Bring HPC into a Parallel Programming and Algorithms Course

SGX3 award # 2231406

Revised Course Description

- The CS-425 course introduces students to the foundations of parallel computing including the principles of parallel algorithm design, analytical modeling of parallel programs, programming models for shared- and distributed-memory systems, parallel computer architectures, along with numerical and non-numerical algorithms for parallel systems.
- The course will include material on emerging multicore hardware, shared-memory programming models, message passing programming models used for cluster computing, data-parallel programming models for GPUs, and problem-solving on large-scale clusters using MapReduce.
- A key aim of the course is for students to gain a hands-on knowledge of the fundamentals of parallel programming by writing efficient parallel programs using some of the programming models that students learn in class.

Implementation Schedule

1. Introduction to Parallel Computing
2. Parallel Programming Platforms
3. Principles of Parallel Algorithm Design
4. Basic Communication Operations
5. Analytical Modeling of Parallel Programs
6. Programming Using the Message Passing Paradigm, e.g., Message-Passing Interface (MPI)
7. Programming Shared Address Space Platforms
8. Dense Matrix, Sorting, Searching, and Graph Algorithms
9. Graphics Processing Units (GPUs)
10. Compute Unified Device Architecture (CUDA)

Sample HPC/Gateways Exercise

1.Create accounts and get access to Kettering University High Performance Computer Cluster (KU-HPC). We will be using these machine for all our assignments and projects.

2. Compile hello_mpi.c and submit a job on both machines.

3. Write a simple MPI program that does the following: generate a random integer array, with 100 elements per process, on all processes. hint: initialize the RNG using some function of the rank of the process.

4.Broadcast one value (say the first) from the root (rank 0 or any process really) to all other processes. Let call this the pivot.

```
int root = 0;
int pivot = array[0];
MPI_Bcast(   &pivot,   1,   MPI_INT,
root, MPI_COMM_WORLD );
```

Resource Needs/List

1. Textbooks
- Multicore and GPU Programming: An Integrated Approach 2nd Edition, Gerassimos Barlas (2022)
- Introduction to Parallel Computing, 2nd Edition, Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar (2003)
- Programming Massively Parallel Processors: A Hands-on Approach, 3rd Ed. - David B. Kirk, Wen-mei W. Hwu (2016)

2. Hardware
- Kettering University High Performance Computer Cluster (KU-HPC)
- Multiprocessor computers: for shared-memory parallel programming exercise
- Graphics Processing Units (GPUs)

3. Software & Development Tools
- Parallel Libraries and Frameworks (e.g., OpenMPI, CUDA, OpenCL)
- Debugging Tools

4. Online Platforms and Repositories (e.g., HPC at ORNL)

Gateway Community Mentor Syllabus Suggestions

For further improvement, you can consider the following suggestions:

1. Objectives: Start with clear and measurable learning objectives that students should achieve by the end of the course. For instance: understand and explain the principles and challenges of parallel computing; design and analyze parallel algorithms for efficiency and correctness, etc.

2. Assessments: You should mention how students will be evaluated. Will there be quizzes, projects, mid-term and final exams? For instance, 3 programming assignments on different parallel programming models; 1 course project in which you implement a parallel solution a real-world problem; 4 quizzes and 1 final exam.

3. Guest Lectures or Special Sessions: You can plan to have industry experts or researchers give lectures and introduce students to more real-life use cases. It might be a selling point for the course.

Resources / Science Gateways

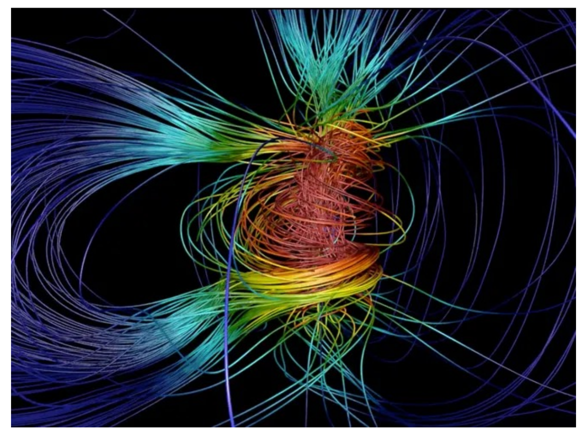
- ORNL HPC Allocation: https://docs.olcf.ornl.gov/accounts/accounts_and_projects.html#request-a-new-allocation
- ORNL's data framework: https://www.olcf.ornl.gov/olcf-resources/rd-project/constellation-doi-framework-and-portal/
- Science Gateways Catalog: https://catalog.sciencegateways.org/#/home

Use Cases

1. High-Performance Computing (HPC):
- Scientific Simulations
- Climate Modeling
- Astronomy

2. Graphics and Gaming:
- Real-time Rendering
- Ray Tracking

3. Machine Learning:
- Distributed Data Processing
- Deep Learning Training



Datasets

- CRAWDAD dataset from IEEE DataPort
- Gait-TNDA from IEEE DataPort
- Labeled images from Populus Trichocarpa genotypes cultivated from ORNL
- Diamonds and Ice dataset from ORNL
- HPC Power and Thermal Characteristics dataset from ORNL
- Scattering dataset from ORNL

Possible Expansions

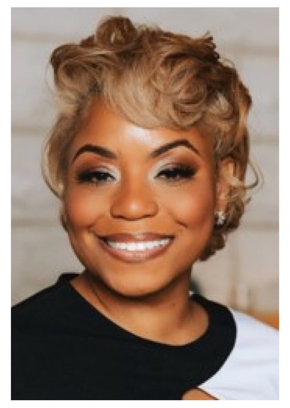
There are some potential topics and areas to expand the Parallel Programming and Algorithms course:

1. Advanced Parallel Algorithms: Cover more advanced or specialized parallel algorithms, such as parallel sorting algorithms (e.g., Bitonic sort, Parallel merge sort), Parallel Matrix Operations (e.g., matrix multiplication, LU decomposition)
2. Integrate Parallel Algorithms into Machine Learning, Cryptography and other fields (e.g., use GPU programming to speed up password cracking)

Authors

Rui Zhu
Dept. of Computer Science
Kettering University
rzhu@kettering.edu

HPC/Gateways Mentor
John Holmen
Oak Ridge National Laboratory
holmenjk@ornl.gov



HPC/Gateways Mentor
Yvonne Phillips
Morehouse College
Yvonne.Phillips@morehouse.edu

MORE INFORMATION → https://hackhpc.github.io/facultyhack-gateways23