unibz Freie Universität Bozen
Libera Università di Bolzano
Università Liedia de Bulsan

UNIVERSITY OF OULU

POLITÉCNICA

COURSE DESCRIPTION – ACADEMIC YEAR 2020/2021

Course title	**Introduction to Parallel Computing**
Course code	76085
Scientific sector	INF/01
Degree	Master in Software Engineering for Information Systems (LM-18)
Semester	1
Year	1
Credits	6
Modular	No

Total lecturing hours	20
Total exercise hours	40
Attendance	Attendance is not compulsory, but strongly recommended. Students who are unable to follow all lectures and labs are encouraged to interact with the lecturer.
Prerequisites	Good knowledge of the following subjects is expected: • Programming • Computer Systems • Algorithms and Data Structures
Course page	https://ole.unibz.it/

Specific educational objectives	The course belongs to the type caratterizzanti – discipline informatiche – Specialization topics Students will acquire a deep knowledge of how to design faster and efficient applications by exploiting modern parallel architectures (e.g., GPUs). Under such a light, students will acquire professional skills and knowledge in parallel computing by understanding the most advanced techniques that researchers have developed in the last years.

Lecturer	Flavio Vella
Contact LA	Flavio.Vella@unibz.it Piazza Domenicani, 3 – Office POS 3.13
Scientific sector of lecturer	INF/01
Teaching language	English
Office hours	Friday 10:00 - 12:00. Extra office hours will be offered by arranging a meeting by email.
Lecturing Assistant (if any)	--
Contact LA	--
Office hours LA	--
List of topics	• Introduction to parallel and distributed systems • Shared memory model • Distributed memory model • Principle and design of parallel algorithms • Selection of parallel algorithms • Performance Analysis, optimization and tuning
Teaching format	• Frontal lectures • Lab supported by the lecturer and Teaching Assistant (if any)

	In the lectures, new concepts and techniques are introduced by presentation on the blackboard and multimedia material (slides and videos). In the labs, students will: 1. use the tools that will be used during the course and project development; 2. solve simple exercise and discuss the solution; 3. start to prepare the final project; supported by the lecturer or Teaching Assistant (if any).

Learning outcomes	**Knowledge and understanding:** D1.2 To be able to analyze and solve even complex problems in the area of Software Engineering for Information Systems with particular emphasis on the use of studies, methods, techniques and technologies of empirical evaluation; D1.3 To know in depth the scientific method of investigation applied to complex systems and innovative technologies that support information technology and its applications; D1.8 To be able to read and understand specialist scientific documentation, such as conference proceedings, articles in scientific journals, technical manuals. **Applying knowledge and understanding:** D2.1 To know how to apply the fundamentals of empirical analysis of ICT data to the construction of mathematical models for the evaluation and prediction of characteristics of applications and software systems; D2.2 To be able to design and perform experimental analyses of information systems in order to acquire measures related to their behaviour and to evaluate experimental hypotheses in different fields of application, such as business, industrial or research; D2.5 To be able to extend and modify in an original way an existing technical solution or a formal model taking into account changed conditions, requirements and evolution of the technology. **Making judgments:** D3.1 To be able to autonomously select documentation from a variety of sources, including technical books, digital libraries, technical scientific journals, web portals or open source software and hardware tools; D3.4 To be able to reconcile the objectives of the project that are in conflict, to trade-off cost, resources, time, knowledge or risk; D3.5 To be able to work with large autonomy, also assuming responsibility for projects and structures. **Communication skills:** D4.2 To be able to present the contents of a scientific/technical report to an audience, including non-specialists, at a fixed time;

	D4.3 To be able to structure and draft scientific and technical documentation describing project activities; D4.4 To be able to coordinate project teams and to identify activities to achieve project objectives; D4.5 To be able to prepare and conduct technical presentations in English; D4.7 To be able to carry out research and projects in collaborative manner; D4.8 To be able to synthesise knowledge gained from reading and studying scientific documentation. **Learning skills:** D5.1 To be able to independently extend the knowledge acquired during the course of study by reading and understanding scientific and technical documentation in English; D5.3 In the context of a problem solving activity, to be able to extend knowledge, even if incomplete, taking into account the final objective of the project; D5.4 To be able to formulate and validate theories and define new methods through empirical induction and new generation scientific investigation tools.

Assessment	The assessment is based on a final project that will be assigned during the semester. It will consist of solving a particular problem by using parallel computing techniques. The project will be developed by a group of two students (at most). Specifically, the team have to: • release the code by providing the instruction for result reproducibility; • write a short scientific document describing the solution, the methodology, the technology they adopted for solving the problem; • prepare a short oral presentation of the project. After the presentation will follow a Q&A session to assess the knowledge of the candidate and its contribution to the project. The assessment is based on the **individual** contribution of each team member.
Assessment language	English
Assessment typology	Monocratic
Evaluation criteria and criteria for awarding marks	The final mark is composed by evaluating the project in terms of originality of the methods adopted/designed, results obtained and quality and clarity of the presentation which includes code, document and oral presentation. Specifically,

	• Code assessment (20% of the final mark). The software should follow the best practice for code writing. The experiments must be replicable. • Document assessment (40% of the final mark). Ability to introduce a problem. Ability to report the state of the art. Ability to describe the methodology adopted. Ability to comment and report the results obtained. The originality and the soundness of the solution will be also considered in the evaluation. • Quality and clarity of the oral presentation in a fixed time (40% of the final mark). Ability to answer to possible questions to the project and the related topics addressed during the course that can arise during the presentation.

Required readings	There is not a single book that cover all the topics that will be presented during the course. • Introduction to parallel computing 2nd edition (Grama, Karpis, Kumar, Gupta) • Computer Architecture: a quantitative approach. 6th ed (Hennessy, Patterson) • The Art of Multiprocessor Programming (Herlihy, Shavit) • Programming Massively Parallel Processors: A Hands-on Approach 3rd Edition (Kirk, Hwu) Subject Librarian: David Gebhardi, David.Gebhardi@unibz.it
Supplementary readings	• CUDA C++ Best Practices Guide and CUDA C++ Programming Guide; • C++ for OpenCL Programming Language; • CUDA by examples (Sanders and Kandrot). • Patterns for parallel programming (Timothy G. Mattson) • Parallel Computer Architecture. A Hardware / Software Approach (David Culler)
Software used	Programming languages: C/C++ or Python. Compilers GCC and NVCC Other software/frameworks: CUDAToolkit, OpenMPI or OpenCL. Github.