Research on an Intelligent Tutoring System for Discrete Mathematics Integrating Knowledge Graph and LLM

Kun Luo * and Yi Kuang

School of Automation and Electronic Information, Xiangtan University, Xiangtan China
Email: lk25802025@outlook.com (K.L.); 2820525146@qq.com (Y.K.)
*Corresponding author

Manuscript received April 11, 2026; accepted June 11, 2026; published August 31, 2026.

***Abstract*—This study addresses the challenges of knowledge fragmentation and implicit computational thinking cultivation in discrete mathematics education. We propose an intelligent tutoring system that integrates knowledge graphs with large language models using a graph rag approach. The system adopts a three-tier architecture—data foundation, Knowledge Graph (KG) core, and intelligent empowerment—to bridge the gap between multi-source academic data and personalized tutoring. To explicitly support Computational Thinking (CT), the KG models the mapping between discrete mathematics concepts and CT dimensions such as abstraction and decomposition. Experimental results on domain-specific question answering tasks show that the KG-enhanced approach achieves an accuracy of 0.85 and an F1-Score of 0.87, significantly outperforming standard Retrieval-Augmented Generation (RAG) and pure Large Language Model (LLM) methods. The findings demonstrate the technical feasibility of using structured knowledge to provide high-precision, reasoning-based support, laying a foundation for the systematic cultivation of computational thinking in intelligent education.**

***Keywords*—knowledge graph, computational thinking, Large Language Model (LLM), graph rag**

I. INTRODUCTION

In the 21st century's digital era, computational thinking has become a key core competency [1]. With the rapid development of technologies such as big data and artificial intelligence, computational thinking plays a vital role in solving complex problems, designing intelligent systems, and understanding human behavior. The concept of computational thinking, proposed by Professor Jeannette Wing, emphasizes problem-solving, system design, and behavior understanding through the fundamental concepts of computer science [2]. It is considered a basic skill as essential as reading, writing, and arithmetic—a viewpoint that has gained global consensus.

Discrete mathematics [3], as a foundational course in computer science, is crucial for cultivating computational thinking. While topics like set theory and logic provide natural training grounds for abstraction and reasoning, teaching often focuses on isolated theory, leaving computational thinking implicit. This disconnect hinders students from applying knowledge to solve real problems. Current educational tools further limit learning by offering only practice exercises without diagnosing or developing deeper thinking skills. Consequently, students struggle to gain lasting, transferable problem-solving abilities.

This research proposes an intelligent tutoring system that integrates knowledge graphs with Large Language Models (LLMs) to address the challenges mentioned above [4]. The system achieves a transformation in the teaching model through the following innovations: First, we have constructed a discrete mathematics knowledge graph oriented toward computational thinking cultivation, which explicitly represents the mapping relationships between knowledge points and computational thinking components [5]. This knowledge graph not only includes traditional conceptual systems but also specially labels the computational thinking elements trained by each knowledge point, such as abstraction, decomposition, and algorithmic thinking. Second, the system utilizes the deep semantic understanding capabilities of large language models to accurately grasp students' learning needs [6]. By analyzing students' natural language questions, the system can identify their knowledge gaps and thinking deficiencies, providing personalized learning support. Most importantly, the system establishes a closed-loop mechanism of "knowledge acquisition-thinking diagnosis-targeted training." When a student shows weakness in a specific computational thinking component, the system can recommend learning materials that target the same thinking pattern but are situated in different contexts, thereby achieving precise ability training.

The primary objective of this study is to promote the in-depth development of computational thinking education through technological innovation. Specifically, our contributions are manifested at three dimensions: (1) at the theoretical level, we propose a knowledge restructuring methodology with computational thinking as the core thread; (2) at the technical level, we design an intelligent tutoring mechanism featuring the synergy of knowledge graphs and large language models; (3) at the practical level, we develop a prototype intelligent tutoring system for discrete mathematics, thereby providing a referential case for related research.

