Syllabus
Course Program

Parallel Computing on CPU/GPU/CUDA

Specialty
113 – Applied Mathematics

Institute
Institute of Computer Modeling, Applied Physics and Mathematics

Educational program
Computer and Mathematical Modeling

Department
Mathematical Modeling and Intelligent Computing in Engineering (161)

Level of education
Master's level (1 year 4 months)

Course type
Special (professional), Elective

Semester
1

Language of instruction
English

Lecturers and course developers

Oleksii Vodka (responsible lecturer)

Oleksii.Vodka@khpi.edu.ua
PhD, docent, Head of Mathematical modelling and intellectual computing in engineering department

General information, number of publications, main courses, etc.

More about the lecturer on the department's website

Ruslan Babudzhan (assistant)

Ruslan.Babudzhan@infiz.khpi.edu.ua
Assistant lecturer

More about the lecturer on the department's website

General information

Summary

Students will study the theoretical foundations of parallel computing systems and get initial experience in developing software that implements parallel computing. Lab sessions will provide initial experience building software systems with parallel computing on general-memory, distributed-memory, and GPU platforms. They will develop and research parallel systems of systems of various purposes using the C/C++ programming language and the OpenMP, MPI, and CUDA libraries.

Course objectives and goals

This course is aimed at developing knowledge of the structure of multiprocessor systems with an emphasis on the development of scientific software and numerical algorithms to improve the performance of application programs. The course covers multiprocessor systems with shared and

distributed memory, as well as techniques for programming graphics video accelerators to increase the speed of computing on conventional processors.

Format of classes

Lectures, laboratory classes, consultations, self-study. Final control in the form of an test.

Competencies

GC1. Ability to generate new ideas (creativity) and non- standard approaches to their implementation.
GC7. Ability to think abstractly, analyse and synthesise.
PC1. Ability to solve tasks and problems that can be formalised, require updating and integrating knowledge, in particular in conditions of incomplete information.
PC2. Ability to conduct scientific research aimed to develop new and adapt existing mathematical and computer models to study various processes, phenomena and systems, conduct appropriate experiments and analyse the results.
PC3. Ability to develop methods and algorithms for the
construction, research and software implementation of mathematical models in engineering, physics, biology, medicine and other fields and to analyse them.
PC7. Ability to design and develop software to solve formalised problems, including systems with large amounts of data.

Learning outcomes

LO4. Build mathematical models of complex systems and choose methods of their research, implement the built models in software and check their adequacy using computer technologies.
LO5. Justify and, if necessary, develop new algorithms and software tools for solving scientific and applied problems, apply, modify and investigate analytical and computational methods for solving them.
LO8. Develop and implement algorithms for solving applied problems, system and application software of information systems and technologies.

Student workload

The total volume of the course is 90 hours (3 ECTS credits): lectures - 16 hours, laboratory classes - 16 hours, self-study - 58 hours.

Course prerequisites

Basic programming skill with C/C++ language, knowledge in computation method and algorithm.

Features of the course, teaching and learning methods, and technologies

Lectures and practical classes, which includes problem discussion.

Program of the course

Topics of the lectures

Topic 1. Introduction to parallel computing
General information, classification of high-speed computing systems, goals, tasks and problems in creating parallel programs, mathematical regularities of developing multithreaded programs. (2 hours)
Topic 2. Standard C++ library for multithreading
C++ programming language in the development of parallel programs. Creating threads, their synchronization, the problem of race condition, atomic operations and types. (4 hours).
Topic 3. OpenMP programming standard
Compilation of OpenMP programs, syntax of parallel regions, setting the number of threads, an example of a parallel program for multiplying a matrix by a vector, features of memory allocation in parallel programs. Integration by the rectangular method parallel version, analysis of its performance, state race, ways to fix it and its impact on performance (critical sections, atomic operations). Parallelization of loops, algorithms for distributing loop tasks among threads. An example of determining prime numbers and its

parallelization, the impact of task distribution algorithms on the overall performance of the program. Data locality, the impact of locality on cache operation and program performance. An example of matrix multiplication, the impact of the algorithm's sequence of actions on the cache state and performance. Nested parallel regions and recursive parallel algorithms. Limitations of depth. Parallel traversal of a binary tree and its sum. Parallelism of tasks. Multithreaded quicksort algorithm. (8 hours)

Topic 4. MPI programming standard

Principles and examples of building systems with distributed memory. Supercomputers. Compiling and running MPI programs. Implementation of prime number search using MPI. Problems of performance and load balancing between threads. Communication between program nodes. Algorithms and functions of message exchange, the problem of mutual blocking of threads, obtaining information about messages. Blocking and non-blocking functions of sending messages. Data reduction, reduction operations. Transferring custom data types. Combining different data into one message. Examples: Image Contrast, Game of Life, Neighborhood Search, Process Topology. Performance analysis. (8 hours)

Topic 5. CUDA technology.

