COURSE DESCRIPTION

Parallel and Distributed Programming

Academic year 2026/2027

1. Programme-related data

1.1. Higher Education Institution	Babes-Bolyai University
1.2. Faculty	Faculty of Mathematics and Computer Science
1.3. Department	Department of Computer Science
1.4. Field	Computer Science
1.5. Level of study	Bachelor
1.6. Degree programme / Qualification	Computer Science
1.7. Form of education	Full Time

2. Course-related data

2.1. Course title	**Parallel and Distributed Programming**			Course code	**MLE5077**
2.2. Course coordinator	Assoc. prof Niculescu Virginia				
2.3. Seminar coordinator	Assoc. prof Niculescu Virginia				
2.4. Year of study	3	2.5. Semester	5	2.6. Type of assessment	Exam
2.7. Course status	Compulsory			2.8. Course type	Core subject

3. Total estimated time (hours per semester of teaching activities)

3.1. Number of hours per week	**5**	of which: 3.2. course	**2**	3.3. seminar/ laboratory/ project	**3**
3.4. Total of hours in the curriculum	70	of which: 3.5. course	28	3.6. seminar/ laboratory	**42**
Time allocation for individual study (IS) and self-taught activities (ST)					**hours**
Learning from textbooks, course materials, bibliography, and notes (IS)					5
Additional research in the library, on subject-specific electronic platforms, and on-site					10
Preparing seminars/ laboratories/ projects, assignments, reports, portfolios, and essays					30
Tutoring (professional guidance)					5
Examinations					5
Other activities [i.e.: two-way communication with the course coordinator/tutor]					0
3.7. Total hours of individual study (IS) and self-taught activities (ST)				**55**	
3.8. Total hours per semester				**125**	
3.9. Number of credits				**5**	

4. Prerequisites (where applicable)

4.1. curriculum-related	Object Oriented Programming, Advanced Programming Methods, Operating Systems, Systems for programming and design
4.2 skills-related	Programming in C++ and Java

5. Specific conditions (where applicable)

5.1. course-related	Lecture room with videoprojector
5.2. seminar/laboratory-related	Room with computers and videoprojector;

6.1. Competencies resulting from the completion of the degree programme *(as referred to in the curriculum)*[1]

[1] The professional and/or transversal skills targeted by the subject for which the course description is prepared will be copied from the curriculum of the degree programme. For each competency, the complete entry, including the competency code, will be copied with the exact wording that appears in the curriculum, without any changes. If no competency is copied from either of the two categories, the row corresponding to that category is deleted from the table.

Professional competencies	
Competency code	**Competency**
PC1	create software
PC5	define technical requirements
PC1	use software frameworks
PC11	use software design patterns
Transversal competencies	
Competency code	**Competency**
TC1	Work independently
TC2	Solves problems
TC3	Think analytical

6.2. Learning outcomes relevant to the degree programme (as referred to in the curriculum)[2]

Learning outcomes targeted by the subject		
Competency code	**Knowledge and comprehension**	**Specific academic skills**
PC1 **PC6**	1. The student/graduate identifies, explains and justifies fundamental concepts of data structures, algorithms, and programming paradigms, as well as computer architecture.	1. The student/graduate designs, develops and demonstrates complex software solutions using efficient algorithms and diverse programming paradigms
PC9 **PC10**	2. The student/graduate selects, describes, analyzes and explains modern programming paradigms, including functional, object-oriented and parallel programming, using current languages and frameworks.	2. The student/graduate designs, plans, builds, develops scalable software applications, and efficiently uses hardware and software resources.

7. Subject-specific learning outcomes

Knowledge and comprehension
1. The student/graduate understands the issues encountered by parallel and distributed applications
2. The student/graduate knows the basic mechanisms and frameworks used for building parallel and distributed applications
3. The student/graduate knows the patterns of developing parallel and distributed applications.
Specific academic skills
1. The student/graduate is able to design parallel algorithms and applications
2. The student/graduate understands and is able to find solutions to the parallelization issues
3. The student/graduate is able to assess the performance improvements obtained by parallelization

8. Contents

[2] The learning outcomes relevant for the degree programme and targeted by the subject for which the course description is prepared will be listed. The entries, copied without any changes from the Curriculum by subject type (Core Subject/Specialisation Subject/Complementary Subject), are listed under the corresponding competency.

8.1. Course	Teaching and learning methods	Remarks[3]
C1. General introduction to parallel and distributed programming	Exposure, description, explanation, examples, debate.	
C2. Parallel architectures – Taxonomies (Flynn)	Exposure, description, explanation, examples, debate.	
C3. Types and models of parallelism	Exposure, description, explanation, examples, debate.	
C4. Parallel programming in distributed memory environments (MPI)	Exposure, description, explanation, examples, debate.	
C5. Parallel programming in shared memory environments	Exposure, description, explanation, examples, debate.	
C6. Concurrency – concepts Synchronization: monitors, semaphores, conditional variables	Exposure, description, explanation, examples, debate.	
C7. Concurrency – patterns Producer-Consumer	Exposure, description, explanation, examples, debate.	
C8. Parallel programming in shared memory environments - OpenMP	Exposure, description, explanation, examples, debate.	
C8. Asynchronous parallel programming	Exposure, description, explanation, examples, debate.	
C9. Performance evaluation of parallel programs:	Exposure, description, explanation, examples, debate.	
C10. Data parallel programming: CUDA programming model	Exposure, description, explanation, examples, debate.	
C11. Design patterns for parallel programming Master-slaves; Task-Farm/Work-Pool; Divide & Conquer	Exposure, description, explanation, examples, debate.	
C12. Design patterns for parallel programming Divide & Conquer; Pipeline	Exposure, description, explanation, examples, debate.	
C13. Design patterns for distributed programming	Exposure, description, explanation, examples, debate.	
C14. General comparative analysis of the new concepts, techniques, principles, patterns introduced	Exposure, description, explanation, examples, debate.	

