BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, Pilani
Pilani Campus
AUGS/ AGSR Division

SECOND SEMESTER 2020-21
COURSE HANDOUT

Date: 18.01.2021

In addition to part-I (General Handout for all courses appended to the time table) this portion gives further specific details regarding the course

Course No. : **CS F422**
Course Title **:** **Parallel Computing**

Instructor-in-Charge**: Hari Babu K (email: khari)**

Course Website: https://canvas.instructure.com/courses/2517982

1. Scope and Objective:

This course is an introduction to techniques for development of parallel software systems and the requisite foundations. The scope includes an overview of underlying systems architectures and program design principles but the primary coverage is on software development methodologies with emphasis on development for commodity distributed systems: clusters of multi-core workstations. The course objective is to enable students to understand and leverage commonly available forms of parallelism at the architecture level (e.g. pipelining, multiple cores, clusters) and at the systems level (i.e. shared memory and threads or message passing and processes) to build parallel systems and applications with predictable performance.

2. (a) Text Book:

T1. Ananth Grama, Anshul Gupta, George Karypis & Vipin Kumar *Introduction to Parallel Computing*, Second Edition, Pearson Education, First Indian Reprint 2004.

2. (b) Reference Books:

R1. M.J. Quinn, *Parallel Computing : Theory & Practice*, McGraw Hill Inc. 2nd Edition 1994.
R2. Maurice Herlihy and Nir Shavit. *The Art of Multiprocessor Programming*. Morgan Kaufmann 2008.
R3. J. Hennessey and D. Patterson, *Computer Architecture: A Quantitative Approach*, Morgan Kaufmann, 5th Edition.
R4. Ian Foster, Designing and Building Parallel Programs. Online Publication by Addison Wesley. (available at Ian Foster's website at Argonne National Lab).
R5. Kai Hwang, Geoffrey C. Fox, Jack J. Dongarra. Distributed and Cloud Computing. From Parallel Processing to the Internet of Things. Morgan Kauffman (Elsevier) 2002.

(**AR**) Additional reading in the form of papers from journals / conference proceedings or articles from other sources will be prescribed by the instructor. See course website for specific reading per lecture / topic.

3. Course Plan:

a. Modules and Learning Outcomes

Module	Theme	Learning Outcomes (The student should be able to)	Prior Knowledge (assumed)
M1	Introduction to Parallel Processing, Parallel Computing Architectures, and Parallel Systems	understand and leverage architectural features (such as pipelining, multi-threading, and multiple processors) for developing parallel programs.	Computer Organization – Von Neumann Architecture, Memory Hierarchy and Caching, Locality.
M2	Parallel Algorithms and Programming	design parallel algorithms using conceptual models (e.g. PRAM)	Data Structures and Algorithms - Algorithm Design Techniques, Basic Sorting algorithms and Graph Algorithms.
M3	Performance Models, Metrics and Techniques	(1) understand performance characteristics of parallel algorithms / programs and the factors that affect them (2) analyze performance of parallel algorithms / programs (3) apply performance improvement techniques on parallel algorithms / programs	-
M4	Software Design for Parallel Systems	apply software design methodologies (e.g. Foster's variant of task-graph approach) to design parallel programs	Operating Systems -Tasks, Processes and Threads; Scheduling;
M5	Programming for Shared Memory Multi-Processors and Messaging Passing Systems	implement parallel algorithms on shared memory multi-processor systems, messaging processing systems, or hybrid systems	Data Structures– Linked Lists, Stacks/Queues, Hashing; Concurrency - Mutual Exclusion, Synchronization, and Deadlocks.

