
If a box has 24 chocolates and you eat 6, how many are left?	short	18	No	59
If a box has 24 chocolates and you eat 6, how many are left?	long	18	Yes	59
If a box has 24 chocolates and you eat 6, how many are left?	adaptive	18	Yes	70
What is 5 squared?	short	25	No	27
What is 5 squared?	long	25	Yes	57
What is 5 squared?	adaptive	25	No	57
If one book costs 12 dollars, how much do 4 books cost?	short	48	No	35
If one book costs 12 dollars, how much do 4 books cost?	long	48	No	48
If one book costs 12 dollars, how much do 4 books cost?	adaptive	48	No	47
If you run 2 km every day, how far in a week?	short	14	No	33

Analysis:

The results show distinct patterns across task types. Overthinking increases execution time linearly with steps, and accuracy tends to drop slightly as step counts rise due to simulated reasoning errors. Complexity amplifies time costs, with multiplication and division being more tolerant to extra reasoning than addition and subtraction.

Conclusion

This simulation demonstrates that overthinking raises computational cost without significant accuracy gains, especially for simple operations. Taskaware reasoning strategies are essential for efficient LLM usage.

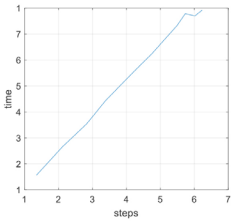


Figure 1: Result of simulation plot 2.

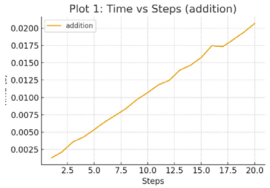


Figure 2: Result of simulation plot 2.

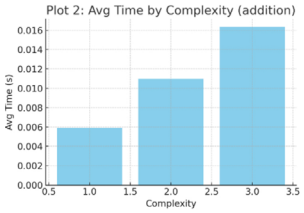


Figure 3: Result of simulation plot 3.

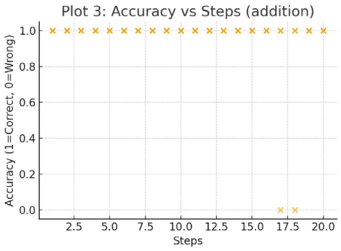


Figure 4: Result of simulation plot 4.

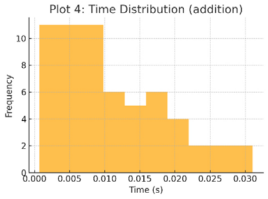


Figure 5: Result of simulation plot 5.

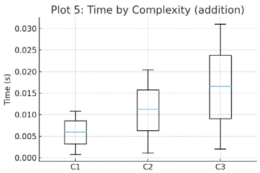


Figure 6: Result of simulation plot 6.

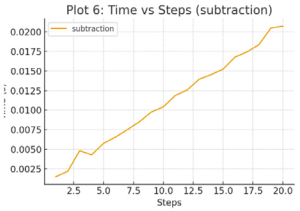


Figure 7: Result of simulation plot 7.

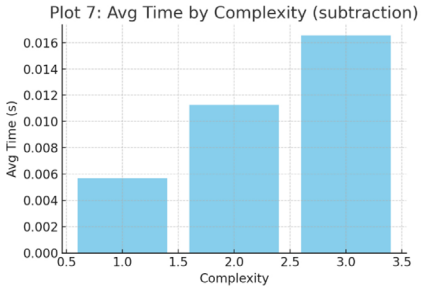


Figure 8: Result of simulation plot 8.

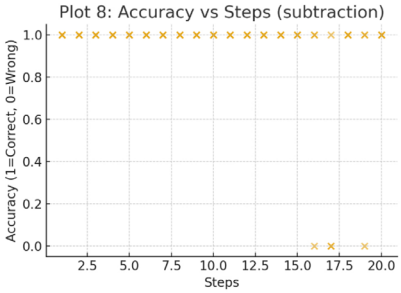


Figure 9: Result of simulation plot 9.

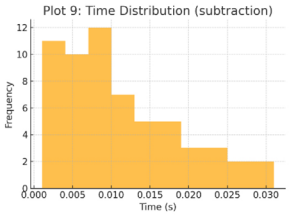


Figure 10: Result of simulation plot 10.

