DOI: 10.66069/ojspub.5170260807

Research on Personalized Learning Strategies for Parallel Computing Courses in the Era of Artificial Intelligence: From the Perspective of Situated Learning Theory

Wu Wen, Ye Zhu, Ying Fu, Hao Yang

School of Computer Science, Chengdu University of Information Technology, Chengdu 610225, China

Abstract: ***The rapid advancement of artificial intelligence poses new challenges to the cultivation of parallel computing talents. Traditional teaching approaches often suffer from the dual dilemmas of "separation between learning and application" and "cognitive overload." Grounded in situated learning theory and empowered by artificial intelligence, this study proposes a personalized learning strategy model driven by a dual-wheel mechanism of "situatedness-driven + AI-empowerment." A three-tier dynamic support system encompassing "diagnosis, recommendation, and regulation" is constructed, along with a multi-level task system anchored in authentic engineering contexts. The model is implemented and evaluated in a parallel computing course. The practical case centers on "parallel clustering analysis of large-scale high-dimensional data with holes," guiding students through progressive mixed programming practices with OpenMP, CUDA, and MPI. Evaluation results show that the proposed model significantly improves students' system-level tuning capabilities and engineering confidence: the experimental group achieved a speedup 2.4 times that of the control group. When facing real-world pain points such as load imbalance, 72% of the experimental groups proactively adopted dynamic data redistribution strategies, reducing communication waiting time from 38% to 14%. The study confirms that the effective integration of situated learning and AI empowerment can resolve the long-standing challenges in parallel computing education, achieving the personalized teaching goal of "context-driven instruction and learner-adaptive guidance." Future work will expand interdisciplinary context case libraries, introduce multimodal cognitive load monitoring, and promote the development of an autonomous HPC education ecosystem.***

Keywords: Parallel computing, Situated learning, Personalized learning, AI empowerment, Engineering education.

1. Introduction

The rapid development of artificial intelligence technology, particularly breakthroughs in areas such as large language model training and scientific AI, has made parallel and high-performance computing a core capability for supporting AI infrastructure. The industry has seen a sharp increase in demand for interdisciplinary talent proficient in new computing technologies such as heterogeneous computing and distributed training optimization. However, current university parallel computing curricula generally suffer from a disconnect between teaching content and industrial practice: most courses still rely on traditional parallel programming paradigms like MPI/OpenMP, with abstract content and outdated updates, leading to a pronounced "learning-application gap." At the same time, student populations exhibit significant diversity in programming foundations, disciplinary backgrounds, and cognitive styles, making a uniform teaching model inadequate to meet diverse learning needs, resulting in the structural contradiction where some students "cannot get enough" while others "cannot keep up" [1]. To address this, this study draws on Lave and Wenger's theory of situated learning as its theoretical foundation, and from the perspective of the "learner–object–context" triadic interaction, explores personalized learning strategies for parallel computing courses in the era of artificial intelligence. The research aims to achieve a dual objective: theoretically, to extend the application paradigm of situated learning theory in engineering education; and practically, to provide a transferable pathway reference for personalized teaching reform of core courses under the context of emerging engineering education [2].

2. Theoretical Framework: The Integration Mechanism of Situated Learning and Personalized Learning

2.1 Core Mechanisms of Situated Learning

Situated learning theory emphasizes the dynamic construction of knowledge within authentic social and practical contexts. Lave and Wenger [3] [4] proposed the concept of "legitimate peripheral participation," which reveals the essence of learning: learners evolve from peripheral participants to core contributors by **integrating into** a community of practice. The "cognitive apprenticeship" model developed by Collins et al. [5] [6] provides an actionable implementation pathway for this process, encompassing three key phases—expert modeling, **scaffolding**, and gradual fading—thereby enabling the explicit transmission of tacit engineering experience [7].

2.2 Technological Empowerment of Personalized Learning

To address individual differences among learners, personalized learning leverages artificial intelligence technologies to achieve precise adaptation. Learning analytics can track behavioral traces in the learning process, knowledge graphs can represent complex relationships among concepts, and intelligent recommendation systems can dynamically match resources and tasks based on learner profiles, thereby forming an adaptive response mechanism to prior knowledge,

cognitive styles, and interest preferences [8].