b. Lecture Schedule

Lectures	Module#	Topics	Readings

1	M1	Computational Models - Sequential vs. Concurrent Execution, Logical vs. Physical Concurrency.	-
2.a	M1	Physical Concurrency – Intra-Processor Concurrency.	-
2.b	M1	Intra-Processor Concurrency – Pipelining- Structure and Operation - Implementation Issues.	R3 (Appendix C:C1-C4)
3	M1	Intra-Processor Concurrency – Pipelining- Hazards. Instruction Level Parallelism. Performance Factors.	T1 (Sec 2.1.1); T1 (Sec. 3.6.5); R3 (Appendix C: C1-C4)
4.a	M1	Intra-Processor Concurrency - Superscalar Architectures. Implementation Issues. Instruction Level Parallelism. Performance Factors and Limitations.	R3 (Sec. 3.1-3.4, 3.7, 3.10)
4.b	M1	Intra-Processor Concurrency - VLIW Architectures - Performance Factors. Compilers for VLIW Architectures: Scheduling and Optimization.	AR
5	M1	Aside: Memory Hierarchy - Memory Bandwidth Requirements, Memory Hierarchy - Motivation, Locality of Reference, Inclusion Principle, Performance.	-
6.a	M1	Shared Memory Parallelism: Multi-threaded architectures.	AR
6.b	M1	Shared Memory Parallelism: Multi-core architectures: Introduction and Motivation	AR
7	M1	Shared Memory Parallelism - Multi-core architectures - A few examples - General Principles, Design Parameters and Components.	AR
8	M1	Shared Memory Parallelism – Multi-processor Architectures - Symmetric Multiprocessing / Uniform Memory Architectures vs. Distributed Shared Memory / NUMA, Caching and Cache Coherence in Shared Memory Systems.	R3 (Sec. 5.1-5.3)
9	M1	GPGPU architectures	AR
10.a	M1	Forms of Parallelism - Introduction: Forms of Parallelism - Different Characterizations (Synchronous vs. Asynchronous, Shared Memory vs. Message Passing, Custom-made vs. Commodity). Flynn’s Taxonomy - SISD, SIMD, MISD, MIMD;Software Level Version of Flynn's Taxonomy: SPSD, SPMD,MPSD, MPMD; Parallel vs. Distributed Systems.	R3 (Sec.1.2, Sec. 4.1-4.3);AR
10.b	M1	Spectrum of Parallelism - Instruction Level to Task Level to service Level. Abstractions for Sequential and Parallel Computing - Mapping of Tasks to Systems.	-

11	M1	Message Passing Model: - Abstractions for Sequential vs. Parallel Computing. Shared Memory Parallelism vs. Distributed Memory Parallelism. Introduction t0 the Messaging Passing Paradigm of Programming: SPMD vs. MPMD, Synchronous vs. Asynchronous Programming.	T1. (Sec. 6.1)
12.a	M1	Parallel / Distributed Computing: Clusters: Computing Model; Structure and Components: Characterizations	R5 (Sec. 2.1); AR
12.b	M1	Clusters: Cluster Interconnects – Examples and Performance Aspects.	R5 (Sec. 2.2); AR
13.a	M1	Clusters: Cluster Middleware - Single System Image - Features.	R5 (Sec. 2.3) AR
13.b	M1	Clusters: Cluster Middleware - Single System Image - Implementation Support in Hardware.	R5 (Sec. 2.3) AR
14	M1	Clusters: Cluster Middleware - ClusterOS: Single System Image - Implementation in OS Kernel or Gluing Layer: Resource Pooling and Provisioning - Scheduling, Load Balancing, and Process Migration – Examples Cluster Operating Systems	R5 (Sec. 2.4) AR
15.a	M1	Clusters: Cluster Middleware - Single System Image – Implementation Support in Application Layer / User Level. Case Study: GFS	R5 (Sec. 6.3.2)
15.b	M2	Message Passing Model: Communication Primitives (send and receive for Point-to-Point communication). Syntax and Semantic Issues.	T1. (Sec 6.2.1)
16.a	M2	Communication Primitives: Blocking Operations: Non-buffered Mode: Protocol - Different scenarios. Idling Overhead. Deadlock Scenarios.	T1 (Sec 6.2.1)
16.b	M2	Communication Primitives: Blocking Operations: Buffered Mode: Protocols: with and without Communication Hardware- Different scenarios- Idling Overhead. Deadlock Scenarios: Tagged Messaging.	T1 (Sec 6.2.1)
16.c	M2	Communication Primitives: Non-Blocking Operations.	T1 (Sec 6.2.2)
17	M2	The Message Passing Interface (MPI): Primitives for Message Passing - Blocking vs. Non-Blocking, Buffered Messages and User Level Buffering; Collective / Group Communication	AR
18.a	M2	Computational Models - Random Access Machine (RAM) Model and Parallel RAM Model	R1 (Sec 2.1-2.2)
18.b	M2	Computational Model: PRAM: Variants of PRAM - Concurrent Memory Access Models: EREW, CREW, CRCW. Simulation of Priority CRCW by EREW.	R1 (Sec 2.2)