II. METHODOLOGY

A. Technical Route

The intelligent tutoring system for discrete mathematics proposed in this study adopts a rigorous layered architecture design, as illustrated in Fig. 1. Aiming to construct a comprehensive technical system driven by data as the foundation, knowledge graphs as the core, and intelligence as the empowerment. The overall architecture consists of three closely collaborative layers—the data foundation layer, the knowledge graph core layer, and the intelligent application layer—which together achieve a deep transformation process from multi-source heterogeneous raw data to intelligent pedagogical applications [7].

doi: 10.18178/ijssh.2026.16.3.1290

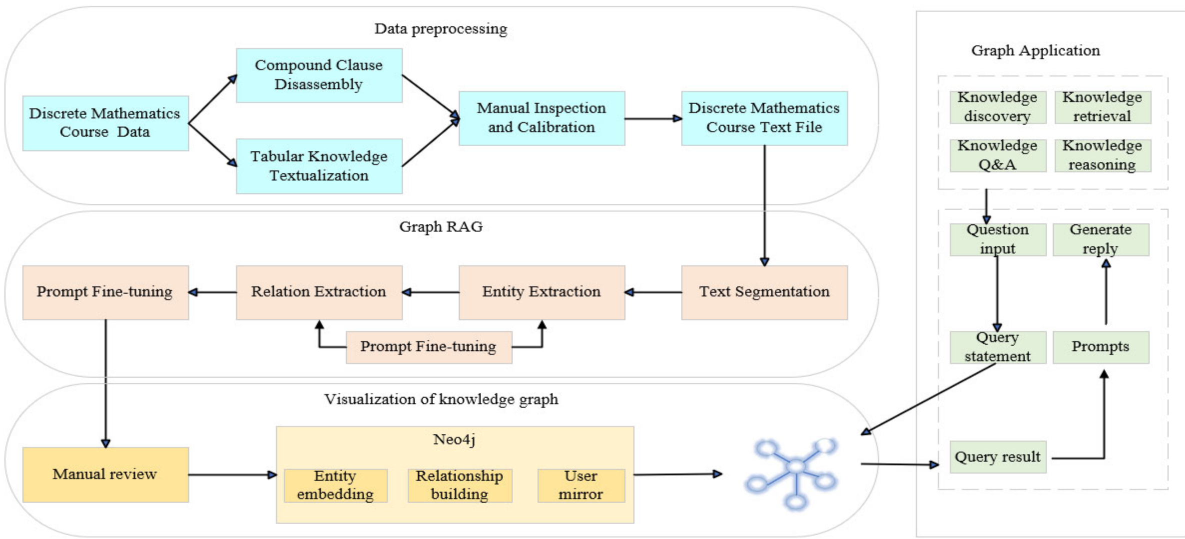

Fig. 1. Overall architecture of the intelligent tutoring system.

In the core considerations of the architectural design, the system primarily addresses the structured representation of discrete mathematics knowledge by restructuring traditional linear knowledge systems into semantic association networks, thereby overcoming the challenges of knowledge fragmentation. Simultaneously, the system is committed to establishing explicit mappings between underlying knowledge elements and high-order computational thinking components, making implicit thinking patterns such as abstraction, decomposition, and algorithmic design concrete and visible to provide logical support for subsequent instructional interventions. This route deeply integrates Graph Retrieval-Augmented Generation (Graph RAG) technology, leveraging the topological structure of the graph database to support multi-hop reasoning for complex problems, supplemented by vector embedding techniques to enhance semantic retrieval precision. By constructing a closed loop of problem perception-knowledge localization path generation-personalized feedback, the system aims to provide students with precise guidance that is not limited to answers but rich in reasoning logic, thereby providing the technical possibility for the all-dimensional cultivation of computational thinking.

B. Data Processing

As the cornerstone of knowledge graph construction, the core objective of data processing is to transform multi-source, heterogeneous, and unstructured teaching materials into high-quality, machine-understandable standardized text corpora. This system adopts the process shown in Fig. 2, with the following specific steps.