2.3 The "Situation-Driven + AI-Empowered" Dual-Engine Model

Integrating the above theoretical perspectives, this study proposes a dual-engine model of "context-driven and AI-empowered" approaches (see Figure 1). The model comprises two core mechanisms: first, a situated embedding mechanism, which anchors parallel computing knowledge points in authentic AI training and scientific computing scenarios (e.g., communication optimization in distributed training and breaking through the memory wall), using real engineering tasks to stimulate intrinsic learning motivation; second, an intelligent support mechanism, which constructs a closed-loop system of "diagnosis–recommendation–regulation" to achieve dynamic optimization of learning paths. The synergy between these two mechanisms ultimately realizes the personalized teaching goal of "teaching adapted to context and guided by learner characteristics" [9].

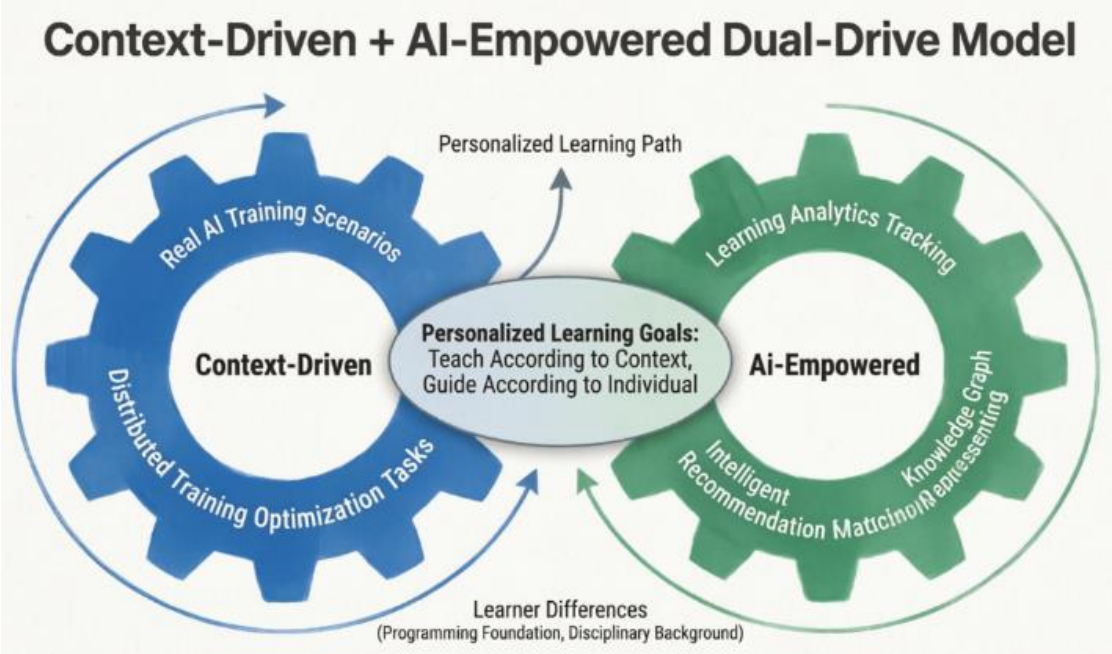

Figure 1: The "Situation-Driven + AI-Empowered" Dual-Engine Model

3. Current Landscape: Pain Points and the Deficit of Situated Context in Parallel Computing Education

Drawing on observations of current parallel computing teaching practices, a review of relevant educational literature, and an analysis of industry technological evolution trends, this study identifies significant deficits in situated context and structural pain points in university parallel computing courses across three dimensions: curriculum content, learner adaptation, and environmental support [10].

