Exploration of the Application of Multi-Agent Collaboration in the Teaching of Operating System Principles Course

Yanpeng Ou

Hohhot Minzu College, Hohhot, China

ouyanpeng@126.com

Abstract. With the rapid development of artificial intelligence technology, the application of Multi-agent systems in the field of education has demonstrated immense potential. In response to challenges in the teaching of operating system principles, such as the difficulty in understanding abstract concepts, the weakness in practical components, and the lack of personalized teaching. It proposes a teaching model framework based on multi-agent collaboration, which achieves dynamic organization of teaching resources, personalized learning path planning, intelligent experimental guidance, and comprehensive learning support throughout the process. This provides new insights for the reform of teaching core computer science courses.

Keywords: Multi-agent; Operating System Principles; Collaborative Learning; Knowledge Graph

1 Introduction

In recent years, artificial intelligence technologies, represented by large language models, have brought new opportunities for educational transformation[1]. As a vital branch of distributed artificial intelligence, Multi-Agent Systems (MAS) can simulate the process of human teams solving complex problems through the collaboration and division of labor among multiple autonomous agents[2]. Introducing multi-agent technology into operating system instruction and establishing a "teacher-agent-student" tripartite collaborative teaching model holds the potential to break through the bottlenecks of traditional teaching methods and achieve intelligent, personalized learning processes.

1.1 Research Status

The application of agent technology in the field of education has achieved preliminary results. The research by Swan et al. (2023) [3]shows that multi-agent systems can simulate classroom teaching interactions and promote personalized learning. The Agentverse framework proposed by Chen et al. (2023)[4], by simulating human team collaboration, has verified the advantages of multi-agent systems in handling complex

A. Y. M. A. Islam et al. (eds.), *Proceedings of the 2025 International Conference on Educational Technology and Management Information Systems (ETMIS 2025)*, Advances in Computer Science Research 129,
https://doi.org/10.2991/978-94-6239-630-2_51

tasks. The team of Professor Gu Xiaoqing from East China Normal University has conducted in-depth research on the reshaping effect of multi-agent systems on educational technology systems and proposed a theoretical framework in which multi-agent systems promote the achievement of goals through mechanisms such as communication, collaboration, planning, and execution[5]. Zhai Xuesong et al. [6]proposed that in multi-agent environments, compared to single-agent settings, learners tend to actively adopt multi-dimensional questioning strategies, thereby solving complex learning problems more effectively.

However, the existing research mostly focuses on general education or the development of specific subject tools, and there is a lack of systematic and structured research on multi-agent teaching application frameworks for highly abstract core computer science courses such as operating systems(OS). This study aims to explore the in-depth application patterns of multi-agent collaboration in operating systems teaching.

1.2 Existing Problems

The current teaching of Operating System principles primarily faces the following four issues:

Firstly, the comprehension barrier caused by knowledge abstraction and system complexity[7]. OS involves numerous abstract concepts related to parallelism, concurrency, and resource management. Its dynamic interaction characteristics are difficult to fully convey through static graphics and texts, leading students to often "see the trees but not the forest."

Secondly, insufficient depth, breadth, and personalized support in practical sessions. Existing experiments are mostly verification-based or simplistic in design, lacking exploration into the overall system architecture, design trade-offs, and performance analysis. At the same time, experimental guidance is "one-size-fits-all," failing to provide students of varying proficiency levels with appropriate hints, challenges, or error diagnosis.

Thirdly, deficiencies in the timeliness and continuity of learning support services. Limited teacher-student ratios make it difficult for instructors to provide sustained, in-depth, and personalized guidance to each student. Traditional Q&A channels are inefficient and struggle to form a continuous learning companionship.

Fourthly, the separation between knowledge impartation and value guidance. Current teaching predominantly focuses on explaining technical principles and implementation mechanisms, failing to explicitly and systematically integrate the ideological and political elements inherent in OS design—such as holistic perspective, collaborative spirit, security responsibility, and innovative awareness. This results in a separation between "technique" and "principle," which is not conducive to cultivating students' core competencies that combine solid engineering capabilities with a strong sense of social responsibility[8].

1.3 Research Purposes

The purpose of this study is to address the aforementioned issues by exploring and designing an intelligent teaching framework for Operating System principles based on multi-agent collaboration and knowledge graphs. Its core objective is not to replace instructors, but to serve as a powerful intelligent enhancement tool, extending teachers' instructional capabilities and empowering students' autonomous learning.