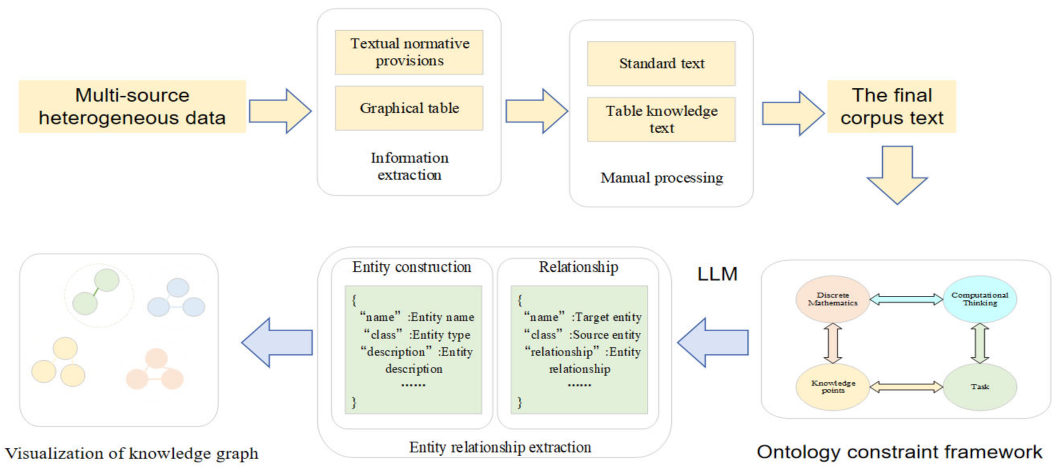

Fig. 2. Data preprocessing workflow for knowledge graph construction.

The system implements a multi-channel data collection scheme, covering paper textbooks, electronic syllabi, lecture notes, exercise sets, and unstructured resources from academic websites and open courses. During the preprocessing stage, to handle the significant differences in format and semantic density across various materials, the system applies key technologies such as compound clause decomposition, table-to-text conversion, and semantic reconstruction. These techniques ensure semantic integrity, particularly when dealing with complex logical propositions and truth tables in discrete mathematics. Given that large language models may exhibit hallucinations or biases when parsing highly specialized technical expressions, the system has established a rigorous quality control pipeline where all automatically processed texts must undergo manual review by domain experts. Through multiple rounds of iterative labeling, cross-validation, and conflict resolution algorithms based on consistency checks, the system effectively maintains the accuracy and rigor of the knowledge base. This stringent data processing workflow ensures that all corpora meet the validation standards of domain experts before entering the knowledge graph construction phase, laying a solid foundation of reliability for subsequent entity extraction and relationship fusion.

III. Knowledge Graph Construction

A. Ontology Type Definition

The construction of the knowledge graph begins with a rigorous ontology design that serves as the conceptual backbone for structuring discrete mathematics knowledge and computational thinking dimensions [8]. Unlike traditional educational ontologies that focus solely on the hierarchical nesting of concepts, this study develops a multi-dimensional framework that explicitly links domain knowledge with cognitive abilities. The selection of entity types is grounded in Professor Jeannette Wing’s foundational theory, which defines computational thinking through elements such as abstraction, decomposition, and algorithmic design. Within this ontology, Computational Thinking entities represent the high-order cognitive goals, while Knowledge Points and Chapter represent the domain-specific content of discrete mathematics. This mapping is designed to transform implicit thinking processes into explicit, traceable graph nodes, allowing the system to diagnose not only what a student knows but how they think. The relationship Cultivates is particularly critical, as it creates a semantic bridge between specific mathematical concepts, such as Propositional Logic, and CT dimensions like Systematic Reasoning. By defining Task entities as the practical application of these concepts, the ontology establishes a complete pedagogical loop where exercises are no longer isolated tests but instruments for measuring specific cognitive competencies (see Table 1).

Table 1. Definitions and examples of entity types in the ontology

Entity type	Example
Computational thinking	Formal Abstraction, Modeling Ability, Systematic Reasoning
Classification	Mathematical Logic, Set Theory, Graph Theory, Combinatorics
Chapter	Basic Concepts of Propositional Logic, Set Algebra, Matrix Representations of Graphs
Knowledge points	Propositions and Logical Connectives, Operations on Relations, Properties of Functions
Task	Write the negation of the following propositions in natural language:1. It is raining and windy today. 2. If it rains, I will not go out. 3. All people can swim.

