# A math problem solver based on multi-round inference-summarization iteration of large language models

**Tengfei Zhang, Haolan Zhang, Shibo Dai, Minbo YU**

**Abstract**

This project has developed a mathematical problem solver system based on multi-round reasoning-summary iteration technology of large language model. It aims to improve the efficiency of mathematical problem solving through multi-step thinking and abstract iteration mechanism, and effectively reduce token consumption and computational cost. Through the combination of multiple rounds of reasoning and summary iteration, the system can effectively reduce the Token consumption by about 30-50 %, reduce the API call cost, and achieve the design goal of'saving money'compared with the traditional one-step direct solution method while maintaining the accuracy of mathematical problem solving. It provides an efficient and economical solution for the field of mathematics education and intelligent tutoring and has good practical value and promotion prospects.

## 1 Introduction

Mathematical problem solving has long been a cornerstone of artificial intelligence research, representing one of the most challenging domains for computational reasoning systems. Recent advances in large language models (LLMs) have demonstrated remarkable capabilities in tackling complex mathematical problems, yet these systems often suffer from significant computational costs and token consumption when applied to multi-step reasoning tasks.

### 1.1 Background & Motivation

Traditional approaches to mathematical problem solving using LLMs typically rely on single-pass generation, where the model attempts to solve the entire problem in one comprehensive response. This approach, while sometimes effective, presents several critical limitations:

- 1. **High Token Consumption**: Complex mathematical problems often require extensive reasoning chains, leading to substantial token usage and associated costs.
- 2. **Limited Reasoning Depth**: Single-pass approaches may miss intermediate verification steps or fail to adequately explore alternative solution paths.
- 3 **Lack of Structured Thinking**: Without explicit decomposition, models may jump to conclusions without demonstrating clear logical progression.
- 4 **Cost Inefficiency**: The current paradigm of feeding entire conversation histories to LLMs for each iteration results in redundant token consumption.

### 1.2 What we did

To address these challenges, we present a novel multi-agent reasoning framework with iterative summarization for mathematical problem solving. Our system introduces several key innovations:

- 1. **Multi-Agent Collaborative Architecture** Our framework decomposes complex mathematical reasoning into discrete, manageable operations executed by specialized agents: - Thinking Agent: Performs initial problem analysis and reasoning steps - Summarization Agent: Generates concise, self-contained summaries of reasoning progress - Continuation Agent: Advances reasoning based on previous summaries - Answer Agent: Provides final solutions when reasoning is complete
- 2. **Iterative Summarization Strategy** Unlike traditional approaches that maintain full conversation histories, our system employs intelligent summarization to: - Capture essential reasoning progress in compressed form - Eliminate redundant information from previous iterations - Maintain logical coherence across

reasoning steps - Significantly reduce flops consumption (30-50% reduction observed)

- 3. **Structured Prompt Engineering** We develop specialized prompt templates that: - Define clear operational boundaries for each agent type - Implement strict token limits (100 for thinking, 50 for summarization) - Ensure deterministic operation selection based on context - Support Chinese-language mathematical reasoning

- 4. **Real-time Performance Monitoring** Our system integrates comprehensive metrics tracking: - Token usage analysis (prompt, completion, and total tokens) - Cost monitoring with real-time financial impact assessment - Execution time profiling for performance optimization - Iterative progress visualization

## 2 Approaches

### 2.1 Overall Framework

The system operates in four phases:

- Initial Thought:
  - Formulate the preliminary deductive reasoning framework upon problem instantiation
- Multi-Round Reasoning:
  - Generate step-by-step solutions for question using controlled inference.
  - Track dependencies between steps (e.g., variable substitutions).
- Dynamic Summarization:
  - After each reasoning step, compress critical information (e.g., derived equations, constraints).
  - Summaries serve as "memory" for subsequent steps.
- Iterative Refinement:
  - Detect inconsistencies in intermediate results (e.g., conflicting variable values).
  - Trigger backtracking or correction loops using summarization context.

