Engineering Educators Bringing the World Together
2025 ASEE Annual Conference & Exposition
Palais des congrès de Montréal, Montréal, QC • June 22–25, 2025
Paper ID #46592

A Unique Course Designed for Graduate Students: Integrating High-Performance Parallel Computing into Machine Learning and Artificial Intelligence

Dr. Handan Liu, Northeastern University

Handan Liu is a Full Teaching Professor of Multidisciplinary Master of Science (MS) programs (Software Engineering, Data Architecture, Information Systems) in the College of Engineering at Northeastern University. Her research interests include heterogeneous high-performance computing, programming structure and algorithms, machine learning and AI, NLP research and development, LLM reasoning and AI agent in engineering courses, improving teaching for engineers.

A Unique Course Designed for Graduate Students: Integrating High-Performance Parallel Computing into Machine Learning and Artificial Intelligence

Introduction

With the rapid development of machine learning and artificial intelligence, the amount of data that needs to be processed is ever increasing, and the models are becoming more and more complex. Traditional single-core CPUs and single-machine memory systems are inadequate for these escalating demands, resulting in prolonged data processing and model training times. The exponential growth of data and computing requirements in machine learning and artificial intelligence highlights the importance of high-performance parallel computing (HPC). Many modern AI systems require efficient algorithms that can take advantage of multi-core processors, multiple GPUs, and distributed systems [1-3].

The convergence of data science with high-performance computing (HPC) and parallelism development is increasingly recognized as essential in both industry and academia. Therefore, both industry and academia are increasingly seeking professionals who are proficient in HPC principles and parallel development to address the challenges posed by massive data processing, machine learning, and AI [3-5].

However, existing curriculum in academia often fails to provide a comprehensive understanding of HPC within context of machine learning and AI. Against this background, the author developed a course, "CSYE7105-High Performance Parallel Machine Learning and Artificial Intelligence", targeted for graduate students in 2020. This course fills this gap by offering graduate students a comprehensive learning experience that combines the fundamentals of parallel computing with practical applications in machine learning and AI.

CSYE7105 guides students through the principles of high-performance computing and engages students in the practice of the emerging parallel-based machine learning paradigm, learning high-performance parallel architectures and parallel programming models, and researching the parallelism of machine learning and deep learning. Students achieve high speed and high performance on heterogeneous cluster architectures, and apply them in multiple fields such as image classification, speech recognition, and natural language processing, etc. The course curriculum balances theoretical concepts with hands-on labs and research projects, fostering both analytical and research skills.

Course Design and Structure

Course Objectives:

The primary objective of this course is to equip students with a comprehensive understanding of high-performance computing (HPC) principles and the emerging paradigm of parallel-based machine learning and AI. By the end of the course, students will:

1. Understand the foundational principles of high-performance computing architectures and parallel programming models.
2. Gain proficiency in implementing parallelized machine learning and deep learning techniques to achieve high speed and performance on heterogeneous cluster architectures.
3. Apply HPC and parallel machine learning techniques to practical fields such as time-series forecasting, image classification, speech recognition, natural language processing (NLP) and large language models (LLM), etc.
4. Develop the skills needed to scale the application of HPC and parallel machine learning to larger HPC supercomputing platforms.

Course Structure:
The course is divided into four key parts to ensure comprehensive coverage of theoretical principles and practical applications. Each part builds on the previous one to provide a progressive learning experience.

Part 1: High-Performance Computing (HPC) Architectures and Parallel Programming Models

High-performance computing (HPC) architectures are designed to solve complex computational problems by leveraging the parallel processing capabilities of multiple computing resources. These architectures typically consist of clusters of interconnected nodes, each with its own processor(s), memory, and storage, working collaboratively to execute tasks simultaneously. HPC systems rely on efficient interconnect networks for high-speed communication between nodes and often utilize specialized hardware, such as GPUs or accelerators, to enhance performance for specific workloads. Software frameworks like MPI (Message Passing Interface) and OpenMP enable the parallelization of tasks across these resources [6-8]. HPC is used in areas such as scientific simulations, big data analytics, and machine learning, where large-scale computations and data processing are essential.

This course introduces students to HPC architectures and parallel programming using OpenMP and MPI. It covers the fundamentals of parallel computing, including types of parallelism (data, task, and pipeline) and explores high-performance architectures like multi-core processors, GPUs, and heterogeneous systems. Students will learn shared memory parallelism with OpenMP, distributed memory parallelism with MPI, and analyze performance metrics such as speedup and scalability through practical examples.

Hands-on practice includes writing and optimizing OpenMP programs for shared memory systems and implementing MPI programs for distributed memory systems. By completing these exercises and analyzing performance, students will gain foundational skills in parallel programming, preparing them to apply these principles to advanced applications like machine learning.