3.1 The "Generational Mismatch" Between Curriculum Content and Industry Demands

Current curricula commonly suffer from outdated technology stacks and fragmented scenarios. Syllabi largely remain anchored in traditional MPI/OpenMP syntax instruction. Although this instruction serves as the foundation of parallel computing, it exhibits a significant disconnect from the contemporary AI large-model training ecosystem represented by PyTorch Distributed, DeepSpeed, and Megatron-LM. Students generally face the practical dilemma of "mastering basic communication primitives but struggling with multi-GPU cluster environment configuration and distributed training framework invocation." Meanwhile, experimental designs tend to be oversimplified (i.e., limited to "toy" examples): they mostly remain at the verification level—such as parallel summation and matrix multiplication—lacking realistic workload injection. Without the introduction of advanced engineering challenges like race conditions, communication bottleneck analysis, and fault-tolerant checkpointing, students find it difficult to develop the capability to handle the uncertainties of complex concurrent systems, resulting in a severe disconnect between theoretical understanding and engineering practice.

3.2 Heterogeneity of Learner Profiles and Cognitive Conflicts

As a typical high-cognitive-load course, parallel computing is characterized by a highly heterogeneous learner cohort. Teaching observations reveal three common types of learner profiles in current classrooms. The first is the theoretical deep-diver: often with a computer science background, skilled in deriving communication complexity and consistency models, but frequently experiencing "code implementation anxiety" due to the tediousness of environment configuration. The second is the engineering tuner: enthusiastic about improving performance by adjusting parallel strategies, yet is prone to treating distributed frameworks as a "black box," and lacks systematic diagnostic approaches when facing low-level memory consistency errors or NCCL communication anomalies. The third is the domain applier: who come from interdisciplinary fields such as geoinformatics, bioinformatics, or meteorology, whose core needs are the usability of toolchains and the decoupling of domain-specific problems from the underlying computing infrastructure. They often experience frustration early in the course due to weak Linux

foundations or dependency conflicts. A uniform teaching pace and resource provision cannot accommodate these differences, often resulting in cognitive overload or decline in motivation [11].

3.3 The "Situational Vacuum" in Online Learning Environments

Mainstream online teaching platforms still follow the linear paradigm of "video lectures + static assessments," which struggles to accommodate the unique non-deterministic debugging requirements of parallel computing. On one hand, parallel errors (such as deadlocks, livelocks, and load imbalance) are highly stochastic and environment-dependent. Traditional video demonstrations cannot reconstruct the dynamic execution context, and students lack integrated visual communication link monitoring and performance profiling tools, leading to a situation where they "know what happens but not why." On the other hand, parallel program optimization is inherently a collaborative engineering endeavor. Existing platforms lack code-level collaboration mechanisms and real-time performance feedback, reducing the learning process to isolated single-machine exercises and stripping away the original engineering collaboration and performance tuning attributes of high-performance computing.

4. Strategy Formulation: Design of a Situated Personalized Learning Model

4.1 Multi-Tiered Situated Task System

Guided by the developmental trajectory of "cognition–application–innovation" competencies, a three-tier situated task architecture is constructed (see Table 1) to achieve a progressive transition from conceptual understanding to complex engineering problem-solving [12].

4.2 Personalized Learning Path Generation Mechanism

A three-tiered dynamic support system of "diagnose–recommend–adjust" is constructed (as shown in Figure 2) to achieve the precise adaptation and dynamic evolution of learning paths:

Table 1: Task Architecture

Tier	Contextual Features	Typical Tasks	Competency Objective
Cognitive Tier	Simulated Context	Visualizing the convergence effects of Amdahl's Law and Gustafson's Law across different parallel scales	Conceptual Understanding and Intuition Building
Application Tier	Authentic Context	Accelerating distributed training for image classification and analyzing communication overhead based on real-world datasets	Skill Transfer and Engineering Practice
Innovation Tier	Complex Context	Optimizing communication strategies or memory management schemes for large model training targeting domestic AI chip architectures	Problem Solving and Innovative Exploration