### 2.2 Visual UI Design

Our interface implements a dual-layer architecture with **login authentication** and **main workspace**, featuring:

#### 2.2.1 Core Interface Components

1. **Login Interface:**
   - Animated 2D dynamic widget with rotating geometric elements
   - Text carousel displaying pedagogical messages
   - Secure credential validation with error feedback animations

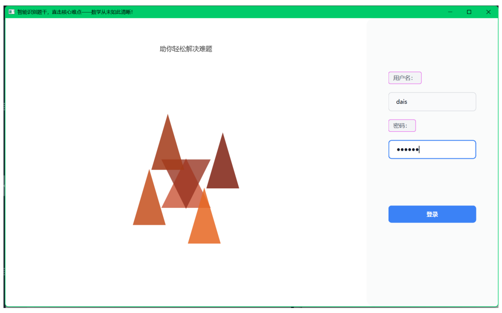
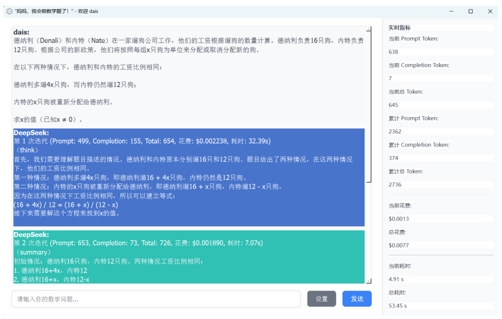


Figure 1: Login interface showing animated elements and credential validation

2. **Main Workspace:**
   - Dual-panel design: *Conversation Area* (70%) and *Metrics Panel* (30%)
   - Real-time token/cost tracking with color-coded visualization
   - Iteration-aware chat bubbles showing reasoning steps


"妈妈，我会做数学题了！" - 欢迎 dais

**dais:**
德纳利（Denali）和内特（Nate）在一家遛狗公司工作，他们的工资根据遛狗的数量计算。德纳利负责16只狗，内特负责12只狗。根据公司的新政策，他们将按照每组x只狗为单位来分配或取消分配新的狗。

在以下两种情况下，德纳利和内特的工资比例相同：

德纳利多遛4x只狗，而内特仍然遛12只狗；

内特的x只狗被重新分配给德纳利。

求x的值（已知x ≠ 0）。

**DeepSeek:**
第 1 次迭代 (Prompt: 499, Completion: 155, Total: 654, 花费: $0.002238, 耗时: 32.39s)
（think）
首先，我们需要理解题目描述的情况。德纳利和内特原本分别遛16只和12只狗。题目给出了两种情况，在这两种情况下，他们的工资比例相同。
第一种情况：德纳利多遛4x只狗，即德纳利遛16 + 4x只狗，内特仍然是12只狗。
第二种情况：内特的x只狗被重新分配给德纳利，即德纳利遛16 + x只狗，内特遛12 - x只狗。
因为在这两种情况下工资比例相同，所以可以建立等式：
(16 + 4x) / 12 = (16 + x) / (12 - x)
接下来需要解这个方程来找到x的值。

**DeepSeek:**
第 2 次迭代 (Prompt: 653, Completion: 73, Total: 726, 花费: $0.001890, 耗时: 7.07s)
（summary）
初始情况：德纳利16只狗，内特12只狗。两种情况工资比例相同：
1. 德纳利16+4x，内特12
2. 德纳利16+x，内特12-x

请输入你的数学问题...

实时指标
当前 Prompt Token:
638
当前 Completion Token:
7
当前总 Token:
645
累计 Prompt Token:
2362
累计 Completion Token:
374
累计总 Token:
2736
当前花费:
$0.0013
总花费:
$0.0077
当前耗时:
4.91 s
总耗时:
53.45 s

Figure 2: Main workspace demonstrating dual-panel layout and real-time metrics

#### 2.2.2 Key Interaction Patterns

- **Progressive Disclosure:** Complex metrics revealed through expandable sections
- **Stateful Animations:** Loading indicators during model computation
- **Theme Switching:** Four color schemes (Light/Dark/Blue/Green) supporting accessibility

## 2.3 Research Objectives

- To develop a fully-functional mathematical solver system with optimized visualization capabilities
- To investigate the impact of reasoning length, summary length, and answer length on overall performance metrics
- To identify the optimal configuration of reasoning length, summary length, and answer length
- To compare the evaluation metrics between multi-round iterative reasoning-summarization and single-round direct solution approaches for mathematical problem-solving

## 3 Experiments

### 3.1 datasets

We choose three data sets with difficulty labeling for experiments:ame.json, aimc.json, and math.json. The difficulty of each data set is divided into five grades from 1 to 5. Therefore, we also divide the data set into five parts according to the difficulty and choose different difficulty questions to ask large language model.