Bibliography

1. Ian Foster. Designing and Building Parallel Programs, Addison-Wesley 1995.
2. Michael McCool, Arch Robinson, James Reinders, Structured Parallel Programming: Patterns for Efficient Computation, Morgan Kaufmann, 2012.
3. F. Buschmann, K. Henney, D. C. Schmidt. Pattern-Oriented Software Architecture Volume 4: A Pattern Language for
4. Grama, A. Gupta, G. Karypis, V. Kumar. Introduction to Parallel Computing, Addison Wesley, 2003.
5. D. Grigoras. Calculul Paralel. De la sisteme la programarea aplicatiilor. Computer Libris Agora, 2000.
6. B. L. Massingill, T.G. Mattson, and B. A. Sanders, A Pattern Language for Parallel Programming. Wesley Software Patterns Series, 2004.
7. V. Niculescu. Calcul Paralel. Proiectare si dezvoltare formala a programelor paralele. Presa Univ. Clujana, 2006.
8. M. Richards. Software Architecture Patterns. Understanding Common Architecture. Patterns and When to Use Them 2015 O'Reilly Media.
9. D. Schmidt, M. , H. Rohnert, F. Buschmann. Pattern-Oriented Software Architecture Volume 2: Patterns for Concurrent and Networked Objects Volume 2. Wiley, 2000.
10. B. Wilkinson, M. Allen, Parallel Programming Techniques and Applications Using Networked Workstations and Parallel Computers, Prentice Hall, 2002

***, Tutorials OpenMP
***, Tutorials MPI
***, Tutorials CUDA

8.2. Seminar/ laboratory	Teaching and learning methods	Remarks

[3] For example, organisational aspects, recommendations for students, specific aspects relating to the course/seminar, such as inviting experts in the field, etc.

L1 –L4 . Multithreading programming in C++ si Java	Dialogue, debate, examples, guided discovery.	
L5-L6 Distributed programming using MPI	Dialogue, debate, examples, guided discovery.	
L7- L9. Concurrency – conditioners, syncronizations	Dialogue, debate, examples, guided discovery.	
L10. Programaring using OpenMP	Dialogue, debate, examples, guided discovery.	
L11, Project CUDA	Dialogue, debate, examples, guided discovery.	
L12- L14 Complex project Client-Server	Dialogue, debate, examples, guided discovery.	

Bibliography

1. Eckel, B., Thinking in Java, 4th Edition, New York: Prentice Hall, 2006.
2. Fowler, M., Patterns of Enterprise Application Architecture, Addison-Wesley, 2002.
3. J. Sanders, E. Kandrot. CUDA by Example. An Introduction to General-Purpose GPU Programming. Addison-Wesley. 2010.
4. A. WILLIAMS. C++ Concurrency in Action. PRACTICAL MULTITHREADING. MANNING, 2012.
5. ***, Tutoriale Java http://download.oracle.com/javase/tutorial/
6. ***, Tutorial C++ [https://en.cppreference.com/]
7. ***, OpenMP[http://openmp.org/wp/]

***, MPI[http://www.mpi-forum.org/]

9. Evaluation

Type of activity	9.1 Evaluation criteria[4]	9.2 Evaluation methods[5]	9.3 Percentage in the final grade
9.4. Course	Knowledge of basic concepts and mechanisms	Theoretical exam	40%
9.5. Seminar/ laboratory	Using the concepts and mechanisms introduced in the course to solve concrete problems	Laboratory programs -Evaluation of the quality and completeness of the received assignments -Evaluation of the understanding of the application of the concepts and mechanisms used	40%
	Complex project	Presentation and verification of the implementation, and associated documentation	20%
9.6 Minimum standard for passing			
The final average must be at least 5 (on a scale from 1 to 10). Passing is conditional on achieving a minimum grade of 5 in the theoretical exam.			

10. SDG labels (Sustainable Development Goals)[6]

[4] The evaluation criteria must directly reflect the learning outcomes targeted at the level of the degree programme respectively at the level of the subject. More specifically, the learning outcomes set out in the expected learning outcomes are assessed.

[5] Both final evaluation methods and ongoing evaluation strategies should be established.

[6] Select a single label which, according to the *Implementation of SDG labels in the academic process*, best matches the subject. If the subject addresses sustainable development in a generic manner (i.e. by presenting/introducing

	○	Sustainable Development Generic Label

								
○	○	○	○	○	○	○	○	x
								No label applies
○	○	○	○	○	○	○	○	○

Date of entry:
10.05.2026

Signature of course coordinator

Virginia Niculescu

Signature of seminar coordinator

Virginia Niculescu

Date of approval in the department:
...

Signature of the head of department

.....

the general framework of sustainable development, etc.), then the Sustainable Development generic label may be applied. If none of the labels describe the subject, select the last option: *"No label applies."*