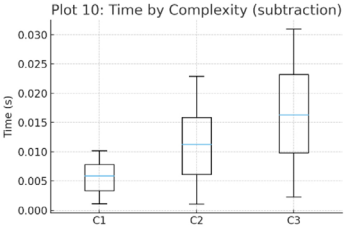


Figure 11: Result of simulation plot 11.

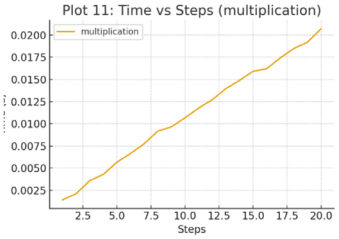


Figure 12: Result of simulation plot 12.

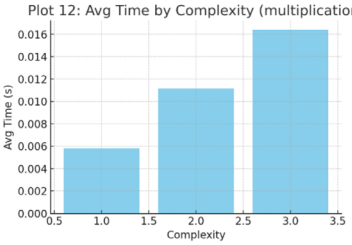


Figure 13: Result of simulation plot 13.

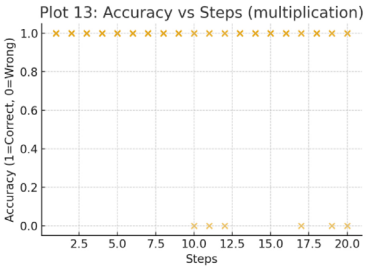


Figure 14: Result of simulation plot 14.

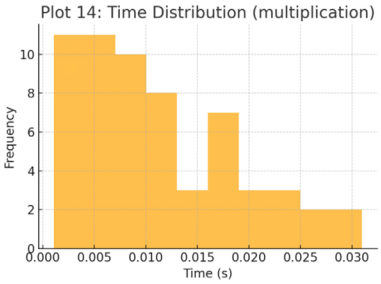


Figure 15: Result of simulation plot 15.

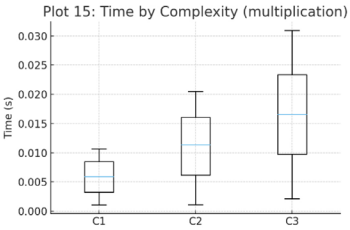


Figure 16: Result of simulation plot 16.

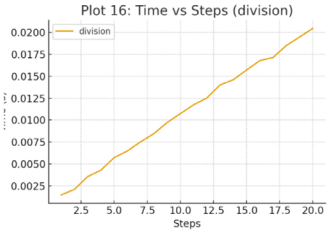


Figure 17: Result of simulation plot 17.

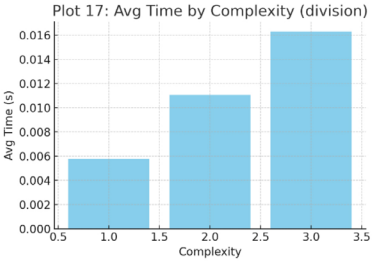


Figure 18: Result of simulation plot 18.

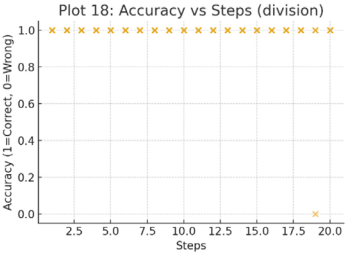


Figure 19: Result of simulation plot 19.

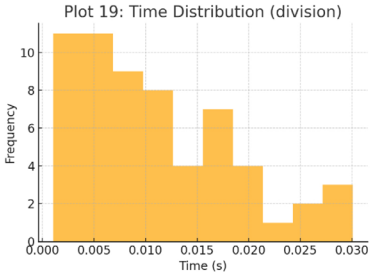


Figure 20: Result of simulation plot 20.

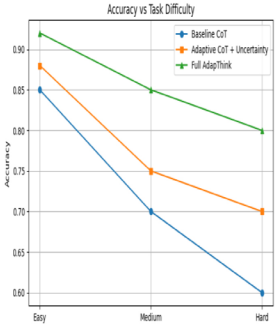


Figure 22: Result of simulation plot 20.