Part 2: HPC Supercomputing Clusters and Their Operations

Northeastern University has its own HPC supercomputing cluster. It is a powerful computational resource designed to handle complex and large-scale computing tasks. Built with commonly used hardware structures found in *top500* supercomputers [9], it operates on CentOS Linux, a stable and secure operating system widely used in enterprise and research environments. The cluster is managed by SLURM (Simple Linux Utility for Resource Management) [10], a robust job scheduler used in many *top500* supercomputers. SLURM efficiently allocates resources such as CPU and GPU nodes, ensuring that tasks are executed in a well-organized and optimized manner.

Students enrolled in this course are provided with sponsored accounts that grant them access to the cluster's computational resources. This includes both multi-CPU nodes for parallel data loading/processing and multi-GPU nodes for computationally intensive parallel tasks, such as machine learning and scientific simulations. By leveraging the HPC cluster, students can perform advanced computations, analyze large datasets, and gain hands-on experience with cutting-edge technology used in industry and research.

This part introduces students to HPC clusters, focusing on their architecture, operational workflows, and practical usage. Students will learn about compute nodes, interconnects, storage, and widely used HPC systems. Key topics include job scheduling and resource allocation using SLURM, cluster management, and essential skills such as logging into HPC systems, setting up environments, transferring data, and executing parallel programs.

Through hands-on practice, students submit and manage jobs, monitor performance, and optimize parallel applications on HPC clusters. By the end of the part, students will have developed a strong understanding of HPC operations and gained practical experience, equipping them for real-world HPC applications.

Part 3: Parallel Data Processing and Machine Learning on Multi-CPU Architectures

Parallel data processing and machine learning on multi-CPU architectures leverage the ability to divide tasks across multiple CPUs to improve efficiency and performance. In parallel data processing, large datasets are split into smaller chunks, which are processed simultaneously by multiple CPUs. This reduces computation time and is particularly useful for data-intensive tasks such as filtering, aggregation, and transformation. Frameworks like Multiprocessing, Spark and Dask enable efficient parallelism by managing task scheduling, memory, and inter-CPU communication.

In machine learning, multi-CPU architectures facilitate faster training of models by parallelizing operations such as matrix computations and gradient updates. Libraries like TensorFlow and PyTorch can distribute workloads across CPUs, allowing for scalable learning on large datasets. Multi-CPU architectures are particularly effective for tasks that involve iterative processes, such as training neural networks or performing ensemble methods like Random Forests. However, performance depends on minimizing bottlenecks in data transfer and synchronization between CPUs, making efficient algorithm design and memory management critical.

This part focuses on parallelizing data processing and machine learning workflows to optimize performance on multi-CPU architectures. Students will learn techniques for partitioning data, managing dependencies, and implementing parallel versions of machine learning algorithms.

Topics include leveraging libraries such as multiprocessing, Scikit-learn and Dask, load balancing, minimizing communication overhead, and performance tuning. Research contents include parallelizing models across multiple CPUs using multiple parallel methods and backends for real-world case studies such as reinforcement learning trains agents (e.g., game-playing AI) and time series forecasting (e.g. stock price forecasting or public transit scheduling).

Through hands-on practice, students will parallelize data processing tasks using Python libraries and train machine learning models with parallel backends. By the end, students will gain practical skills in optimizing workflows, enabling efficient and scalable use of multi-CPU systems.

Part 4: Parallel Deep Learning on Multi-GPU Architectures

Parallel deep learning on multi-GPU architectures enables the efficient training and inference of complex neural networks by distributing computations across multiple graphical processing units (GPUs). GPUs excel at handling parallelizable tasks, such as matrix operations and tensor computations, making them ideal for deep learning workloads [11-12]. In this setup, data or model parameters are split among GPUs to reduce training time and increase scalability. Two common parallelism strategies are data parallelism, where the same model processes different data batches on each GPU, and model parallelism, where different parts of the model are assigned to different GPUs.

This part focuses on implementing and optimizing deep learning models for multi-GPU architectures to achieve significant acceleration using PyTorch. Students will explore GPU architectures, communication between multiple GPUs and research the performance advantages of multiple GPUs over single GPU in deep learning. Research contents include parallelizing models across multiple GPUs using data parallelism and model parallelism, along with case studies in image classification, speech recognition, and large language models, etc.

Through research, students will implement distributed data parallelism (split data onto multiple GPUs) and model parallelism (split neural network layers onto multiple GPUs), as well as study mixed precision methods for high-performance deep learning, and other parallel methods to achieve the feasibility and acceleration of large data sets and large models. Ultimately, students will gain expertise in high-performance parallel deep learning equipping them for advanced research and industry applications.

Teaching Methodology

The course employs a mix of teaching methods to help students understand leading knowledge, skills and insight.