B. Enhance the Prompt Learning Method of the Ontology

Knowledge mining is the pivotal process of extracting structured entity knowledge from natural language texts. Traditional entity mining methods, such as regular expression matching, are often restricted to specific text structures, while unsupervised learning approaches tend to produce fragmented knowledge with limited accuracy, thereby reducing the overall usability of the knowledge graph. To address the semantic diversity in discrete mathematics resources, this study proposes an LLM-based knowledge mining method consisting of two core modules: knowledge extraction and graph integration. By injecting ontological constraints directly into the prompts, the system normatively extracts entities and relationships, which are then structurally integrated through multi-dimensional semantic similarity comparisons.

In this framework, a “prompt” serves as the instructional bridge to activate specific model parameters, enhancing generalization for domain-specific tasks. Our prompt template leverages “explicit instruction”, “Chain-of-Thought (CoT)”, and “formatted input-output” to stimulate the model’s reasoning capabilities. The formalized structure of the prompt designed for this study is defined as follows [9]:

1) Instruction

Given a discrete mathematics document, first identify all entities within the ontology framework; then, based on the ontological relationships, pair and constrain all inter-entity links.

2) Ontology framework

Entity Types: Computational Thinking, Chapter, Knowledge Point, Task.

Relationships: Belongs to, Cultivates, relates to, Applies.

3) Chain-of-thought step 1: Entity construction

Identify all entities as complete events. Extract the name, category, context, and description for each, formatting them into a specified dictionary structure.

4) Chain-of-thought step 2: Relationship construction

Determine the relationships between Step 1 entities based on the relationship list. Extract the source, target, category, description, and strength into a dictionary format.

5) Output example

{“nodes”: [{“name”: “< entity _ name >,” “class”: “<entity_class>, context: <entity_context>,” description: <entity_description>},], edges: [{“source” ource_entity>, “target”: “<target_entity>,” “class”: <relation_class>, “context”: < relation _ context >,” description: “<relationship_description>,” strength”: <relationship_strength>},]}

This prompt-based extraction method focuses on mining entities from high-coverage knowledge, enabling the retrieval of complete semantic expressions [10]. By treating each entity as a complete “event,” the system effectively avoids the exponential increase in graph complexity that typically arises from repeated segmentation. Furthermore, retaining the contextual information of each entity during the mining process facilitates robust knowledge provenance tracing, ensuring that every node in the discrete mathematics knowledge graph can be verified against its original source.

C. Knowledge Graph Construction Results

The finalized discrete mathematics knowledge graph, constructed through the integrated LLM mining and manual expert verification pipeline, consists of 189 distinct entities and 345 semantic relationships is shown in the Fig. 3. The graph provides a dense, multi-hop network covering major classifications such as Mathematical Logic, Set Theory, Graph Theory, and Combinatorics. Within this structure, the

"Computational Thinking" nodes act as high-degree hubs that aggregate knowledge points across different chapters, effectively breaking the "knowledge silos" common in traditional linear curricula. For example, the CT dimension of "Formal Abstraction" is linked to multiple topics ranging from set algebra to predicate logic, providing the system with the structural capacity to recommend cross-chapter learning paths. This structured knowledge base serves as the essential factual foundation for the subsequent Graph RAG-based question-answering services, ensuring that the tutoring system's responses are grounded in verified pedagogical logic rather than probabilistic language generation [11].