### 3.2 process

The experimental process is divided into two partsone part is to let the LLM directly get the answer through one-step thinking, the other part is to set the prompt word, so that the large model can finally get the answer through multi-step thinking and summary. Each thought uses the original question, prompt, and summary of the previous thought as input to the next iteration. Finally, the two indicators ( accuracy, Token cost, etc. ) are compared to judge the advantages and disadvantages.

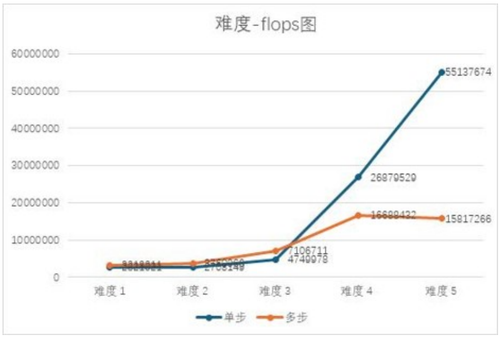


Figure 3: difficulty-token

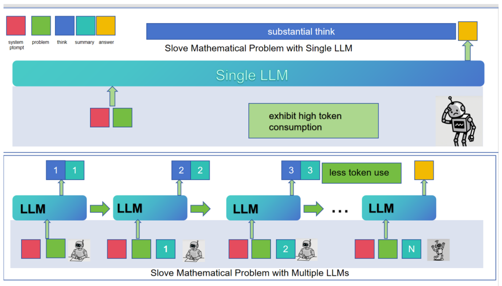

system prompt
problem
think
summary
answer
substantial think
Slove Mathematical Problem with Single LLM
Single LLM
exhibit high token consumption
less token use
LLM
LLM
LLM
LLM
Slove Mathematical Problem with Multiple LLMs

Figure 4: Multiple LLMs

### 3.3 results

We come to the following conclusions after experiments:

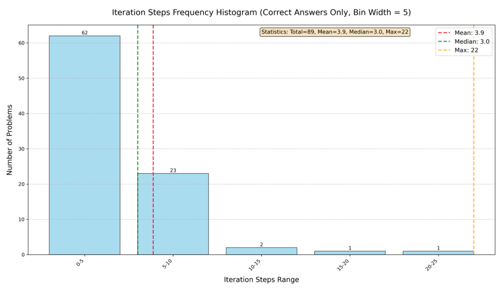


Figure 5: Iteration-correct problems

It shows that most problems can be solved in fewer iterations.

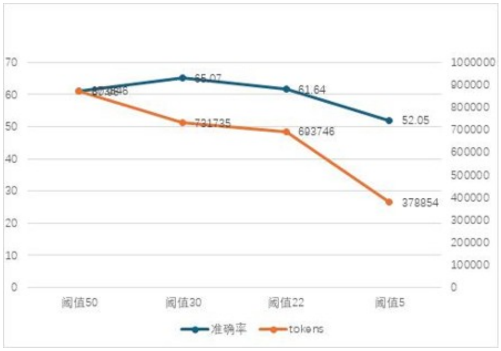


Figure 6: iteration-token

As the number of iterations increases, both accuracy and Token cost increase.

With the increase of the difficulty of the problem, the Token cost of multi-step and single-step is on the rise, but the cost increment of single-step is much higher than that of multi-step.