1. **Lectures and Tutorials**: These sessions cover foundational concepts in HPC parallel computing and practical ML/AI applications. Visual aids and examples are frequently used to enhance comprehension.
2. **Hands-On Labs**: Students engage in lab exercises where they write, debug, and optimize parallel code on real HPC systems.

3. **Industry Workshops and Lectures**: Every semester, students are guided to attend relevant online conferences or workshops. Experts from industry share insights into current trends and challenges in high-performance computing and AI.
4. **Research Projects:** Group projects research and develop parallel methodologies on a specific task (such as image classification, speech recognition, natural language processing, etc.), and finally implement these parallel methods with programming on the HPC cluster. Group projects also foster teamwork and allows students to solve complex problems together.

The combination of theory and practice ensures that students develop both conceptual understanding and technical expertise.

Assessment and Outcomes

Assessment methods include:

1. **Quizzes and Assignments**: Evaluating understanding of parallel concepts, algorithms and programming models.
2. **Lab Exercises**: Hands-on operations and implementation of commands and parallel code on HPC systems.
3. **Research Projects**: The primary research project, lasting for half of the semester, accounts for 35% of the grade total. Students complete this project in several steps: first, they choose an area of interest, such as recommendation systems or speech processing, and obtain a data set. Then they write a research proposal which is evaluated and revised by the professor. Students then research and develop the parallel mechanisms required for the project to apply the machine learning and deep learning models and methods used for the project. As a final step, the students analyze the performance metrics obtained. This project comprehensively evaluates students' ability to integrate course concepts into real-world applications.

By the end of the course, students will:

- Demonstrate proficiency in parallel computing techniques.
- Apply HPC methods to solve machine learning and AI problems.
- Develop scalable solutions for real-world applications.

Student Feedback and Impact

Feedback collected over the past five years highlights the course’s impact on students’ academic and professional trajectories.

The course was initially set up with a course cap of 30 seats, and typically enrolled 15 to 30. From the spring semester 2024, due to the popularity and demand of this course, the course cap was increased to 50 seats. As you can see in Figure 1, the course enrollment grew considerably, including a waitlist of 15 seats.

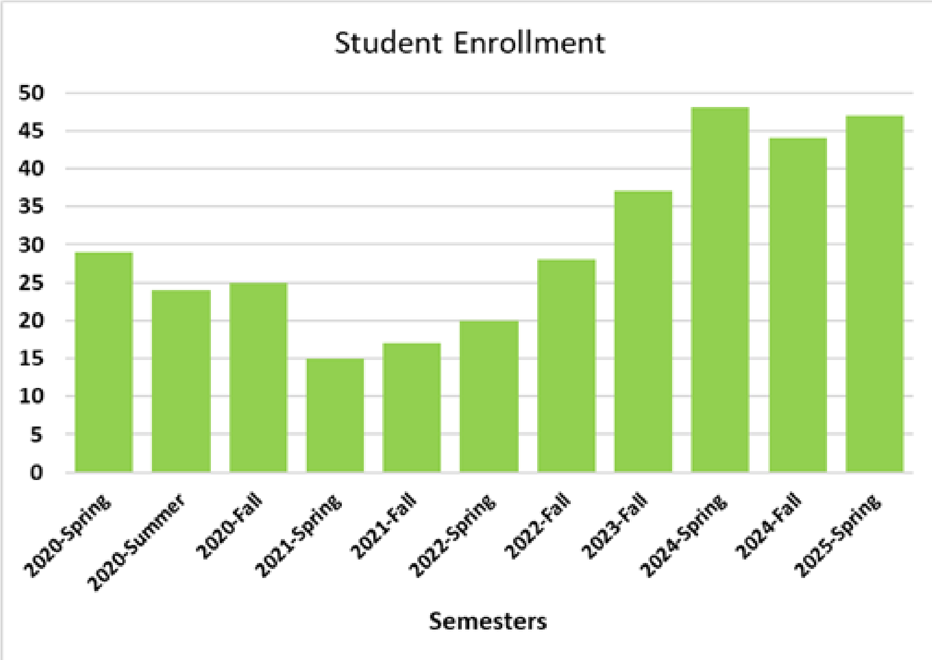

Figure 1: Student enrollment in the past five years

Northeastern University uses TRACE (Teacher Rating And Course Evaluation) at the end of each semester to allow students to conduct an anonymous survey on course content and instructors [14]. The rating range is from 0 to 5. The most important indicator of TRACE is the Teaching Effectiveness (TRACE Evaluation Question "**What is your overall rating of this instructor's teaching effectiveness?**"). From the TRACE feedback over the past five years, it is shown that the students have a very high evaluation of my teaching, which can be confirmed by the following TRACE data analysis:

Semesters	TRACE Teaching Effectiveness
2020-Spring	**4.6**
2020-Summer	**5.0**
2020-Fall	**3.8**
2021-Spring	**4.9**
2021-Fall	**5.0**
2022-Spring	**4.9**
2022-Fall	**4.5**
2023-Fall	**4.8**
2024-Spring	**4.8**
2024-Fall	**4.6**

Furthermore, graduates have reported success in applying the skills learned to:

- Research career paths in academia: for graduate students, this can make them stand out when applying for PhD programs.

