I033	**Parallel Programming**	L	P	S	ECTS
		2	2	0	6

Course objectives. Students will be introduced to the latest ideas of parallel programming. Parallel computer models and parallel programming paradigms are presented. The use of technology for developing parallel algorithms in various computing environments is described. Particular attention is given to the possibility of developing modular parallel programs for more complex parallel system.

Prerequisites Undergraduate study programme in mathematics or computer science.

Course content.

1. Parallel computer models. Parallel programming paradigms. Properties of parallel programs. Sequential to parallel program conversion.
2. MPI - Message Passing Interface standard
3. Synchronous shared memory parallel computer model (PRAM).
4. Sum of prefixes algorithm.
5. Asynchronous parallel computer (APRAM). Model and program complexity.
6. Designing parallel algorithms. Design phases.
7. Algorithm partitioning. Communication structure definition.
8. Task agglomeration. Task to processor mapping.
9. Quantitative analysis of parallel algorithms. Algorithm performance definitions.
10. Parallel algorithm scalability analysis.
11. Development of modular parallel programs. Modular development support in MPI.
12. Parallel computation on graphical chip. Data parallelism.
13. CUDA program structure, kernel functions and threads, synchronization and scalability.

LEARNING OUTCOMES

No.	LEARNING OUTCOMES
1.	Describe parallel computation and parallel programming models.
2.	Describe PRAM computer model.
3.	Apply PRAM programming model in parallel programming.
4.	Apply MPI technology in parallel program development.
5.	Recognize phases of parallel algorithm design.
6.	Combine parallel algorithm development elements.
7.	Evaluate efficiency and scalability of parallel algorithms.

RELATING THE LEARNING OUTCOMES, ORGANIZATION OF THE EDUCATIONAL PROCESS AND ASSESSMENT OF THE LEARNING OUTCOMES

TEACHING ACTIVITY	ECTS	LEARNING OUTCOME **	STUDENT ACTIVITY*	EVALUATION METHOD	POINTS	
					min	max
Attending lectures and exercises	1	1-7	Lecture attendance, discussion, teamwork and independent work on given tasks	Attendance lists, tracking activities	0	4
Homework Assignments	1	1-7	Solving programming tasks independently	Evaluating solutions	0	4

Written exam (Mid-terms)	2	1-7	Preparing for written exam	Evaluation	25	46
Final exam	2	1-7	Revision	Oral exam	25	46
TOTAL	6				50	100

Teaching methods and student assessment. Lectures and exercises are obligatory. The exam consists of a written and an oral part. Upon completion of the course, students can take the exam. Successful midterm exam scores replace the written exam. Exercises are performed laboratory using computers. Students can influence their final grade by doing homework assignments in which they will apply the knowledge acquire in lectures and exercises.

Can the course be taught in English: Yes

Basic literature:

1. A. Grama, A. Gupta, G. Karypis, V. Kumar, Introduction to parallel computing, Addison – Wesley, 2002.
2. D. B. Kirk, W. W. Hwu, Programming Massively Parallel Processors - A Hands-on Approach, Morgan Kaufmann, 2013.
3. Materials from lectures and exercises.

Recommended literature:

1. B. Parhami, Introduction to Parallel Processing, Algorithms and Architectures, Kluwer academic publishers, 2002.
2. I. Foster, Designing and Building Parallel Programs, Addison – Wesley, 1995.
3. J. Sanders, E. Kandrot, CUDA by Example - An Introduction to General-Purpose GPU Programming, Addison-Wesley, 2011.