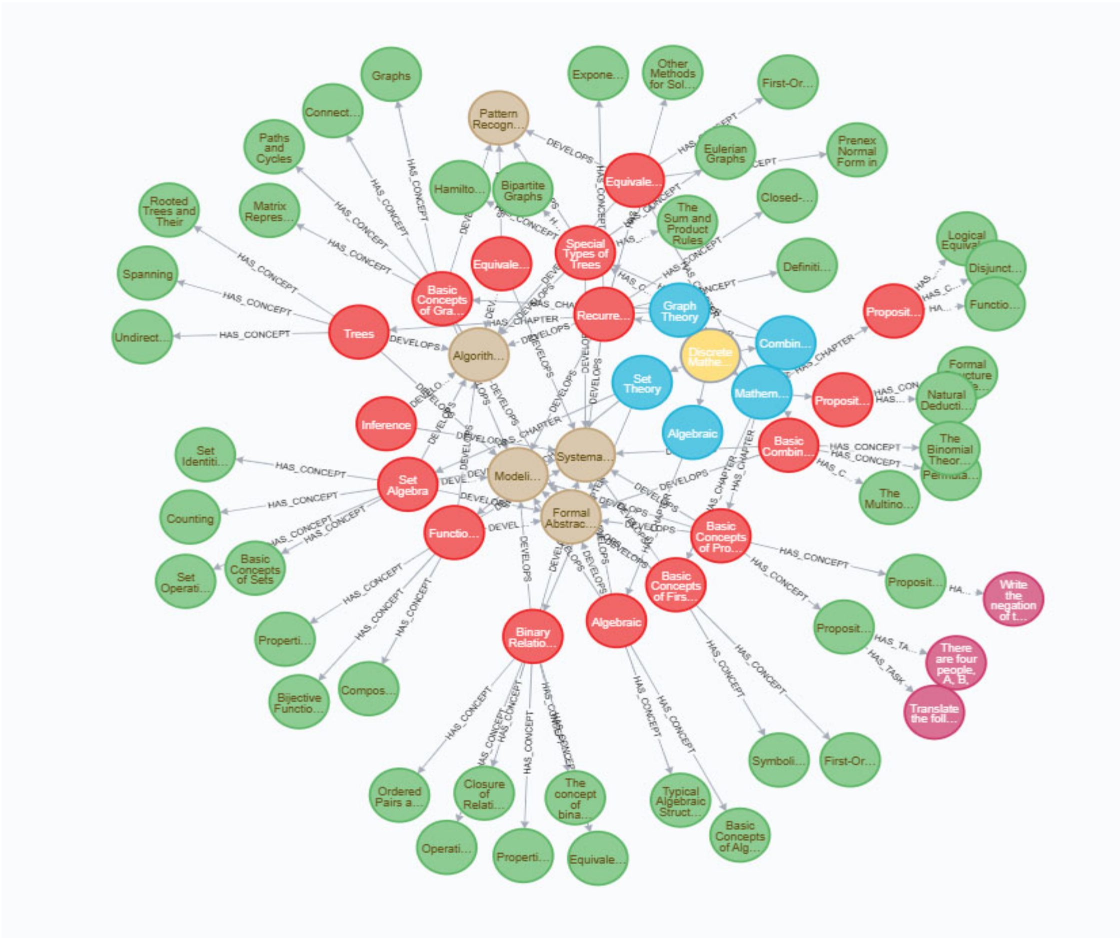

Fig. 3. Discrete mathematics knowledge graph (partial).

IV. Intelligent Question Answering System

A. System Architecture Design

Based on the constructed discrete mathematics knowledge graph, this study significantly enhances the model's cognitive and question answering capabilities in this vertical domain by deeply integrating it with the DeepSeek-V3 large language model [12]. As illustrated in Fig. 4, the core workflow is as follows: first, the system receives a user's natural language question, which is parsed by the LLM and transformed into a Cypher statement that can query the knowledge graph; second, the query is executed to accurately retrieve relevant concepts, theorems, and relational triples from the graph; next, these structured knowledge elements are integrated into a preset prompt template, providing factual context and reasoning support for the LLM; finally, the LLM generates an answer that incorporates the knowledge from the graph.

Based on the constructed discrete mathematics knowledge graph, this study has designed an intelligent question answering system [13]. Its core idea is to enhance the system's question-answering capability in specialized domains through the deep integration of the knowledge graph and large language models. The system employs a layered architecture design to ensure efficient collaboration among all modules. The overall system is built on a Browser/Server (B/S) architecture: the frontend uses the Vue 2.0 framework to implement the user interface, the backend employs the Flask framework to handle business logic, and Axios is used for frontend-backend data communication. The data layer integrates the Neo4j graph database to store the knowledge graph and utilizes the D3.js library for the visual presentation of knowledge. As shown in Fig. 5 and Fig. 6.