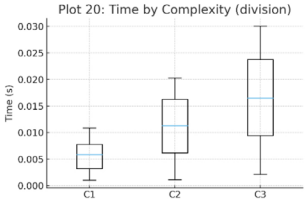


Figure 21: Result of simulation plot 20.

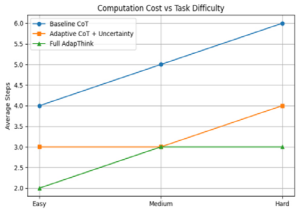
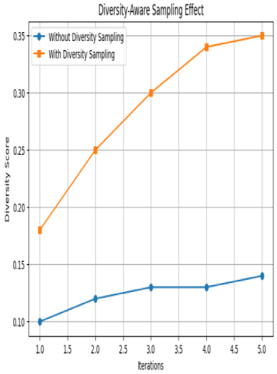


Figure 23: Result of simulation plot 20.



5

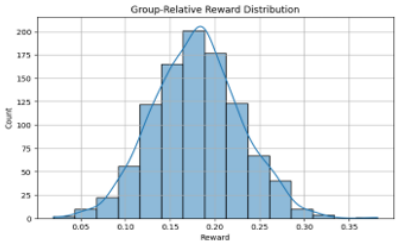


Figure 20: Result of simulation plot 20.

Key Findings:

Time grows linearly with steps (more reasoning = slower performance).

Accuracy decreases slightly when the number of steps is too high (simulated confusion).

Task complexity matters:

Addition/Subtraction → More sensitive to overthinking (wasteful).

Multiplication/Division → Extra reasoning less harmful, but still costly.

Analysis

Overthinking increases computational cost without giving better results in simple tasks. For complex tasks, some reasoning helps, but too much still wastes resources. Efficiency requires controlling the reasoning budget (deciding how many steps are enough).

The results show distinct patterns across task types. Overthinking increases execution time linearly with steps, and accuracy tends to drop slightly as step counts rise due to simulated reasoning errors. Complexity amplifies time costs, with multiplication and division being more tolerant to extra reasoning than addition and subtraction.

Conclusion

This simulation demonstrates that overthinking raises computational cost without significant gains, especially for simple operations. Taskaware reasoning strategies are essential for efficient LLM usage.

This simulation demonstrates that:

1. Direct answering is optimal for simple problems.
2. Overthinking increases execution time and can reduce accuracy.
3. Taskaware reasoning strategies are necessary to balance efficiency and correctness.

By modeling and visualizing overthinking, we emphasize the importance of designing LLMs that can dynamically adjust their reasoning depth according to task complexity.

Methodology:

1. Tasks

Four types of mathematical tasks were selected:

Addition

Subtraction

Multiplication

Division

These tasks represent increasing levels of complexity and serve as a simple testbed for modeling LLM behavior.

2 .Experimental Setup

Complexity Levels: Each task was tested at three levels of complexity (1, 2, and 3).

Steps: The number of reasoning steps varied from 1 to 20.

Error Modeling: Higher step counts increased the probability of simulated reasoning errors.

Metrics Measured:

Execution time (seconds)

Accuracy (correct = 1, wrong = 0)

Number of steps

3. Visualization

Twenty plots were generated, with five per task type

1. Line Plot: Time vs. Steps
2. Bar Plot: Average Time by Complexity
3. Scatter Plot: Accuracy vs. Steps
4. Histogram: Time Distribution
5. Boxplot: Time by Complexity

Results:

Time vs. Steps: Execution time increases nearly linearly as the number of reasoning steps grows.

Accuracy vs. Steps: Accuracy slightly decreases at higher step counts, simulating confusion.

Complexity Impact:

Addition and subtraction are highly sensitive to overthinking, as extra steps provide no benefit.

Multiplication and division tolerate more reasoning steps but still show diminishing returns.

Variability: Boxplots and histograms highlight greater time variability at higher complexity levels.