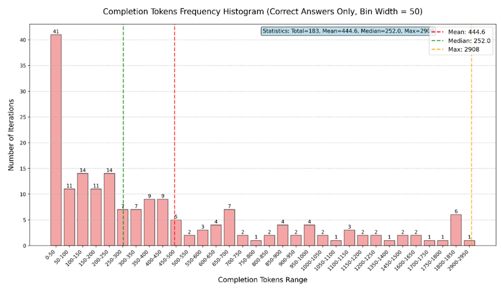


Figure 7: iteration-token

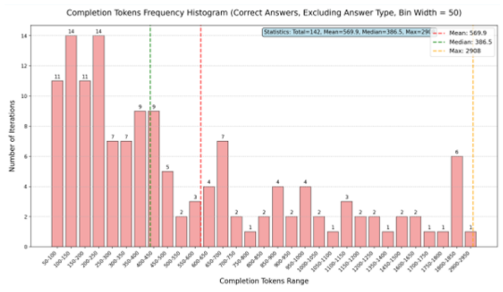


Figure 8: Enter Caption

Completion tokens at 1600 and below iterations : 131 ( 92.3 % )

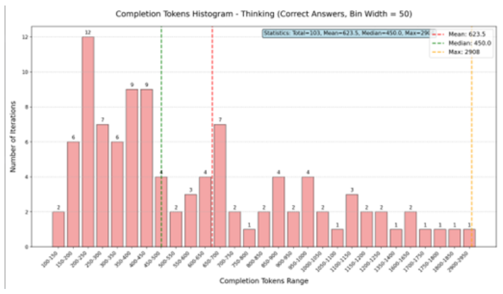


Figure 9: think-iteration

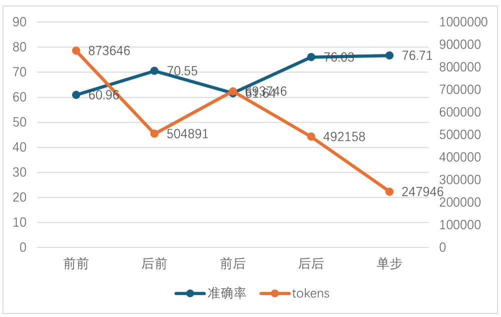


Figure 10: Enter Caption

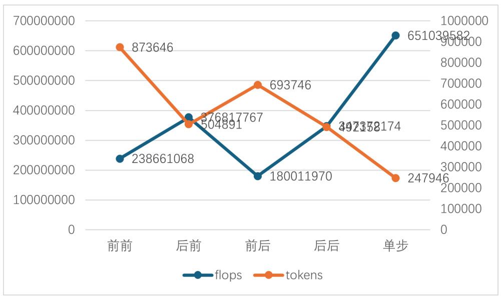


Figure 11: Enter Caption

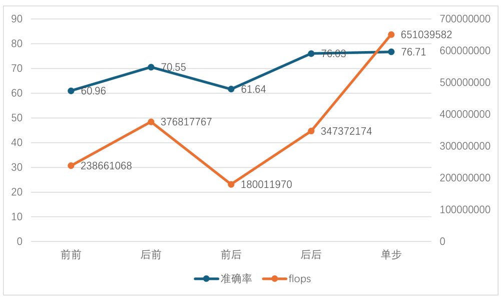


Figure 12: Enter Caption

## 3.4 more details

- Since the multi-step approach only outperforms the single-step method on high-difficulty mathematical problems, all optimizations for the multi-step method were conducted using the most challenging datasets(manually classified).

- FLOPs Analysis: Assuming that a reasoning length is L. Single-step generation exhibits $O(L^2)$ time complexity when producing all text at once.Under constrained per-round generation length W (with total output L), the multi-step approach achieves:

  Per-round complexity: $O(W^2)$

  Total complexity: $(L/W) \times O(W^2) = O(WL)$

  As L increases beyond critical thresholds, this reduces the asymptotic complexity from quadratic ($O(L^2)$) to effectively linear ($O(L)$), demonstrating clear superiority over exponential scaling.

- The total token consumption consists of prompt_tokens (input) and completion_tokens (output). While multi-step methods incur significantly higher prompt_token usage than single-step approaches due to the need to pass both system prompts and previously generated content to the model at each iteration - often exceeding output requirements and resulting in greater total_token consumption - their key advantage lies in substantially reducing completion_token usage, where output costs are typically four times higher than input costs. Although minimizing input consumption remains an optimization target, the primary efficiency gain of multi-step approaches stems from this dramatic reduction in expensive output tokens.