The specific objectives include:

1) Developing a composite knowledge graph that deeply integrates domain-specific knowledge and educational objectives. This involves constructing a structured and visualized knowledge graph for the field of Operating Systems to organize fragmented knowledge points into a systematic cognitive framework. More crucially, it entails building a closely associated ideological and political education graph. This graph aims to systematically organize and make explicit the ideological and political elements inherent in OS design, such as systems thinking, holistic perspective, collaborative spirit, security responsibility, and innovative awareness. It will clearly map their relationships to specific technical knowledge points, algorithms, and design trade-offs. This provides structured, visualized content support and logical rationale for the natural, precise, and effective integration of value guidance throughout the entire teaching process, achieving the organic unity of knowledge impartation and value cultivation.

2) Designing and implementing a set of functionally complementary, collaboratively working teaching agents that simulate various roles and interactions in the teaching process, providing end-to-end, personalized learning support from concept learning, algorithm simulation, and experimental guidance to collaborative discussion.

3) Exploring a new, human-machine collaborative, intelligent, and flexible teaching model. This model aims to lower the OS learning curve, deepen students' understanding of the overall system architecture and design philosophy, cultivate their system-level thinking ability, problem-solving skills, and collaborative learning capabilities, and ultimately enhance both the quality of course teaching and learning outcomes.

2 Implementation Plan of the Project

2.1 Graph Construction

First, conduct knowledge graph. The data source is the textbook Computer Operating Systems edited by Tang Xiaodan et al[9]. It is necessary to abstract the core entities, attributes, and relationships in the operating system course. Entity types include: core concepts, such as processes, threads, address spaces, file descriptors; key algorithms, such as First-In-First-Out scheduling (FIFO), Least Recently Used (LRU), Banker's Algorithm; system mechanisms, such as interrupt handling, system calls, memory paging; important data structures, such as Process Control Blocks (PCB), page tables, File Control Blocks (FCB); experimental modules, such as memory management experiments, file system experiments; typical problems, such as deadlock, fragmentation, thrashing; and learning resources, such as textbook chapters, papers, classic code snippets, video explanation links. Relationships are used to connect these entities, for example: an "inherits from" relationship indicates that a thread is a type of lightweight process; a "consists of" relationship indicates that an address space includes code segments, data segments, etc.; an "implemented as" relationship indicates that the LRU algorithm can be implemented using the second-chance method; a "used for managing" relationship indicates that page tables are used to manage virtual memory; a "may lead to" relationship indicates that improper concurrency control may lead to deadlock; an "associated with experiment" relationship indicates that the concept of process scheduling is associated with CPU scheduling simulation experiments; and a "common misconception" relationship, such as equating virtual memory size with disk swap space size. Taking the chapter on process synchronization as an example, the knowledge graph as shown in Figure 1.

