Effective Teaching of “Computer Networks”

(A Module in the CSEDU – Certificate Program in CS Education)

Objective

The aim of this module is to help teachers in colleges/universities (attendees) improve their teaching of the Computer Networks course, as per the draft AICTE syllabus (given at the end of this document). After this module, the participants will improve their teaching of this course leading to improved learning by their students and more students achieving the learning outcomes of the AICTE course. This module is part of the Certificate Program in CS Education initiative (csedu.iiitd.ac.in). The main learning outcomes of ***this module*** are at the end of the module, a participant will:

- Have clearer understanding of importance of this course in a CSE program, the desired learning outcomes established in the AICTE course, and the course syllabus
- For most topics in the AICTE syllabus, have a deeper understanding of concepts, what is important, how to teach, what type of assignments to give, etc.

The focus of the module is to help in delivering the Essential Learning Outcomes of the AICTE syllabus. Some advanced topics may also be discussed, based on the inputs from the participants.

Requirements for Module Participants

The participants for this module should:

- Have taught Computer Networks earlier
- Have access to a good laptop (or desktop) with Linux and Internet
- Commit to spending at least 5 hrs per week (average) for the module

Module Syllabus

Each weekly session will discuss the topics listed below, how to effectively teach them and sample assignments. Lab assignments will be given each week. Some sessions will have in-class assignments where the participants will be expected to work out something and discuss with the instructor/other participants.

Slides / Notes that can be used for teaching students will be discussed and provided; sample assignments that the participants can use in their class will also be provided.

Week-wise syllabus for this module

Wk	Topic	Approach - TBD	Assignment – Self-work examples for the week
1	a. Discuss role of this course in overall CS syllabus, and overall learning b. Discuss the AICTE syllabus in detail – learning outcomes, modules, approach c. Discuss this module's goals for learners (teachers), approach d. Discuss the feedback received from the attendees		a. Some reading, watching some video, ..relating to teaching of this b. Other general work c. Installation of Linux (if needed)
2	**Architecture & Design Principles:** a. History of the Internet: 1970s to the present b. Layering as a means of structuring the architecture of a complex system. c. How an architecture from 1970s has today become a critical infrastructure supporting applications not imagined at that time. ● Two approaches to complex system design. ● The end-to-end principle in systems design. ● Best-effort packet-switching. d. The downsides of using a 40-year old architecture.		Lab exercises to gain familiarity with Linux networking tools.

3	**Approaches to Teaching CN: top-down vs. bottom-up** a. Application-driven design and component-driven design. b. Pros and cons of top-down and bottom-up approach to teaching networks. **c.** Understand how to teach top-down effectively.		a. Installation of network tools b. Installation of Apache web server or other network application
4	**Linux Network Programming** a. Understand how the various network layers map to software or hardware in a Linux-based system. b. Understand how to build simple networking applications over the socket API. c. Common design patterns of web/application servers that concurrently handle a large number of clients.		Socket programming project to build a simple client-server application, with single and multiple clients
5	**Internet Addressing, Scale and Topology** a. Scale and topology of the Internet, in terms of IP addresses, DNS names, autonomous systems, ISPs, and so on. b. Design principles of the Internet that enable scaling: hierarchical IP addresses and DNS names, separation of inter-domain and intra-domain routing, hierarchical routing and forwarding.		a. Query internet registries and web-based routing looking glasses to understand Internet topology b. Use ping and traceroute to understand network paths to various popular servers. Measure typical Internet latencies.

6	**Protocols:** a. The need for protocols, how to design and reason about protocols; performance aspects. b. Finite-state machines, space-time diagrams and other tools. c. Gateways: protocol conversion, forwarding at various layers.		Install NS2/NS3. Simulation of a simple protocol
7	**Performance:** a. Performance metrics, contexts in which a particular metric is important b. Resources that impact link-level or network-level performance, including bandwidth, buffers, etc c. Underlying phenomena that influence performance, including queuing, multiplexing and routing d. Little's Law a. Measurement of performance in a large network vs. steady-state analysis		a. Install and run iperf3 to measure performance of a network. b. Simulate a small network using NS2/NS3 to assess performance with varying bandwidth, traffic, routing, etc.
8	**Error Handling and Recovery:** a. Causes of errors; Error detection vs. error detection and correction b. Error recovery using retransmissions, ACKs, or sliding window protocols c. Role of RTT and delay-BW product in fixing optimum timeouts and window size d. Recovery from errors at layers from MAC through Transport to Application		Build upon the NS2/NS3 simulation in Week 7 to study the impact of packet drops on throughput and delay.