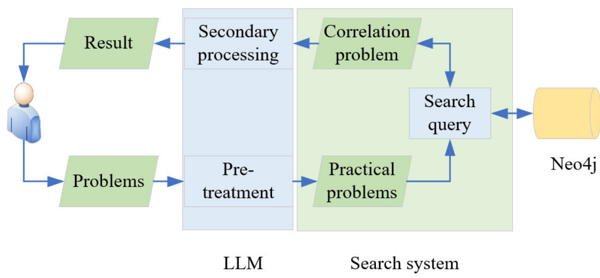

Fig. 4. Question answering workflow.

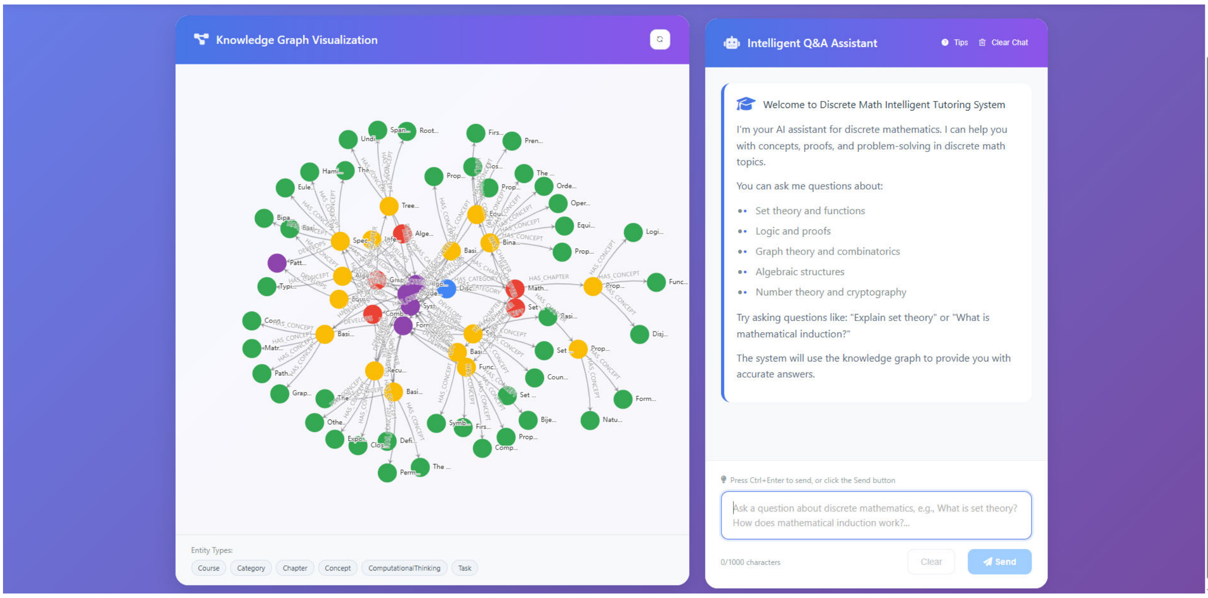

Fig. 5. The main interface for system interaction.