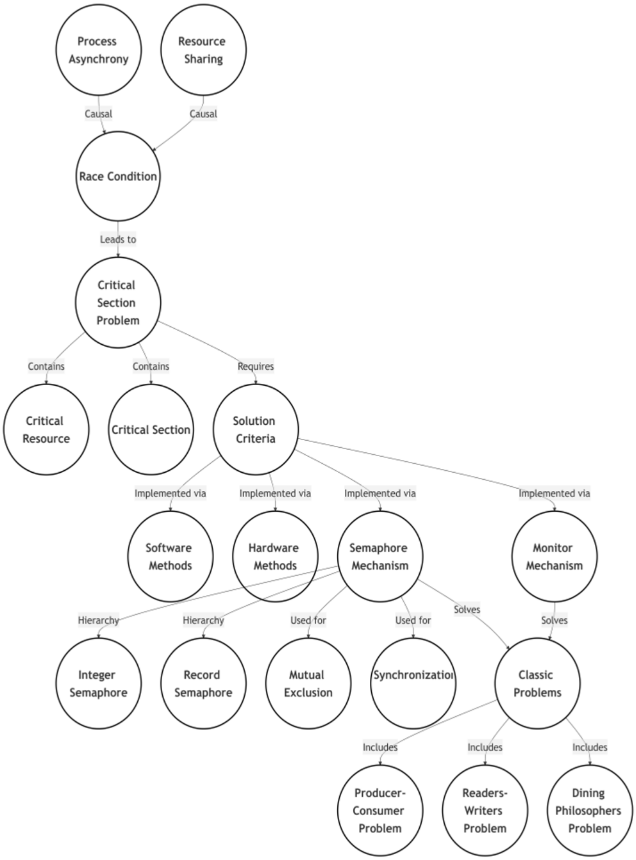

Fig. 1. Knowledge graph of process synchronization

Second, construct the ideological and political education graph. To implement the fundamental task of fostering virtue through education, it is necessary to simultaneously build an ideological and political education knowledge graph that maps onto the professional knowledge graph. Data sources primarily include the curriculum ideological and political case database and related philosophical and social science materials. This graph focuses on the scientific spirit, engineering ethics, national sentiment, and philosophical thinking embedded in the principles of operating systems. Entity types include: ideological and political elements, such as system perspective, holistic view, security awareness, collaborative spirit, craftsmanship spirit, innovative consciousness, ethical dilemmas; classic cases, such as the struggle and development of domestic operating systems, major security incidents caused by critical system vulnerabilities, fairness issues in resource sharing; value goals, such as self-reliance and strength in science and technology, cyberspace security, and responsible innovation. Relationship definitions are used to build bridges between professional knowledge and ideological-political elements, for example: an "embodies ideological and political elements" relationship indicates that process synchronization mechanisms embody the spirit of collaboration and order; an "reflects design philosophy" relationship indicates that the hierarchical structure of operating systems reflects systems theory and modular thinking; an "associated with security ethics" relationship indicates that memory protection mechanisms are associated with information security and national interests; and a "maps to engineering trade-offs" relationship indicates that the design of scheduling algorithms maps to the eternal theme of efficiency versus fairness. The ideological and political education graph and the professional knowledge graph are interconnected through cross-graph relationships, forming a unified network of "knowledge impartation and value shaping."

Third, perform knowledge population. Domain experts and ideological and political education experts collaborate to review, correct, supplement, and enrich the content, particularly by adding implicit design trade-offs, algorithm variants, points students are prone to confuse, and profound ideological value connotations. This process is iterative and evolves with teaching feedback.

Finally, implement graph storage, visualization, and interfaces. The constructed dual graphs are stored using the graph database Neo4j to efficiently handle complex relational queries and cross-graph reasoning. Simultaneously, a visualization interface is developed, allowing teachers and students to browse the professional knowledge and ideological-political mappings in an interactive graphical manner. For example, clicking on the "process synchronization" node can expand to display technical details such as semaphores and monitors, while also highlighting associated ideological and political nodes like "collaborative spirit" and "order rules," along with real-world cases. More importantly, the graphs provide rich semantic query services to multi-agent systems through APIs, serving as a composite knowledge engine for agents to conduct knowledge reasoning, value guidance, and question answering.

2.2 Agent Construction

Within the multi-agent system, various types of agents are designed, each with distinct roles. Through communication and collaboration, they collectively build an immersive and responsive learning environment. The overall system architecture is based on collaborative learning theory and the principles of intelligent tutoring systems.

Agent Types and Role Design:

Personalized Learning Navigator Agent: This agent acts as a personal tutor. Its core function is to plan dynamic, personalized learning paths for students based on student models and the dual knowledge graphs. For example, upon detecting a student's weakness in process synchronization, it would not only recommend a technical learning path—such as reviewing the critical section concept, studying semaphore mechanisms, and completing a producer-consumer problem simulation experiment—but may also simultaneously recommend exploring ethical dilemma cases in concurrency control or introduce China's developmental achievements in the field of parallel computing. This achieves resonance between knowledge acquisition and value guidance. Based on the student's completion status and feedback, it can adjust the proportion and sequence of subsequent recommended content in terms of technical difficulty and ideological depth in real-time.Importantly, the agent evaluates student competence based on three key metrics: 1) the mastery and stability of knowledge graph nodes, 2) the independence and efficiency in completing complex tasks, and 3) the strategic and metacognitive abilities demonstrated in challenging tasks. Guided by these assessments, it gradually reduces its support intensity to encourage students to solve problems autonomously. Moreover, for every recommendation and each adjustment in support level, the agent generates clear natural-language explanations. These explanations allow both students and instructors to understand the pedagogical logic behind the agent's decisions, thereby enhancing the system's transparency and credibility.