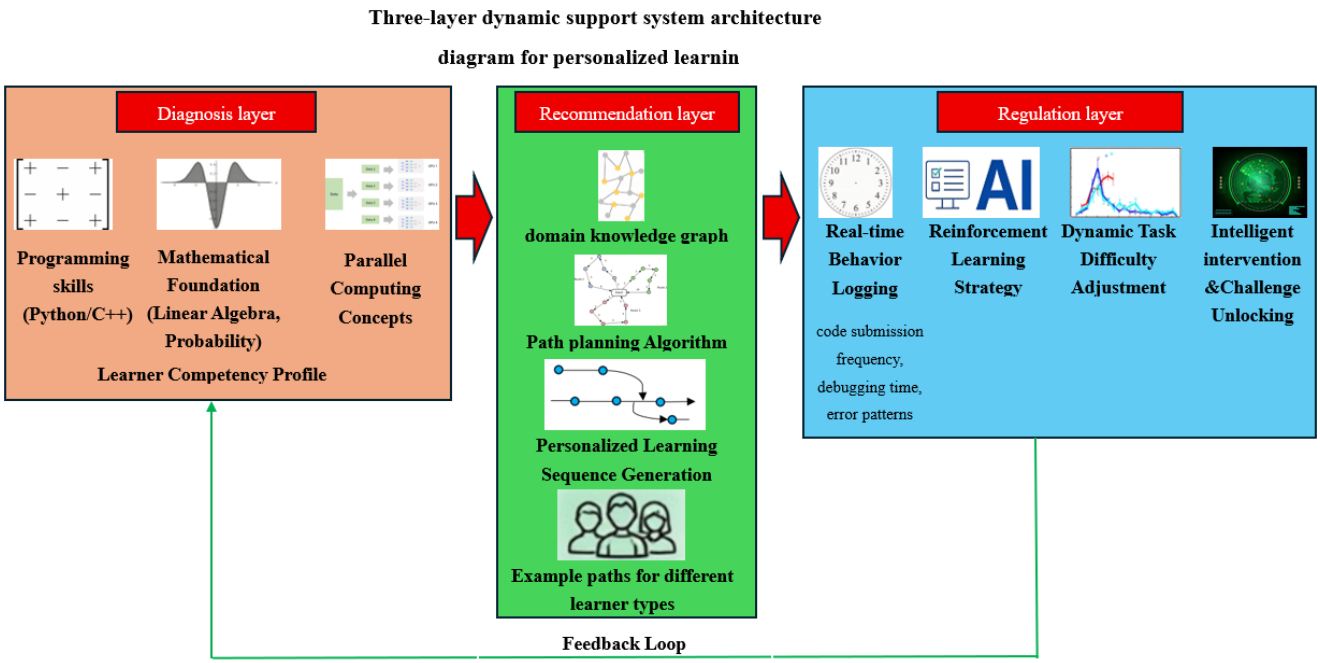

Figure 2: Three-Layer Dynamic Support System for Personalized Learning

1) Diagnosis layer: A pre-assessment evaluates learners across three dimensions—, mathematical foundation (linear algebra, probability theory), and comprehension of parallel computing concepts—to generate an initial learner capability profile.

2) Recommendation layer: Based on a domain knowledge graph, a path planning algorithm is used to generate a personalized initial learning sequence. For example, for domain-applier learners, a path such as "Introduction to CUDA Programming → Practical Parallelization of Image Convolution → Multi-GPU Training Tuning" is recommended, with moderate de-emphasis on low-level communication primitive theory; for theoretical deep-diver learners, the path emphasizes modules on consistency protocols and communication complexity derivation.

3) Regulation layer: Platform behavioral logs (code

submission frequency, debugging duration, distribution of error types) are collected in real time, and reinforcement learning strategies are employed to dynamically adjust task difficulty. When prolonged unsuccessful debugging is detected, Jupyter debugging templates or performance analysis guides are automatically delivered; when high-level performance is identified, extended challenge tasks are unlocked, ensuring that learners always remain within their "zone of proximal development."