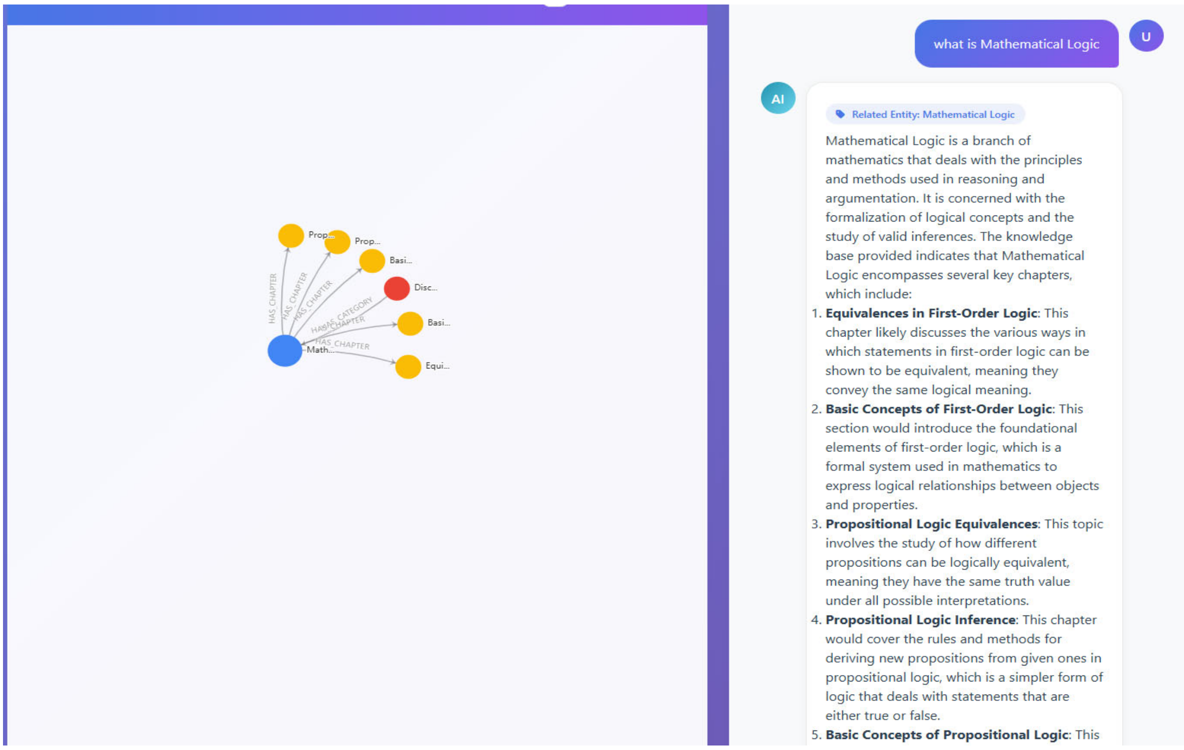

Fig. 6. System interaction example.

B. *Performance Evaluation*

To systematically evaluate the performance differences of Grap-Based Retrieval-Augmented Generation (Graph RAG), standard Retrieval-Augmented Generation (RAG), and pure LLMs in domain-specific question answering tasks, and to simultaneously assess the system's capability to recognize question types, this study designed a comprehensive evaluation framework covering both generation and classification dimensions.

For the generation performance evaluation, the experiment used the same set of test questions as input while strictly controlling variables such as the underlying large language model and prompt templates. Responses were generated using three distinct methods，pure LLM, standard RAG, and Graph RAG, to compare the extent to which different knowledge, augmentation strategies improve answer accuracy and reasoning depth [14].

For the classification performance evaluation, we constructed 50 representative questions for each of the three defined question types—"Debug," "Knowledge," and "Task". Using common classification metrics—accuracy, precision, recall, and the F1-Score, we quantitatively assessed the overall effectiveness and robustness of the system in identifying question types. The specific formulas are as follows:

$$Accuracy = \frac{Number\ of\ correctly\ classified\ questions}{Total\ number\ of\ questions} \quad (1)$$

$$Precision = \frac{Number\ of\ correctly\ classified\ questions}{Number\ of\ questions\ predicted\ as\ that\ class} \quad (2)$$

$$Recall = \frac{Number\ of\ correctly\ classified\ questions}{Total\ Number\ of\ questions\ belonging\ to\ that\ class} \quad (3)$$

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4)$$

First, accuracy was calculated for each type to measure the overall correctness of classification. Next, precision was computed to evaluate how many of the questions predicted as belonging to a given type were correct. Recall was used to measure the proportion of questions that actually belong to a type and were correctly identified [15]. On this basis, the F1-Score—the harmonic mean of precision and recall—was further calculated to provide a comprehensive reflection of system performance and to effectively mitigate potential evaluation bias caused by an imbalanced sample distribution. The detailed evaluation results for each question type are presented in Table 2.

Table 2. The performance comparison of QA methods

Template Type	Accuracy	Precision	Recall	F1
Graph RAG	0.85	0.89	0.85	0.87
RAG	0.72	0.75	0.72	0.73
LLM	0.52	0.54	0.52	0.53

Through systematic experimental evaluation, we have obtained the following important results: In terms of generation performance, the method based on knowledge graph enhancement performed the best in all indicators, with an accuracy rate of 0.85, a precision rate of 0.89, a recall rate of 0.85, and an F1-Score of 0.87. In contrast, the accuracy rate of the standard retrieval enhancement generation method is 0.72, with an F1-Score of 0.73, while the accuracy rate of the pure large language model method is only 0.52, with an F1-Score of 0.53. demonstrating the superiority of the proposed approach.

