ASSUMPTION UNIVERSITY

**VINCENT MARY SCHOOL OF SCIENCE AND TECHNOLOGY**

**MASTER OF SCIENCE IN COMPUTER SCIENCE**

**Course outline** (as of semester 1/2019)

**SC6201 (Advanced Computing Systems)**

**Course status:** Required Course

**Pre-requisites:** SC 5212 Computing Systems or any equivalent

**Class:**  Wednesday 18:30 – 21:30 (3 hrs) at A82

**Instructor:** Asst. Prof. Dr. Anilkumar K.G

**Office:** **E64** - Hua Mak Campus

 **Phone:**  0891351711 **Email:** anil@scitech.au.edu

**Course Description:**

Multi-core operating systems, distributed computation, distributed objects and middleware, distributed storage systems, failures and recovery, internet scale computing, cloud systems, virtualization and hypervisors, security in the cloud, and case studies of state-of-the-art solutions for cloud computing.

**Course Objective:**

Through this course it is expected that the students will gain the theoretical viewpoints of various design aspects of an advanced computer system including its CPU structure, Storage system, Hardware System, Operating System, hypervisors, cloud computing, distributed computing, system failure and recovery.

# Text Books (References):

1. Kai Hwang, Geoffrey C. Fox, and, Jack J. Dongarra, *Distributed and Cloud Computing From Parallel Processing to the Internet of Things*, Morgan Kaufmann Publishing, 2014.
2. Abraham Silberschatz, Peter B. Galvin, and Greg Gagne, *Operating System Concepts*, Wiley, 2013.
3. John L. Hennessy and David A. Patterson, *Computer Architecture: A Quantitative Approach*, Morgan Kaufmann Publishing, 2010.
4. Douglas E. Comer, *Internetworking with TCP/IP Principles, Protocol and Architecture*, Pearson, 6th edition, 2017.
5. Mukesh S and Niranjan S, *Advanced Concept in Operating Systems*, McGraw-Hill, 1994.

# Mark allocation:

Projects: 30%

 Quizzes 10%

 Midterm examination 20%

 Final examination 40%

**Lesson Schedule**

|  |  |
| --- | --- |
| **Week** | **Topics** |
| 1 | **Computer system:**  Operating System, Memory management, CPU structure, CPU management, I/O management, and Multi-core Systems. |
| 2 | **Virtual Machine:** Overview, Virtualization, Hypervisors**;** virtual building blocks,issues in virtualization,Para-virtualization, and Emulation. |
| 3 | **Virtualization:** Application Containment, Virtual CPU Scheduling, Memory management, I/O management, Live migration, VMware, and Java Virtual Machine. |
| 4 | **Internet Scale of Computing:** Overview of underlying network technologies, Internetworking Concept, Protocol Layering, Internet addressing, Addressing Mapping, Routing architecture, and Error and control Messages. |
| 5 | **Distributed Network Structure:** Distributed systems, Data migration and Computation, Communication Structure, naming and name resolution, routing strategies, packet strategies, and connection strategies, design principle, circuit switching, message switching, and packet switching. |
| 6 | **Distributed Network Protocols:** Communication Protocols, Failure Detection; Reconfiguration, Recovery from failure, and Fault tolerance, Design Issues; Scalability Issue, Distributed File Systems, Network Naming; Naming and Transparency, Naming structures, and Naming schemes. |
| 7 | **Distributed System Models:** Scalable computing over internet; Internet computing, High-throughput computing, Multi-threading technologies; multi-core CPU, and GPU programming model, Clusters, P2P networks, Grids, and Cloud platforms. |
| 8 | **Virtual Machines and Virtualization in Cloud Computing:** Hardware virtualization, Virtualization of server, Virtualized resources in cloud computing: Amazon Web Service, Microsoft Azure and Google App. |
| 9, 10 | **Computer Clusters for Scalable Parallel Computing:**Clustering: design objective, design issues, dedicated clusters, enterprise clusters, load balancing clusters, resource sharing, system interconnect, design principle, fault tolerance and recovery. |
| 11 | **Cloud Computing from Clustering:** GPU clusters, Single-system image features; single file hierarchy, Single networking, single point of control, single memory space, Fault tolerant configurations, and Recovery Schemas. |
| 12 | **Cloud Platform Architecture:** Cloud computing service models; Three cloud service models, and Cloud types; public, private and hybrid clouds centralized and distributed computing, Cloud development trends; cloud ecosystem, storage clouds, generic cloud architecture, and layered cloud architecture.  |
| 13 | **Multi-Core Operating Systems I:**Multi-core architecture, Symmetric shared memory systems, Distributed memory systems, Cache coherence issue, Snooping and directory based cache protocol systems. |
| 14 | **Multi-Core Operating Systems II:** Invalidation protocols in snoopy and directory based systems, Fault tolerant configurations: recovery Schemas, and analysis of job scheduling methods. |

# Projects:

1. Show a simulation of **banker’s algorithm** for multiple resource unit allocation problems. The program should show whether a solution end with a ***safe state*** or not. And also the program should have the facility to collect input dataset from both a text file and from a random generator (6%).
2. **Research paper presentation** (9%): Each student should study and present existing research papers (high quality) in the field of advanced computing.
3. **Distributed Thread scheduling** (15 %): Each student should simulate and present a thread scheduling program. The details of a sample thread scheduling program are described below:
	* Consider a job scheduling problem: ***J*** = {1, .. , *j*} and ***M***= {1,…, *m*} as *J***jobs** and *M* **machines**  of the problem, where *j* and *m* are the independent jobs and unit machines of *J* and *M* respectively. User should provide a job set and machines to the scheduler in prior to a scheduling process. The scheduler is arranged in such a way that it always works with a set of jobs (***J*** > ***M***).
	* Each job is assumed to handle *k* subtasks (considered as *threads*) and is dependent to each other. Similarly, the number of subtasks of a job is not fixed. Prior to each scheduling process, it is assumed that a set of jobs with their subtasks is always available in the *job queue* of the scheduler.
	* Each subtask of a job is represented as a set of attributes. That is, let *T*11 is a subtask of job *J*1 and can be represented as {*a*11 /\ *a*12 /\ …./\ *a*1*n*}, where *a*11, *a*12,.., etc., are the conjunction of the attributes of subtask, *T*11. For example, a subtask can be represented by four attributes; *arrival time, waiting time, processing time*, and *deadline*.
	* All attributes of a subtask are known in advance.

Based on the above description of the job scheduling problem, show the following:

i). Find the precedence order of subtasks of each job by using a suitable precedence order detection algorithm (no deadlock or starvation allowed).

ii). Find the priority of each job (by applying a standard rule format to its subtasks)

iii) Distribute the subtasks of each job to the given machines based on their priority without causing idle machine state and violating task precedence order.

iv). Calculate the FT (finishing time) of each job set.