5. Practical Case: Course Implementation and Process Analysis

5.1 Core Project Design: Parallel Clustering Analysis of Large-Scale High-Dimensional Data

To overcome the limitations of traditional "toy" experiments, this course features a semester-long hands-on project: Accelerating K-Means Clustering for Massive High-Dimensional Data in Earth Sciences (e.g., remote sensing image interpretation, seismic exploration data). Based on parallel granularity and architectural complexity, the project is decomposed into three progressive sub-modules, comprehensively covering mainstream parallel computing paradigms:

1) Intra-node data-level parallelism (OpenMP): Focuses on single-node multi-core environments. Targeting the nested loops for distance calculation and centroid update, students practice loop unrolling, parallel reduction, and cache-friendly memory layouts. A key focus is identifying and resolving performance degradation caused by false sharing due to frequent writes to adjacent memory addresses by multiple threads.

2) Intra-node many-core acceleration (CUDA): Focuses on GPU heterogeneous computing. High-dimensional vector distance calculations are offloaded to the GPU, where students practice thread block partitioning, shared memory tiling, and coalesced global memory access. Additionally, a parallelized K-Means++ initialization strategy is introduced to algorithmically mitigate the "empty cluster" problem caused by uneven data distribution.

3) Cross-node distributed parallelism (MPI): Focuses on cluster environments. For terabyte-scale datasets, students implement data partitioning and Map-Reduce-style centroid aggregation. Emphasis is placed on practicing the MPI_Allreduce communication primitive and exploring computation-communication overlap strategies to hide cross-node network latency.

5.2 Teaching Implementation Process

Step 1: Situated Introduction

This step takes real geophysical data processing as the entry point. Facing increasingly large and complex exploration datasets, the traditional K-Means algorithm faces two challenges: (1) computational efficiency bottlenecks under massive data volumes, and (2) large irregular blank regions (i.e., data voids) in the dataset. Random initialization often causes centroids to fall into these invalid areas, frequently triggering "empty clusters" and consequently severe load imbalance among nodes. This project guides students to use parallel techniques to achieve optimal acceleration while solving the engineering stability problems posed by complex data distributions.

Step 2: Layered Task Assignment

Foundation Group (Cognition & Standardization): Implement an OpenMP-based K-Means algorithm, master #pragma omp parallel for and critical section protection, complete correctness verification and preliminary acceleration for million-scale 2D datasets, and build intuition for parallel programming.

Advanced Group (Performance Tuning): Implement CUDA kernels, profile memory access patterns using Nsight Compute, and compare GFLOPS and L1 cache hit rates before and after optimization. For 10-million-scale 128-dimensional datasets, the focus is on optimizing warp divergence caused by data voids.

Innovation Group (Complex System Engineering): Implement hybrid MPI+CUDA programming. In a supercomputing simulation environment, handle cross-node data distribution and global centroid aggregation. Analyze strong/weak scalability under different node counts, and attempt to introduce non-blocking communication to optimize pipelines and maximize the hiding of communication overhead.

Step 3: Collaborative Inquiry

Teams collaborate to optimize communication bottlenecks and load imbalance. The AI teaching assistant analyzes MPI profiler logs and CUDA occupancy in real time. When it detects situations such as "idle GPU threads due to data voids" or "an anomalously high MPI communication ratio," it automatically delivers targeted data redistribution strategies and code refactoring snippets.

4.3 Typical Learner Growth Trajectories

Domain-applier learners: Transitioning from an initial state of merely invoking high-level APIs like scikit-learn (a "black-box API invocation" state), they are driven by situated tasks to independently write OpenMP/CUDA code. This enables them to perform customized acceleration for domain-specific data sequences containing data voids (e.g., geoscience remote sensing or bioinformatics). The order-of-magnitude improvement in computational efficiency builds their engineering confidence that "computing power is productivity," transforming them from mere tool users into orchestrators of computational power.

Theoretical deep-diver learners: Through graphical performance profiling tools, they dynamically observe the constraining effects of Amdahl's Law on real clusters. While solving practical challenges such as CUDA shared memory bank conflicts or MPI deadlocks, they translate the abstract theories of "communication complexity" and "memory consistency" into intuitive engineering insights, proactively extending their inquiry into low-level hardware architectures

(e.g., NVLink topology, NUMA architecture).