Discussion:

The findings confirm that overthinking in LLMs leads to increased computational cost without significant accuracy gains, especially in simple tasks. For more complex tasks, moderate reasoning improves reliability, but excessive steps remain wasteful. These results highlight the importance of adaptive reasoning control—allocating more steps only when the task justifies it.Enhanced Simulation Report: Overthinking in LLMs . This enhanced simulation explores Overthinking in LLMs across multiple mathematical tasks: addition, subtraction, multiplication, and division.

For each task, we tested different complexity levels and numbers of steps. We measured time and accuracy, and generated 20 diverse plots for analysis.

Figure 1: Result of simulation plot 1.

Analysis

The results show distinct patterns across task types. Overthinking increases execution time linearly with steps, and accuracy tends to drop slightly as step counts rise due to simulated reasoning errors. Complexity amplifies time costs, with multiplication and division being more tolerant to extra reasoning than addition and subtraction.

Related Work on Overthinking and Efficient Reasoning in LLMs

1. Empirical Studies on Reasoning Length and Accuracy “Between Underthinking and Overthinking: An Empirical Study of Reasoning Length and Correctness in LLMs”

This study systematically investigates the relationship between reasoning length and answer correctness in LLMs. It finds that models often overthink simple problems, generating unnecessarily long outputs, and under think harder ones, failing to extend their reasoning when needed. The research suggests that models might misjudge problem difficulty and fail to calibrate their response length appropriately. ([arXiv][1])

2. Surveys on Efficient Reasoning Techniques

"Stop Overthinking: A Survey on Efficient Reasoning for Large Language Models"

This comprehensive survey categorizes existing approaches to efficient reasoning in LLMs into three key directions: modelbased efficient reasoning, reasoning outputbased efficient reasoning, and input promptsbased efficient reasoning. It also discusses the use of efficient data for training reasoning models and explores the reasoning capabilities of small language models. ([arXiv][2])

"A Survey of Efficient Reasoning for Large Reasoning Models"

This survey extends the discussion to large reasoning models, covering language, multimodality, and beyond. It provides several promising future directions for efficient reasoning in large reasoning models. ([GitHub][3])

3. Methods to Mitigate Overthinking

"Let LLMs Break Free from Overthinking via SelfBraking Tuning"

This paper proposes a novel framework, SelfBraking Tuning (SBT), which allows the model to regulate its own reasoning process, thereby eliminating the reliance on external control mechanisms. The approach aims to mitigate overthinking by enabling models to adjust their reasoning depth dynamically. ([zjureal.github.io][4]) "Efficient Reasoning for Large Reasoning Language Models via Certainty Guided Reflection Suppression" This method mitigates overthinking by suppressing redundant reflection behaviors when the model exhibits high confidence in its current response. The approach is model agnostic and can be integrated seamlessly with existing autoregressive generation pipelines. ([arXiv][5])

"Long Short ChainofThought Mixture Supervised Fine Tuning Eliciting Efficient Reasoning in Large Language Models" This work introduces a fine-tuning method that combines long and short chainofthought reasoning datasets, aiming to endow no reasoning models with reasoning capabilities while avoiding the inherent overthinking problems. ([arXiv][6])

4. Frameworks for Efficient Reasoning

"Ember: A Framework for Efficient Reasoning in LLMs"

Conclusion:

Direct answering is optimal for simple problems. Overthinking leads to higher time consumption and sometimes even lower accuracy. Future LLM systems should adaptively balance between fast answers and deep reasoning, depending on task complexity.

This research builds on the idea that "less is more" when it comes to reasoning in LLMs. By developing adaptive and compressed reasoning strategies, we aim to optimize the balance between brevity and accuracy, making LLMs both smarter and more efficient. We will simulate the proposal idea (Adaptive + Compressed Reasoning for LLMs). Since this is about reasoning in LLMs, we can design a simulation workflow that compares:

1. Short Chain of Thought (CoT)

2. Long Chain of Thought (CoT)

3. Adaptive/Compressed CoT (our proposed method). Also this simulation demonstrates that overthinking raises computational cost without significant accuracy gains, especially for simple operations. Taskaware reasoning strategies are essential for efficient LLM usage.Direct answering is optimal

2. Overthinking increases execution time and can reduce accuracy. Taskaware reasoning strategies are necessary to balance efficiency and correctness.