18.c	M2	Computational Model: PRAM: Algorithms - Complexity and Cost.	R1 (Sec 2.2)
19.a	M2	PRAM Algorithms - Task Spawning Model and Cost.	R1 (Sec 2.3)
19.b	M2	PRAM Algorithms: Design Techniques: Parallel Reduction - Example: Summation: Algorithm and Analysis.	R1 (Sec 2.3)
19.c	M2	Parallel Reduction - Reduction Template. Speedup and Efficiency. Divide-and-Conquer – Data Parallelism.	R1 (Sec 2.3) & T1 (Sec 3.6.1)
20.a	M2	Aside: Declarative Programming: Functional Style Programming and List Operations: map and fold	
20.b	M2, M6	Parallel Programming: Parallel Iterations and Realizing map on PRAM. Parallel implementation of fold using the reduction template. The map-reduce paradigm. Examples.	AR
21	M2, M3	Computational Model - Complexity - Work vs. Cost, Number of operations, Cost-optimality, Number of processors, Brent's Theorem Examples.	R1 (Sec 2.4), T1 (Sec 5.5)
22	M3	Parallel Computing and Performance - Speedup : Amdahl's Law, Multi Core as a special case;	T1 (Sec 2.2, 5.1-5.2, 5.4,5.7)
23	M3	Parallel Computing and Performance - Multi-Threading and Latency Hiding; Non-linear Speedup; Scaled Speed-up(Gustafson's Law) and Iso-efficiency.	AR
24	M4	Parallel Systems: Program Structuring and Software Design. Software Architecture – Principles of Modularity: Cohesion and Coupling. Introduction to Foster's Design Methodology.	R4 (Sec 2.1), T1 (Sec 3.1), AR
25	M4	Foster's Design Methodology – Partitioning Phase - Basic Decomposition Strategies: Domain Decomposition and Task Decomposition - Examples.	R4 (Sec 2.2), T1 (Sec. 3.2)
26	M4	Design Methodology – Partitioning Phase - Further Examples - Recursive Decomposition, Exploratory Decomposition, Checklist for Partitioning.	R4 (Sec 2.2), T1 (Sec. 3.2)
27	M4	Foster's Design Methodology – Communication Phase - Different Forms of Communication: Local vs. Global, Structured vs. Unstructured, Static vs. Dynamic, Synchronous vs. Asynchronous Communication. Examples. Checklist for Communication.	R4 (Sec 2.3), T1 (Sec. 3.3)