6. Effectiveness Evaluation: Construction of a Multi-Dimensional Evaluation System

6.1 Design of Evaluation Dimensions

Moving beyond the limitations of the traditional single criterion—"whether the code runs"—this study constructs a three-dimensional evaluation framework encompassing "learning outcomes–process engagement–affective experience" (see Table 2). By integrating parallel computing-specific performance metrics, scalability analysis, and engineering best practices into the core assessment criteria, this framework facilitates a paradigm shift from "syntax verification" to "system-level performance tuning."

Table 2: Three-Dimensional Evaluation Rubric

Dimension	Core Indicators	Measurement Methods
Learning Outcomes	Algorithm correctness, speedup, parallel efficiency, strong/weak scalability	Automated evaluation scripts (job submission via Slurm to capture runtime, GFLOPS, and MPI communication ratio); Code Reviews.
Process Engagement	Number of iterative refactorings, depth of profiler utilization, frequency of collaborative debugging	Analysis of platform Git commit logs; review of screenshots from profiling reports generated by Nsight Compute and mpiP/Vampir.
Affective Experience	Resilience when facing concurrency bugs (e.g., deadlocks/race conditions); engineering sense of achievement after overcoming the "memory wall/communication wall"	Reflective logs with specific technical details (e.g., "How I resolved load imbalance via data redistribution"); Standardized Questionnaire Scales.

6.2 Pilot Validation and Data Observation

In a pilot teaching experiment involving two classes of Computer Science and Technology majors, the experimental group (using the proposed model) was compared with a traditional control group on a final comprehensive project: a hybrid MPI+CUDA clustering task processing one billion seismic exploration data points containing data voids.

Quantitative Results: The average speedup of the hybrid parallel code submitted by the experimental group reached 2.4 times that of the control group ($p < 0.01$). More critically, when facing severe load imbalance caused by data voids, 72% of the teams in the experimental group proactively adopted dynamic data redistribution or non-blocking communication strategies, reducing the MPI communication wait time ratio from 38% in the control group to 14%. The overall parallel efficiency remained above 75% even at a 64-node scale, demonstrating outstanding system-level tuning capabilities.

Qualitative Findings: High-frequency terms in reflective logs shifted from traditional "syntax errors" to "bank conflicts," "the Allreduce communication wall," "false sharing," and "warp divergence." Over 85% of students reported: "Real high-dimensional data containing voids made me deeply realize how fragile the ideal linear speedup in textbooks is when facing real memory walls and load imbalance. This compelled me to re-examine the mapping between algorithms and underlying hardware."

7. Discussion and Reflection

7.1 Key Success Factors

Anchoring in authentic, pain-point-driven scenarios: Instead of idealized uniformly distributed matrices, the course utilizes geoscience datasets with realistic skewed distributions and data voids. This design compels students not only to pursue computational density but also to confront and resolve genuine engineering pain points such as load imbalance and communication hotspots, achieving a seamless alignment between teaching content and real-world HPC requirements in the industry.

White-boxing implicit bottlenecks: We provide a pre-configured Docker container environment encompassing GCC, OpenMPI, CUDA Toolkit, and the Nsight/mpiP suite. This not only eliminates tedious environment setup but, more importantly, transforms invisible issues—such as cache misses, warp divergence, and MPI communication latency—into intuitive visual profiles, shifting performance tuning from "blind trial and error" to "data-driven" practice.

Progressive scaffolding aligned with cognitive development principles: The course follows a progressive trajectory: OpenMP (shared memory, building parallel intuition) → CUDA (heterogeneous many-core, understanding hardware mapping) → MPI (distributed memory, mastering communication coordination). This effectively mitigates cognitive load and prevents beginners from being overwhelmed when directly confronted with heterogeneous clusters.

7.2 Implementation Challenges and Countermeasures

Challenge 1: The exceptionally high debugging threshold for "ghost bugs" in heterogeneous concurrent environments. MPI hangs, CUDA race conditions, or out-of-bounds memory accesses often lead to prolonged student frustration, as traditional compiler error messages provide few useful clues.