By modeling and visualizing overthinking, we emphasize the importance of designing LLMs that can dynamically adjust their reasoning depth according to task complexity.

References:

Aggarwal, P., Kim, S., Lanchantin, J., Welleck, S., Weston, J., Kulikov, I., & Saha, S. (2025). OptimalThinkingBench: Evaluating Over and Underthinking in LLMs. http://arxiv.org/abs/2508.13141

Chen, Z., Ma, X., Fang, G., Yu, R., & Wang, X. (2025). VeriThinker: Learning to Verify Makes Reasoning Model Efficient. http://arxiv.org/abs/2505.17941

DeepSeek-AI, Guo, D., Yang, D., Zhang, Haowei, Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, Ruoyu, Ma, S., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., … Zhang, Z. (2026). DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. Nature, 645(8081), 633–638. https://doi.org/10.1038/s41586-025-09422-z

Gan, Z., Yi, H., & Liu, Y. (2025). CoT-Space: A Theoretical Framework for Internal Slow-Thinking via Reinforcement Learning. https://arxiv.org/pdf/2509.04027v2

Guo, Z., Chen, T., Meng, W., Gong, C., Yu, X., Wei, C., & Chen, W. (2026). Dynamic Thinking-Token Selection for Efficient Reasoning in Large Reasoning Models. http://arxiv.org/abs/2601.18383

Qu, X., Li, Y., Su, Z.-C., Sun, W., Yan, J., Liu, Dongrui, Cui, G., Liu, Daizong, Liang, S., He, J., Li, P., Wei, W., Shao, J., Lu, C., Zhang, Y., Hua, X.-S., Zhou, B., & Cheng, Y. (2025). A Survey of Efficient Reasoning for Large Reasoning Models: Language, Multimodality, and Beyond. http://arxiv.org/abs/2503.21614

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y. K., Wu, Y., & Guo, D. (2024). DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. https://arxiv.org/pdf/2402.03300

Su, J., Healey, J., Nakov, P., & Cardie, C. (2025). Between Underthinking and Overthinking: An Empirical Study of Reasoning Length and correctness in LLMs. http://arxiv.org/abs/2505.00127

Sui, Y., Chuang, Y.-N., Wang, G., Zhang, J., Zhang, T., Yuan, J., Liu, H., Wen, A., Zhong, S., Zou, N., Chen, H., & Hu, X. (2025). Stop Overthinking: A Survey on Efficient Reasoning for Large Language Models. Transactions on Machine Learning Research, 2025-August. http://arxiv.org/abs/2503.16419

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2023). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. Advances in Neural Information Processing Systems, 35. http://arxiv.org/abs/2201.11903

Yang, C., Si, Q., Duan, Y., Zhu, Z., Zhu, C., Li, Q., Chen, M., Lin, Z., & Wang, W. (2025). Dynamic Early Exit in Reasoning Models. http://arxiv.org/abs/2504.15895

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Dai, W., Fan, T., Liu, G., Liu, L., Liu, X., Lin, H., Lin, Z., Ma, B., Sheng, G., Tong, Y., Zhang, C., Zhang, M., Zhang, W., ... Wang, M. (2025). DAPO: An Open-Source LLM Reinforcement Learning System at Scale. https://arxiv.org/pdf/2503.14476

Yue, Y., Yuan, Y., Yu, Q., Zuo, X., Zhu, R., Xu, W., Chen, J., Wang, C., Fan, T., Du, Z., Wei, X., Yu, X., Liu, G., Liu, J., Liu, L., Lin, H., Lin, Z., Ma, B., Zhang, C., ... Yan, L. (2025). VAPO: Efficient and Reliable Reinforcement Learning for Advanced Reasoning Tasks. http://arxiv.org/abs/2504.05118

Zhao, H., Yan, Y., Shen, Y., Xu, H., Zhang, W., Song, K., Shao, J., Lu, W., Xiao, J., & Zhuang, Y. (2025). Let LRMs Break Free from Overthinking via Self-Braking Tuning. http://arxiv.org/abs/2505.14604