Virtual Experiment and Simulation Environment Agent: This agent is responsible for building and managing a highly configurable and observable OS virtual experiment sandbox. Instructors can directly publish experiment tasks generated by the agent according to application scenarios and teaching objectives, along with the corresponding supporting instructions, through the Head Song practical teaching platform. The experimental environment is built and distributed based on Docker containers, ensuring consistency, isolation, and reproducibility across various terminals. At the same time, the code, configuration files, and operation logs generated by students during the experiment process can be conveniently version-controlled, submitted, and collaborated on via Git, thereby forming a complete and traceable experimental workflow.For instance, when studying page replacement algorithms, a student can instruct the agent via natural language or a graphical interface to create a virtual memory access stream with a sequence length of 100 and 4 memory frames, then simulate it using FIFO, LRU, and Clock replacement algorithms, visualizing each page swap process and the final page fault rate. The agent not only executes the simulation but also explains the rationale behind each decision and allows students to "rewind time" to observe and conduct hypothetical analysis at critical steps. For more advanced kernel experiments, this agent can act as an intelligent debugging assistant. After a student modifies teaching kernel code , it can understand the code's intent and, in response to compilation errors or runtime exceptions, provide potential root cause analysis and repair suggestions by leveraging the knowledge graphs. Concurrently, it can prompt during the experiment about the connotations of craftsmanship spirit related to code safety and robustness.

Intelligent Q&A and Dialogue Agent: This agent serves as a student's on-call consultant. It integrates semantic search based on the dual knowledge graphs and natural language understanding/generation capabilities powered by large language models. Students can ask questions in natural language, such as "Why can Multi-level Feedback Queue scheduling balance the experience of both short and long jobs?" The agent extracts relevant concepts from the professional knowledge graph—like Shortest Job First, Round Robin, priority decay—and design goals—like response time, turnaround time. Simultaneously, it can connect to philosophical considerations about the trade-off between system fairness and efficiency from the ideological-political graph, organizing all this into a coherent, easily understandable multi-dimensional explanation, potentially supplemented by dynamically generated diagrams. More importantly, it possesses contextual memory and multi-turn dialogue capabilities, enabling it to deeply track the entire discussion process of a complex question. It not only answers "what" but also guides students to think about "why" and "how" through Socratic questioning, inspiring them towards value reflection.

Collaborative Discussion and Review Agent: This agent is designed to promote social construction among learners. It can organize group projects, such as designing a simple file system, by assigning roles within the group and providing milestone suggestions for the project. During group discussions, it can act as a collaborator or facilitator, monitoring the conversation, introducing relevant nodes from the professional knowledge graph at appropriate moments, or pointing out logical contradictions within the discussion. In the code review phase, it can conduct an initial automated review of the kernel module code submitted by the group, based on coding standards, OS design principles, and software engineering ethics. It flags potential performance issues, possible security vulnerabilities, race conditions, or implementations that do not align with system design philosophy, and generates review comments. Additionally, it can guide the team to reflect on the project's social value, providing a basis for further discussion and revisions by group members.

2.3 Multi-Agent Collaboration Mechanism

a) Task Decomposition and Allocation

The core of the multi-agent collaboration mechanism lies in achieving orderly task decomposition, dynamic inter-agent collaboration, and resolution of potential conflicts through a central coordinator. In this system, the Personalized Learning Navigator Agent is designed to function as this central coordinator, responsible for orchestrating the entire execution workflow of a learning task.

When a student proposes a complex learning task, the system initiates the following collaborative workflow,as shown in Figure 2.