Countermeasure: Constructing a parallel error pattern library and an AI-assisted debugging mechanism. When a student encounters an MPI hang, the AI teaching assistant guides them to use communication trace tools to locate the deadlock cycle; when CUDA results are anomalous, it directs them to run compute-sanitizer to detect race conditions. The AI assistant parses stack traces, matches them against the pattern library, and provides semantic debugging guidelines rather than directly supplying the fixed code.

Challenge 2: Scheduling and fairness of HPC/GPU computing resources. Local hardware varies significantly, and students often lack awareness of the engineering best practices for industrial job scheduling.

Countermeasure: Leveraging university-level HPC or cloud-based JupyterHub to introduce a Slurm job scheduling simulation mechanism. Students must write SBATCH scripts to request compute nodes and GPU quotas; the system then

queues and schedules tasks according to priority and resource limits. This not only ensures instructional fairness but also familiarizes students with authentic submission workflows and resource management norms used in industrial supercomputing centers.

7.3 Transferability of the Model

The core logic of this model—anchoring on real computational bottlenecks, leveraging heterogeneous parallelism as the means, and employing AI as a personalized debugging engine—holds strong potential for transfer to other computationally intensive courses. For example:

Machine learning courses: Introducing distributed deep learning training and massive data preprocessing scenarios. To address the straggler effect and severe load imbalance caused by long-tailed data distributions in heterogeneous clusters, students are guided to: accelerate data augmentation and preprocessing pipelines for massive image/text datasets using OpenMP; implement custom operators (e.g., memory-efficient tiled computation for FlashAttention) with CUDA to overcome memory walls; and perform cross-node gradient aggregation (e.g., Ring-Allreduce) and pipeline parallelism using MPI, with a focus on hiding network latency through non-blocking communication.

Computer graphics courses: Embedding scenarios of cinematic physically based rendering (PBR) and large-scale scene ray tracing. To address ray divergence (which readily induces CUDA warp divergence) caused by complex materials and caustics, as well as memory bottlenecks from massive scenes, students are guided to: optimize the construction of bounding volume hierarchy (BVH) trees and ray traversal using CUDA, minimizing global memory accesses via shared memory; accelerate offline path tracing at the pixel/frame level using OpenMP; and explore the dynamic scheduling of massive frame tasks and final image compositing using MPI in distributed rendering farms.

Through such cross-disciplinary transfer, this teaching model can break down the barriers of siloed courses, allowing students to repeatedly hone their heterogeneous parallel thinking and system-level tuning abilities when facing real-world computational bottlenecks across different domains, ultimately cultivating transferable competencies for solving complex engineering problems.

8. Conclusion and Outlook

This study confirms that, in the era of artificial intelligence, the deep integration of situated learning theory with AI-empowered technologies can effectively resolve the dual dilemmas of the "learning-application gap" and "cognitive overload" in parallel computing courses. By introducing authentic engineering scenarios—such as clustering analysis on massive datasets containing data voids—and implementing progressive hybrid programming practices across OpenMP, CUDA, and MPI, students not only grasp the underlying logic of parallel computing but also achieve a significant engineering capability leap. They transition from theoretical understanding to system-level performance tuning while solving genuine engineering pain points, including load imbalance, the memory wall, and communication bottlenecks. Meanwhile, the AI-driven closed-loop support system of "diagnosis–recommendation–regulation," together with automated performance profiling tools, significantly lowers the debugging threshold in heterogeneous environments. This realizes the personalized teaching goal of "teaching adapted to context and guided by learner characteristics," effectively stimulating students' intrinsic learning motivation and their engineering confidence that "computing power is productivity."

Looking forward, this study will continue to deepen and expand in the following three directions:

First, dynamic expansion of interdisciplinary scenarios and evolution of the case library. We will further integrate cutting-edge AI for Science scenarios, such as single-cell transcriptomics analysis (bioinformatics), global climate modeling (Earth sciences), and molecular dynamics (computational physics), to build an open-source situated case library with self-evolving capabilities. By continuously introducing real computing challenges from industry and scientific research frontiers, the teaching content will remain synchronized with state-of-the-art advancements.