V. Conclusion

This study has successfully developed an intelligent tutoring system for discrete mathematics by integrating knowledge graphs with large language models through a three-tier architecture. By adopting a Graph RAG-based approach, the system explicitly maps computational thinking dimensions to mathematical concepts, effectively addressing knowledge fragmentation and the implicit nature of CT cultivation. Experimental results confirm that the graph-enhanced method significantly outperforms standard RAG and pure LLM methods, achieving an F1 score of 0.87 and an accuracy of 0.85. These metrics demonstrate the system's technical reliability and its superior ability to provide reasoning-rich, factual answers within a specialized vertical domain.

Despite these technical achievements, this research currently lacks empirical evidence regarding direct educational outcomes, as the assessment focused on system performance benchmarking rather than student improvement. Consequently, the claim that the system promotes computational thinking is currently supported by the rigor of the technical framework. To address this gap, future work will prioritize a comprehensive evaluation roadmap including small-scale user studies and pre-test/post-test designs to quantitatively measure the development of students' CT skills. By transitioning from technical benchmarking to pedagogical validation, we aim to provide a more rigorous paradigm for the systematic cultivation of high-order thinking skills in intelligent education.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Kun Luo conducted the research, analyzed the data, and wrote the paper; Yi Kuang provided guidance; both authors approved the final version.

Acknowledgment

This work was funded by the Teaching Reform Research Project for General Undergraduates in Hunan Province (202401000570).

Eferences

[1] B. Singh and C. Kaunert, "Computational thinking for innovative solutions and problem-solving techniques: Transforming conventional education to futuristic interdisciplinary higher education," *Revolutionizing Curricula Through Computational Thinking, Logic, and Problem Solving*, vol. 1, pp. 60−82, 2024.

[2] J. M. Wing, "Computational thinking," *Communications of the ACM*, vol. 49, no. 3, pp. 33−35, 2006.

[3] V. V. Kochergin *et al*., *Chair of Discrete Mathematics*, Moscow University Mathematics Bulletin, vol. 79, no. 6, pp. 322−333, 2024.

[4] X. Song *et al*., *Large Language Models in Nonprofit and Public Good Sectors*, 2024.

[5] B. Choudhury and V. Deka, "A review on question answering system over knowledge graph," *Authorea Preprints*, 2025.

[6] S. Michie, R. West, and J. Hastings, "Creating ontological definitions for use in science," *Qeios*, 2019.

[7] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with ChatGPT," arXiv preprint, rXiv:2302.11382, 2023.

[8] N. Xue *et al*., "How to realize the knowledge reuse and sharing from accident reports? A knowledge-driven modeling method combining ontology and deep learning," *Journal of Loss Prevention in the Process Industries*, 2025, vol. 94, 105525, 2025.

[9] J. Wei *et al*., "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24824−24837, 2022.

[10] E. Yeo *et al*., "Demystifying long chain-of-thought reasoning in LLMS," arXiv preprint, arXiv:2502.03373, 2025.

[11] L. Luo *et al.*, "GFM-RAG: graph foundation model for retrieval augmented generation," *Advances in Neural Information Processing Systems*, vol. 38, pp. 36371−36405, 2026.

[12] C. Zhao *et al*., "Insights into deepseek-v3: Scaling challenges and reflections on hardware for ai architectures," in *Proc. 52nd Annual International Symposium on Computer Architecture*, 2025, pp. 1731−1745.

[13] S. Chowdhury and B. Soni, "Handling language prior and compositional reasoning issues in visual question answering system," *Neurocomputing*, vol. 635, 129906, 2025.

[14] D. Edge *et al*., "From local to global: A graph rag approach to query-focused summarization," arXiv preprint, arXiv:2404.16130, 2024.

[15] K. Nassiri and M. Akhloufi, "Transformer models used for text-based question answering systems," *Applied Intelligence*, vol. 53, no. 9, pp. 10602−10635, 2023.