Features of GPU computing. Compiling and running programs. Example: adding vectors. Memory allocation and data transfer from/to the GPU. Features of processors with SIMT (Single-Instruction, Multiple-Thread) architecture. Example: Calculating a histogram of an image on the GPU. Atomic functions and libraries of CUDA. Overview of standard libraries for GPU computing. Caching data in the GPU. Example: solving the problem of interaction of N-bodies. (8 hours)

Topic 6. OpenACC technology.

Basic ideas and principles. Comparison with OpenMP. Writing simple programs for calculating actions on vectors. Tools for debugging and profiling programs. Examples (2 hours)

Topics of the workshops

Not appliable

Topics of the laboratory classes

Class 1. Creating pure C++ multithreaded program (4 hours)

Creating simple computation program. Working with lists in multithread way. Mutexes and synchronization.

Class 2. Writing OpenMP programs (4 hours)

Creating parallel program, which is calculate defined integrals using rectangle and trapezoid method. Speed-up estimation.

Class 3. Writing MPI program (4 hours).

Creating distributed program, which calculate PI number by Mote-Carlo method. Estimation of speed-up and scalability.

Class 4. Writing CUDA program (4 hours).

Creating GPU-accelerated program for matrix multiplication. Setting-up the environment. Estimation of speed-up and scalability.

Self-study

Topics for self-study:

1. Boost library and multithreading
2. Intel threading building blocks library
3. New features in OpenMP library
4. Library, which implement MPI: MPICH, OpenMPI, Microsoft MPI, HP MPI, Intel MPI, MPJ etc
5. OpenCL and its application
6. Cuda libraries for Fourier transform, Matrix multiplication etc

Also self-study includes preparation to laboratory classes and report preparation on passed laboratory classes

Course materials and recommended reading

1. Pacheco, P., & Malensek, M. (2021). An Introduction to Parallel Programming. Morgan Kaufmann.

2. Pacheco, P. (1997). Parallel programming with MPI. Morgan Kaufmann.
3. Chandra, R., Dagum, L., Kohr, D., Menon, R., Maydan, D., & McDonald, J. (2001). Parallel programming in OpenMP. Morgan kaufmann.
4. Schmidt, B., Gonzalez-Dominguez, J., Hundt, C., & Schlarb, M. (2017). Parallel programming: concepts and practice. Morgan Kaufmann.
5. Tay, R. (2013). OpenCL parallel programming development cookbook. Birmingham, UK: Packt Publishing.
6. Storti, D., & Yurtoglu, M. (2015). CUDA for engineers: an introduction to high-performance parallel computing. Addison-Wesley Professional.
7. Kaeli, D. R., Mistry, P., Schaa, D., & Zhang, D. P. (2015). Heterogeneous computing with OpenCL 2.0. Morgan Kaufmann.
8. Banger, R., & Bhattacharyya, K. (2013). OpenCL programming by example. Packt Publishing Ltd.

Assessment and grading

Criteria for assessment of student performance, and the final score structure

Total mark (100 points) consists of two parts:
1. Test on theory (60 points)
2. Practice (lab) passing (40 points)

The assessment of independent work is carried out by asking questions on relevant topics during the defense of laboratory work and is an integral part of the assessment of practical work.

Grading scale

Total points	National	ECTS
90–100	Excellent	A
82–89	Good	B
75–81	Good	C
64–74	Satisfactory	D
60–63	Satisfactory	E
35–59	Unsatisfactory (requires additional learning)	FX
1–34	Unsatisfactory (requires repetition of the course)	F

Norms of academic integrity and course policy

The student must adhere to the Code of Ethics of Academic Relations and Integrity of NTU "KhPI": to demonstrate discipline, good manners, kindness, honesty, and responsibility. Conflict situations should be openly discussed in academic groups with a lecturer, and if it is impossible to resolve the conflict, they should be brought to the attention of the Institute's management.
Regulatory and legal documents related to the implementation of the principles of academic integrity at NTU "KhPI" are available on the website: http://blogs.kpi.kharkov.ua/v2/nv/akademichna-dobrochesnist/

Approval

Approved by	Date August 30, 2023	Head of the department Oleksii VODKA
	Date August 30, 2023	Guarantor of the educational and professional program (1 year 4 months) Oleksiy LARIN