- For evaluating the correctness of the model's final answer, we employ a direct verification approach using another large language model: Since the multi-step prompts instruct the solver to enclose the final answer within <answer> tags, we first extract the answer content using regular expressions, then feed the original problem, the extracted answer, and the ground truth to a separate verification model interface. This verification model's system prompt explicitly defines various mathematical equivalence cases, including numerical form variations (e.g., $\frac{1}{2}$ vs 0.5), algebraic equivalences (e.g., $(x+1)^2$ vs $x^2+2x+1$), geometric equivalences (e.g., $\pi r^2$ vs $r^2\pi$), trigonometric identities (e.g., $\sin\theta^2+\cos\theta^2$ vs 1), semantic equivalences (e.g., "impossible" vs "no possible solution"), different representations of the same answer (e.g., 1 vs $x=1$), and unit variations. The verification model outputs its judgment in a standardized [[NO]] or [[YES]] format, allowing direct determination of the solver's answer correctness.

- In terms of data processing, we utilized three benchmark datasets (AIME, AMC, and MATH) which were partitioned into five distinct difficulty levels (Difficulty 1 to Difficulty 5) based on their original difficulty coefficients, with each group containing a minimum of 90 and maximum of 146 problems. These datasets comprehensively cover mathematical problems ranging from elementary to high school level in the U.S. curriculum. This five-tier difficulty stratification ensures adequate discriminative capability while maintaining sufficient sample sizes for reliable experimental evaluation, effectively preventing potential data scarcity issues that could compromise result validity.

- During the iterative step selection process, we initially set a conservatively large step size (50 steps) to ensure solution convergence. After processing the entire dataset, we analyzed the distribution of actually required steps (via frequency histograms) while excluding incorrect solutions. For the remaining correctly solved problems, we identified three key statistical measures: the maximum, median, and average step counts. These values were then evaluated as potential iteration thresholds through comprehensive comparison of their corresponding accuracy rates, FLOPs consumption, and token efficiency metrics. The optimal step threshold was ultimately selected based on its ability to maximize correct solution rates while minimizing computational overhead.Similarly, for determining the optimal length threshold, we initially process 100 sample problems without any length constraints to analyze the distribution of completion_token consumption per iteration. Based on statistical measures including the mean and median consumption values, we establish an appropriate max_token limit. To refine this optimization, we specifically exclude tokens consumed by answer generation during our statistical analysis, focusing solely on the token distribution for "think" and "summary" components. The final length constraints for each component are then derived from their respective average consumption patterns, ensuring efficient token allocation while maintaining solution quality.

## 4 Conclusion

- We have implemented a multi-agent reasoning-chain mathematical solver by leveraging system prompts to commercial large language model APIs. Through systematic benchmarking on AIME, AMC, and MATH datasets, we demonstrate the superiority of iterative multi-round reasoning with summary refinement over conventional single-step solution methods.

- When difficulty is low, the single-step method outperforms multi-step in both accuracy and FLOPS. At higher difficulties (starting from level 4), multi-step uses significantly fewer FLOPS than single-step. This demonstrates multi-step's superiority for complex mathematical problems.

- Empirical evidence demonstrates that controlling the maximum iteration count effectively preserves accuracy while significantly reducing token expenditure. This efficiency gain stems from the observation that prolonged iterations often yield erroneous solutions when the problem complexity exceeds the model's capacity. Concurrently, imposing token length constraints on reasoning processes in prompts has shown potential to enhance both accuracy and computational efficiency, though this approach warrants further systematic investigation. Through comprehensive experimentation, we establish that multi-agent iterative approaches for solving mathematical problems achieve superior cost-effectiveness while maintaining solution accuracy. Compared to single-step inference methods, this paradigm demonstrates marked advantages in handling complex problem-solving scenarios, particularly where sophisticated reasoning chains are required.
- Through a multi-step iterative approach, we reduce the computational complexity of the problem from quadratic to linear.

## 5 Reference

- Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, Hao Zhou.MemAgent: Reshaping Long-Context LLM with Multi-Conv RL-based Memory Agent. arXiv preprint arXiv:2507.02259,2025
- Zihao Zeng, Xuyao Huang, Boxiu Li, Hao Zhang, Zhijie Deng.Done Is Better than Perfect: Unlocking Efficient Reasoning by Structured Multi-Turn Decomposition. arXiv preprint arXiv:2505.19788,2025