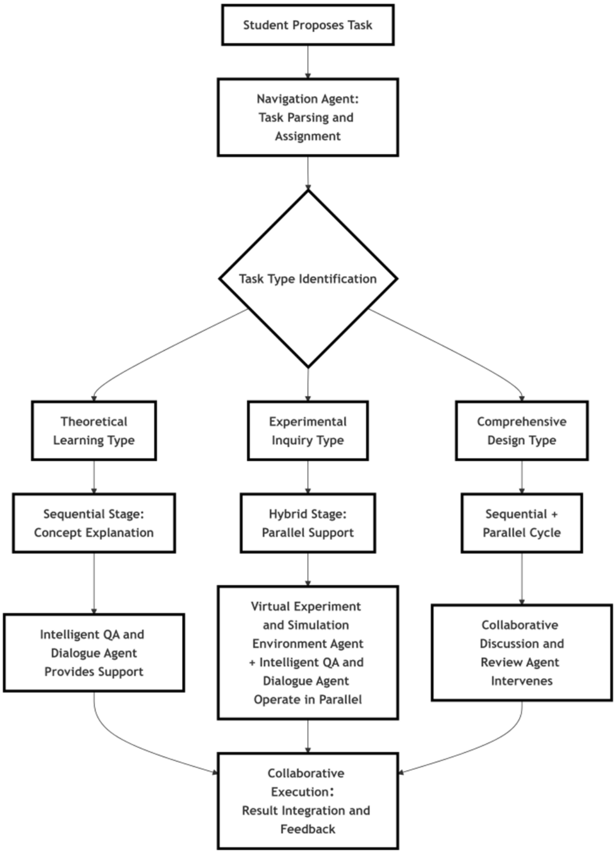

Fig. 2. Multi-Agent Collaboration Mechanism

1)Task Parsing and Capability Assessment

Personalized Learning Navigator Agent as the central coordinator,first conducts a comprehensive analysis of the learning task. It parses the technical type and difficulty of the task, identifies the key knowledge and skill components involved, and pinpoints potential connections with ideological and political education. Simultaneously, the coordinator retrieves and utilizes the latest student model to holistically assess the student's current level of professional knowledge, proficiency in practical skills, and stage of development in value cognition.

2) Role Allocation and Strategic Pre-Coordination

Based on the integrated diagnosis of the task and the student, the central coordinator dynamically selects and activates a set of the most suitable functional agents, such as the Virtual Experiment Agent, the Intelligent Q&A Agent, or the Collaborative Review Agent. Prior to the initiation of collaboration, the coordinator synchronizes the task objectives, contextual information, and the student's specific status with all participating agents. It pre-establishes the fundamental rules for collaboration and the priority order of instructional goals, thereby aligning the agents' action directions from the outset and minimizing the potential for strategic conflicts.

3)Collaborative Execution and Dynamic Conflict Resolution

Each agent carries out its instructional activities in parallel within the established framework. The central coordinator monitors the entire task execution flow in real-time, tracking recommendations issued by each agent and the student's corresponding interactions. Upon detecting potential conflicts in recommendations from different agents—for instance, when a pedagogical agent suggests consolidation review while an experimental agent encourages exploratory challenge—the coordinator immediately activates its built-in conflict resolution protocol. This protocol operates a multi-factor arbitration model. The decision-making is primarily based on: the degree of mastery and stability of the student's relevant knowledge, the alignment between task requirements and the student's Zone of Proximal Development, the contribution weight of different recommendations to short-term task completion versus long-term capability cultivation, and the comprehensive educational value of various instructional actions in the current context. Following the arbitration, the coordinator generates a clear natural language explanation for both the student and the instructor, elucidating the rationale for adopting a particular strategy—for example, justifying it based on the need to enhance the student's application proficiency—thereby ensuring the transparency and trustworthiness of the system's decisions.

4)Result Integration and Feedback Generation

Upon task completion, the central coordinator is responsible for aggregating and synthesizing the output from all agents. It organically integrates technical learning data, such as code performance analysis and error debugging records, with value-based reflective content, such as discussions on engineering ethics and reflections on the craftsmanship spirit. This process culminates in the generation of a dual-dimensional learning report encompassing both technical summaries and value reflections, providing the student with coherent and insightful learning feedback.

b) Shared Memory Mechanism

Each agent utilizes a shared knowledge base to store interaction history, learning trajectories, teaching resources, and ideological-political cases, enabling the continuous accumulation and cross-agent reuse of experience. This includes a student's test scores on professional knowledge graph nodes, their operation sequences and results in virtual experiments, the depth and types of questions in dialogues with the Q&A agent, their contribution level in collaborative discussions, as well as their participation in discussions on ideological-political cases and tendencies in value judgments.

Leveraging this data, the system can more accurately assess a student's mastery level for each knowledge point, their preferred learning style, common error types, as well as their ideological trends and stage of value cognition. All agents base their behavioral decisions on the current student model as a key input, thereby achieving truly personalized adaptation and tailored teaching.

3 The Outcomes of the Application Practice

The course class was divided into two groups: the experimental group and the control group. In the analysis of the evaluation results, the academic performance of the experimental group was significantly improved. Compared with the control group, the experimental group had a higher average score in the final exam, and the improvement in the pass rate of difficult modules such as process synchronization and memory management was more significant. At the same time, both the completion rate and the excellence rate of the experiment were significantly increased. In terms of learning efficiency, the average online learning time of students in the experimental group increased, but more importantly, the effective learning rate (the number of knowledge points mastered per unit time) was significantly improved. The instant response function of the Q&A agent greatly reduced the average waiting time for students' questions and significantly increased the problem-solving rate.