For example, one past student shared her experience during a PhD application interview:
I then had the opportunity to demonstrate my knowledge of parallel machine learning during the interview for my PhD program, leading to a profound conversation facilitated entirely by the knowledge I had gained from your instruction. As a result, I was offered a full funded scholarship for my Ph.D. studies at the University of Tennessee.

And,
In my current studies for a Ph.D. in Computer Science, specializing in Usable Security, my Algorithms class proved to be a significant challenge. To successfully complete the final project, I had to integrate algorithmic principles with parallel machine learning, which I accomplished with confidence as a result of my training with Dr. Liu. The mentorship she provided was evident in the way I articulated complex details about parallelism, earning me recognition from my professors and an 'A' grade in a highly demanding course. The mentorship she provided extended beyond academic instruction. She instilled in her students a sense of curiosity and a rigorous approach to research that has proven invaluable.

- Industry positions where parallel computing and AI expertise are in demand.

Several students have also shared their experiences of leveraging HPC parallel methods to solve domain-specific problems, such as biomedical data analysis and financial modeling. For example, one student sent the thanks note as below:
I wanted to share that I have recently joined Fidelity Investments full-time as a Senior Manager in Data Science. My team will be focusing on developing and deploying Audio LLMs to (semi)automate customer support processes.
I had the privilege of completing the Parallel Machine Learning course under your guidance, which was extremely beneficial in helping me productionize and scale Whisper (ASR models) during my time as a part-time Data Scientist. This also played a key role in securing this full-time opportunity. I am truly grateful for your support and mentorship.

Conclusion

The past five years of teaching this course coupled with student interactive feedback have proven that the latest industry knowledge and technologies learned are used to explore and implement parallelism-based projects of machine learning and deep learning in multiple domains on high-performance clusters, making them stand out in both industrial job hunting and academic doctoral program applications. This unique course has attracted a large number of students because of its novel technologies, challenges, and high demand.

As a teaching professor in a highly dynamic field, I have been consistently updating and polishing the course content and technology every semester. This ensures that students are prepared not only for current industry needs but also for the ever-changing data-driven decision-making environment. Future iterations of the course will incorporate emerging technologies such as quantum computing and explore collaborations with industry partners to further enhance learning outcomes.

Reference

[1] Y. You, A. Buluc, J. Demmel. “Scaling Deep Learning on GPU and Knights Landing clusters”. 9 Aug 2017. [Online]. Distributed, Parallel, and Cluster Computing. https://arxiv.org/abs/1708.02983
[2] E. A. Huerta, A. Khan, E. Davis, etc. “Convergence of Artificial Intelligence and High Performance Computing on NSF-supported Cyberinfrastructure”. *Journal of Big Data*. Vol 7, Article number: 88 (2020). https://arxiv.org/abs/2003.08394
[3] Y. Zhang. "Scalable Parallel Machine Learning on High Performance Computing Systems: Clustering and Reinforcement Learning". Purdue University, 2022.
[4] B. Oliveri. “Why Accelerated Data Processing Is Crucial for AI Innovation in Every Industry”. June 7, 2024. https://blogs.nvidia.com/blog/accelerated-data-processing-ai-industry-innovation/
[5] X. Wu, P. Brazzle, S. Cahoon. "Performance and Energy Consumption of Parallel Machine Learning Algorithms". arXiv, 2023. https://arxiv.org/abs/2305.00798
[6] J. Kraus. “An Introduction to CUDA-Aware MPI”. Mar 13, 2013. [Online]. https://developer.nvidia.com/blog/introduction-cuda-aware-mpi/
[7] H.Liu, D. Tafti. “Hybrid parallelism in MFIX CFD-DEM using OpenMP”. Powder Technology. Vol 259, June 2014, Pages 22-29.
[8] Open MPI: Open Source High Performance Computing. https://www.open-mpi.org/
[9] TOP500 Supercomputers. https://top500.org/lists/top500/2024/11/
[10] Slurm Support & Development. https://slurm.schedmd.com/documentation.html
[11] Deep Learning Frameworks. https://developer.nvidia.com/deep-learning-frameworks
[12] NVIDIA CUDA Deep Neural Network library. https://developer.nvidia.com/cudnn
[13] PyTorch. https://pytorch.org/
[14] Student Evaluation of Courses (TRACE) https://catalog.northeastern.edu/undergraduate/academic-policies-procedures/student-evaluation-of-courses-trace/