Second, real-time monitoring of cognitive load driven by multimodal data. We will explore the introduction of multimodal learning analytics technologies. By deeply mining students' code submission trajectories, temporal features of debugging logs, and platform interaction behavior data, we can accurately assess their cognitive load and affective states when facing complex concurrency bugs (e.g., deadlocks, race conditions). Based on this, we can achieve fine-grained, dynamic adjustment of task difficulty and AI scaffolding intervention strategies, providing a more empathetic and personalized learning experience.

Third, co-construction of an autonomous and controllable HPC education ecosystem. Aligning with the national strategy for independent innovation in information technology and the construction of computing infrastructure, we will promote the deep adaptation and integration of domestic AI chips (e.g., Ascend, Hygon), domestic parallel programming frameworks, and teaching sandbox platforms. This will build a secure, controllable, and software-hardware synergistic high-performance computing education infrastructure, thereby providing a sustained and secure underlying driving force for the large-scale cultivation of interdisciplinary and strategic computing talent in China's AI era.

Fund Project

2024–2026 Sichuan Provincial Higher Education Talent Cultivation Quality and Teaching Reform Project "Building a New Quality Talent Cultivation System with Deep Integration of Artificial Intelligence and Industry Characteristics" (JG2024-0856).

References

[1] Li, C., Guo, X. W., Chen, J., et al. (2026). Empowering High-Performance Computing Competency Cultivation

Through Interdisciplinary Practice. Computer Education, (4), 3-8.
[2] Tong, Y. Q., & Zhang, Q. R. (2025). Personalized Learning Strategies for Online Education Based on Knowledge Graphs and the CKT Model. Journal of Qiqihar University (Natural Science Edition), 41(4), 77-83.
[3] Lave, J., & Wenger, E. (1991). Situated Learning: Legitimate Peripheral Participation. Cambridge University Press.
[4] Wenger, E. (1998). Communities of Practice: Learning, Meaning, and Identity. Cambridge University Press.
[5] Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. In L. B. Resnick (Ed.), Knowing, Learning, and Instruction: Essays in Honor of Robert Glaser (pp. 453-494). Erlbaum.
[6] Collins, A., Brown, J. S., & Holum, A. (1991). Cognitive apprenticeship: Making thinking visible. American Educator, 15(3), 6-11, 38-47.
[7] Wang, Z. Q. (2023). Instructional Inquiry Based on Situated Cognition and Learning Theory: A Case Study of Li Jilin's "Guilin Landscape". Education Observation, 12(23), 56-60.
[8] He, R. Y., & Wang, S. L. (2018). Research on Personalized Learning Strategies Based on Artificial Intelligence. Curriculum and Teaching Research, (11), 8-10, 20.
[9] Jiang, C. C., Wei, Y. A., & Wu, L. M. (2025). Empowering Personalized Learning with Generative AI: Mechanisms, Transformations, and Trends. Jiangsu Higher Education, (12), 61-68.
[10] He, H., Cheng, Y. Q., Yuan, S. Z., et al. (2022). High-Performance Computing Course Development for Interdisciplinary Talent Cultivation. China Journal of Multimedia and Network Teaching (Early-Month Edition), (12), 79-83.
[11] Miao, F. Q., & Chen, H. Q. (2026). Strategies for AI-Empowered Situated Teaching Reform of Engineering Ethics in the Context of Emerging Engineering Education: A Case Study of the "Engineering and Society" Course for Mechanical Engineering in Sino-Foreign Cooperative Education. Education Development Forum, 2(5), 1-4.
[12] Fu, H. (2024). Research on Personalized Education Strategies: Integrating Adaptive Learning Technology and Blended Learning. Education and Teaching Forum, (28), 134-137.

Author Profile

Wen Wu male, associate professor, research interests: artificial intelligence, high-performance computing. wenwu@cuit.edu.cn