From the perspective of ability development, through inquiry-based learning with multi-agent collaboration, students' systems thinking ability was significantly enhanced. The course design report shows that a higher proportion of students in the experimental group could integrate knowledge from multiple modules to solve problems. In addition, collaborative learning promoted by the learning partner agent significantly improved the quality of team projects. In terms of satisfaction evaluation, the course satisfaction survey shows that the experimental group scored significantly higher than the control group in dimensions such as “vividness of teaching content,” “timeliness of experimental support,” and “personalization.” Students' feedback indicates that “the visual explanation of the agent makes abstract concepts more vivid and understandable,” and “24-hour online answers solve the problem of limited learning time.”

4 Conclusion

To address the pain points in operating systems courses, such as abstract theory, complex practical components, and insufficient personalized support, this paper proposes an intelligent teaching solution based on multi-agent collaborative technology. The system is built upon a dual knowledge graph integrating professional and ideological-political education, with the Personalized Learning Navigator Agent serving as the central coordinator. It collaborates with agents such as the Virtual Experiment Agent, Intelligent Q&A Agent, and Collaborative Review Agent to achieve dynamic task analysis, personalized path planning, conflict coordination, and arbitration, while seamlessly incorporating value guidance into technical instruction. The solution faces technical challenges, including the design of multi-agent collaborative logic and the interpretation of complex code, as well as pedagogical challenges such as deep teacher involvement and human-computer interaction design. Looking ahead, with advancements in technologies like large language models, the system holds promising prospects for scenario-based ideological-political guidance and open resource sharing. It has the potential to support the cultivation of outstanding engineering talents equipped with systems thinking and a strong sense of social responsibility.

Acknowledgment

Fund Project: Research Project of the Education Science Research "14th Five-Year Plan" of the Inner Mongolia Autonomous Region in 2024(NGJGH2024135) and the Curriculum Ideology and Political Education Project of Hohhot Minzu College (KCSZ-SFKC-2025017, 2025).

References

1. Li Zhengtao. "Learning and Evolution in the Intelligent Age: Reconstructing the Human 'Learning and Teaching' Paradigm." Journal of Chinese Education, 2024, pp. 48-53.https://kns.cnki.net/kcms2/article/abstract?v=YQFK0XGjx41w7_ZjuM1akRITFPoiClMvJzyQH_Rsjc53l-BVjTJTT0luBnd2ecWznwKflEe0K888iaP4fw_zPIy-yCiWS_zU_eQr_RauJSc6uOYrTlwFpaUtxEIw4YXiy_mhKpl-V_CaQztDS-3gkB-PC9B4C1C97pCAYSVGHG0JGOb6vmiYXw==&uniplatform=NZKPT&language=CHS.
2. Chen Changfeng. The Rise of Intelligent Platforms and the Emergence of Intelligent Agents: How Large Models Will Transform Society and Civilization [J]. Journalism Research, 2024, (02): 15-24+48. DOI:10.15897/j.cnki.cn51-1046/g2.20240226.001.
3. Swan, M., Kido, T., Roland, E., & Santos, R. P. dos. Math Agents: Computational Infrastructure, Mathematical Embedding, and Genomics(Version1), 2023.arXiv. https://doi.org/10.48550/ARXIV.2307.02502.
4. Chen, W., Su, Y., Zuo, J., et al. (2024). Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents. arXiv preprint arXiv: 2308.10848. https://doi.org/10.48550/arXiv.2308.10848.
5. Gu Xiaoqing, Hao Xiangjun. “Wukong’s Hair: Multi-Agent Systems Reshaping Learning Technology Systems.” Journal of East China Normal University (Educational Sciences), 2025, vol. 43, no. 05, pp. 16–29.DOI:10.16382/j.cnki.1000-5560.2025.05.002.
6. Zhai Xuesong, Ji Shuang, Jiao Lizhen, et al. "An Empirical Study on Multi-Agent-Based Human-Machine Collaboration for Solving Complex Learning Problems." Open Education Research, 2024, vol. 30, no. 03, pp. 63–73. DOI:10.13966/j.cnki.kfjyyj.2024.03.007.
7. Wang Jia, Zhang Qian, Qiu Shuang, et al. Personalized Operating System Practice Teaching in the Context of Large Models[J]. Computer Education, 2025, (08): 109-115. DOI:10.16512/j.cnki.jsjjy.2025.08.013.
8. Jin Xueyun, Du Jinlian, Wang Dan. Construction Model and Strategies of Ideological and Political Education Cases in Curriculum—A Case Study of Principles of Operating Systems Course [J]. Computer Education, 2024, (10): 60-64. DOI:10.16512/j.cnki.jsjjy.2024.10.027.
9. Tang, Xiaodan, et al. Computer Operating System (MOOC Edition) . People’s Posts & Telecommunications Press, 2021.