28	M4	Foster's Design Methodology: Agglomeration – Heuristics, Granularity, Impact of Granularity on Performance, Granularity and Execution Environment. Replication Patterns and Examples, Checklist for Agglomeration.	R4 (Sec 2.4), T1 (Sec. 3.5, 5.3)
29	M4	Foster's Design Methodology: Mapping – Static and Dynamic Mapping; Strategies.	R4 (Sec 2.5), T1 (Sec. 5.4)
30	M4, M3	Scheduling and Load Balancing – Techniques, Issues, and Performance	AR
31	M4	Mapping, Interconnects, and Communication Overheads	R4 Sec. 3.7), T1 (Sec.2.4.2-2.4.5, 2.5, 3.5)
32	M4	Mapping: Techniques for commodity computing environments: Clusters of Multi-core Computers; Processes and Threads. Decomposition and Mapping. Implementation Issues. Granularity and levels of mapping.	T1 (Sec. 2.7), AR
33	M5	Designing with Threads. Thread based programs: Creation, Decomposition of Tasks to threads. Thread Cancellation.	T1 (Sec. 7.1, 7.7, 7.9)
34.a	M5	Designing with Threads – Shared Data and Shared memory – Issues. Memory Management issues.	AR
34.b	M5	Shared Memory Programming – Properties: Liveness and Safety. Mutual Exclusion.	R2 (Sec 1.1-1.3)
35.a	M5	Shared Memory Programming – Mutual Exclusion and Synchronization, Synchronization Primitives, Locking. Properties of Locking.	T1 (Sec. 7.5, 7.8); R2(Sec. 2.1-2.4)
35.b	M5	Mutual Exclusion: Locking: Locking and Fairness – Lamport's Algorithm; Implementation Issues and Efficiency.	R2 (Sec. 2.5-2.8)
36	M5	Locking: Locking in Multiprocessor Systems: Spin Locks vs. Blocking Locks; Test-and-Set Locks - Implementation Issues - Cache Coherence and Bus Contention.	R2 (Sec 7.1 - 7.3)
37	M5	Locking: Spin Locks with Backoffs and Spin Locks with Queues. Reentrant Locks. Reader-Writer Locks.	R2 (Sec. 7.4-7.5)
38	M4	Shared Memory Data Structures - Sequential Consistency, Real-Time Order, and Linearization. [Readings R2 Sec. 3.1 to 3.5]	R2 (Sec. 3.1-3.5)
39	M4	Shared Memory Data Structures: Locking Frameworks: Coarse-Grained vs. Fine-Grained Locking: Lock Coupling	R2 (Ch. 9), AR
40-42	M4, M5	Select Advanced Topics	AR

4. Evaluation

4a. Evaluation Scheme:

Component	Weight	Date	Remarks
Assignments (3)	30%	3 to 4 weeks	Take Home
Mid-Term Test (90 minutes)	25%	<TEST_1>	Partly Open Book
Take-Home Exercises	10%	TBA	Take Home
Comprehensive Exam (180 minutes)	35%	<TEST_C>	Partly Open Book

4b. Make-up Policy:

- No Make-up will be available for Assignments or Term Project under any condition.
- Prior Permission of the Instructor-in-Charge is usually required to get make-up for the mid-term test.
- Prior Permission of (Associate) Dean, Instruction is usually required to get make-up for the comprehensive exam.
- A make-up shall be granted only in genuine cases where - *in the Instructor's / Dean's judgment* - the student would be physically unable to appear for the test/exam. Instructor's / Dean's decision in this matter would be final.

4.c. Fairness Policy:

- Student teams are expected to work on their own on assignments / term project.
- All students are expected to contribute equally within a team.
- Individual contributions should be identified and documented in qualitative and quantitative terms by the team members. The instructor's assessment regarding the contributions of team members would be final.
- Any use of unfair means in quizzes, assignment or test/exam will be handled strictly. The minimum penalty would be loss of full weight of the component. Students involved in such activity are liable for further sanctions including being formally reported to the Unfair Means committee and being subject to penalties enabled by Unfair Means Rules of the Institute:
  - Unfair means would include copying from or enabling copying by other students; or copying / borrowing material from the Web or from other sources of information (where not permitted) including all electronic sources.
- Students are allowed to consult/discuss with other students/teams for the take-home assignments or term project but such consultation/discussion should be explicitly acknowledged and reported to the instructor prior to evaluation

5. **Consultation Hours:** See course website.
6. **Notices:** All notices concerning this course will be displayed on the course website only. Official email will be used to communicate.

Instructor –In- Charge
CS F422