DEREE COLLEGE SYLLABUS FOR: **ITC 4458 HIGH PERFORMANCE COMPUTING** (Updated Fall 2025)	**3/0/3** **UK LEVEL: 6** **UK CREDITS: 15**
PREREQUISITES:	ITC 2088 Introduction to Programming ITC 2086 Computer Systems Architecture ITC 2093 Operating Systems Concepts ITC 3006 Mathematics for Computing
COREQUISITES:	None.
CATALOG DESCRIPTION:	Big data challenges; multi-core programming; shared and distributed memory; concurrency models; synchronization and coordination; distributed algorithms and frameworks; GPU programming
RATIONALE:	The course aims to bridge the big gap between traditional programming for serial machines and programming for multi- or many-core machines and large clusters. Students have the opportunity to learn and practice multiprocessor programming along with models and tools for building high-performance applications, and thus develop skills to tackle the challenges associated with the big data world.
LEARNING OUTCOMES:	As a result of taking this course, the student should be able to: 1. Demonstrate understanding of the HPC laws, models and architectures. 2. Critically assess basic patterns for problem decomposition 3. Explain how algorithms can be parallelized. 4. Apply concepts and techniques of programming shared-memory multi-core and cluster computers. 5. Build and evaluate framework-based systems that utilize hybrid shared/distributed memory computer clusters.
METHOD OF TEACHING AND LEARNING:	In congruence with the teaching and learning strategy of the college, the following tools are used: • Lectures, laboratory sessions, and use of generative AI tools to inform course content • Office hours held by the instructor to provide further assistance to students. • Use of the online content management system (Blackboard CMS) to further facilitate communication.
ASSESSMENT:	**Summative:**

1st assessment: Midterm exam Short answers and/or case problems	**30%**
2nd assessment: Portfolio of student work and oral assessment	**10%**
Final assessment: Project High performance framework-based implementation	**60%**

Formative:

Take-home short problems	**0%**

The formative assessments aim to prepare students for the summative

	assessments and expose them to teamwork. The 1st summative assessment tests the LOs 1, 2 and 3. The 2nd summative assessment tests the LOs 1-5. The final summative assessment tests the LOs 1-5. *The final grade for this module will be determined by averaging all summative assessment grades, based on predetermined weights for each assessment. If students pass the **final summative assessment,** which tests all Learning Outcomes for this module, and the average grade for the module is 40 or above, students are not required to resit any failed assessments.*
INDICATIVE READING:	**REQUIRED READING:** 1. M. Herlihy et al., “The Art of Multi-Processor Programming”, 2nd ed. Morgan-Kaufmann, 2021. 2. Instructor’s notes. **RECOMMENDED READING:** 1. M. Zaharia, “An Architecture for Fast and General Data Processing on Large Clusters”, ACM Books, 2016. 2. T. Mattson et al. “Patterns for Parallel Programming”, Addison-Wesley, 2013. 3. A. Kaminsky, “Big CPU, Big Data”, CreateSpace, 2016. *Additional recommended readings list available through Blackboard.*
INDICATIVE MATERIAL: *(e.g. audiovisual, digital material, etc.)*	**REQUIRED MATERIAL**: N/A **RECOMMENDED MATERIAL**: MIT Video Lectures on Parallel Computing on MIT OpenCourseWare: Parallel Computing | Mathematics | MIT OpenCourseWare
COMMUNICATION REQUIREMENTS:	Daily access to the course’s site on the College’s Blackboard CMS and the acg email. Effective communication using proper written and oral English. Use of word processing and/or presentations software for documentation and presentation of deliverables and the final project.
SOFTWARE REQUIREMENTS:	MS Office JDK8+ Apache Spark OpenMPI on a cluster of 2+ nodes CUDA NVIDIA GPU Computing Toolkit
WWW RESOURCES:	• https://www.open-mpi.org/ • https://www.mpich.org/ • https://research.cs.wisc.edu/htcondor/ • https://en.wikipedia.org/wiki/Cilk • https://github.com/ioannischristou/popt4jlib
INDICATIVE CONTENT:	1. HPC Hardware Models and Architectures 2. Parallel Computing Bounds: Amdahl’s Law, Brent’s Theorem etc. 3. Software Concurrency Models: Processes and Threads 4. Synchronization and Coordination Primitives 5. Language Memory Models for Shared-Memory Multi-Processors

	6. Multi-threaded Programming 7. Parallel Algorithms 8. Communication Primitives for Distributed-Memory Clusters 9. Fundamentals of Distributed Algorithms 10. Distributed Computing Frameworks: OpenMPI, Spark, Celery 11. GPU Programming with CUDA