9	**Flow control** & **Congestion control**: a. Flow control and congestion control at multiple layers. b. Understand the difference between flow control and congestion control, through examples of analogous real-life systems. c. Motivate Slow start and Congestion Avoidance (delay/loss based) in TCP. d. Understand network-assisted congestion control -- packet marking and dropping.		Assignments showing performance implications of flow and congestion control
10	**Sharing a Medium**: a. Protocols for access control in shared medium: fixed vs. dynamic allocation, centralized vs. decentralized control, contention vs. polling access b. Contexts in which TDMA, FDMA, CDMA, Aloha, CSMA/CD, CSMA/CA are applicable, c. MAC in WiFi: Hidden-node problem; IEEE 802.11 CSMA/CA with RTS/CTS.		a. Measure RSSI, throughput and latency between client and server connected to a working AP. b. Simulate a small WiFi network using NS2/NS3 to evaluate throughput vs. access delay
11	**Advanced Topics**: Introduction to and pointers for further research in: a. SDN and data centre networking b. Mobile data networks c. Streaming media protocols and services.		

12	a. How mini-projects are handled; sample projects; possible projects b. How to motivate students to learn on their own – participate in contests, etc c. Wrap-up and feedback		Example projects

Schedule

The module will meet online once a week. In addition, a weekly help session to clear doubts and to help with the assignment will be provided through TA s. Details about joining these session will be provided later.

- **Weekly Session**: Monday, 4-5:30 pm (or 4:30 pm to 6 pm)
- **Weekly help Session**: Thursday, 4-5:30 pm (or 4:30 pm to 6 pm)

Text to be used for the Module

The text book suggested in the AICTE syllabus will be used as the basis of this module. Some additional references will be pointed out at the end of each module.

Resources to be provided to participants.

- Lecture Notes / ppt for the different topics in the course – which they can use in their own teaching
- Some sample assignments for each of the major modules
- Some online resources which the attendees can use to revise/update their knowledge, …

Post Module Support

- A mailing list will be created and for a year, once every few months an online session of all attendees will be planned to share experiences and discuss challenges being faced

Appendix: Revised AICTE Syllabus for the course (Tentative)

CS602	**Computer Networks**	**3L:0T: 2P**	**4 credits**	**Pre-Reqs: CS402 Computer Organisation, CS403 Operating Systems**

This course introduces students to the fundamental principles on computer networks, and to their implementation in the Internet. Lab assignments cover network programming, network tools, applications, and simulation. Through these assignments, students get a deeper understanding by building, operating and tuning components of the Internet. The top-down, application-driven design of the course motivates students and enables them to work with real-world applications from early in the course.

Learning Outcomes of the course (i.e. statements on students' understanding and skills at the end of the course the student shall have):

Essential (<=6):

1. Understand the architecture principles that have enabled the orders of magnitude expansion of the Internet
2. Understand networked applications and their protocols; installation, operation and performance tuning
3. Understand layering as a means of tackling complexity, layering applied to the Internet
4. Understand protocols as a structured means of reliable communications
5. Be conversant with network programming using the socket API, building scalable and efficient network applications
6. Be familiar with tools for configuring, monitoring and tuning the Internet and host.

Desirable/Advanced (<= 3):

1. Understand basics of data centre networks and software-defined networks (SDN)
2. Understand cellular networks, mobility and the impact on applications
3. Understand the interaction between streaming media applications and the underlying network infrastructure.

Detailed contents for Essential Learning Outcomes:

Module (approx duration in wks)	Topics	Pedagogy / teaching suggestions	Lab / assignment / practice
Module 1: Introduction to the Internet (1 week)	- Overview of how the Internet works. Understand what happens when we browse a website. Understand basic terminology like browser, web server, URL, domain name, IP address, packets. (1 hour) - Understand the design principles of the Internet: packet switching vs circuit switching, store-and-forward networks, layering for modularity. Introduction to the various layers in the Internet. (1 hour) - Performance metrics like end-to-end throughput, delay, jitter and drop rates in a network. Little's Law. How performance is measured. Confidence intervals. (1 hour)	**Kurose & Ross**: Chap 1 (sec. 1.1-1.5). **Raj Jain**: Sec. 3.2 - 3.4 Top-down sequence is recommended. Performance metrics and measurement concepts are introduced in Module 1 and used in all modules (lectures and labs) as appropriate.	- Use Wireshark to capture packets when browsing the Internet. Examine the structure of packets: the various layers, protocols, headers, payload. - Use Linux tools like ifconfig, dig, ethtool, route, netstat, nslookup, and ip to understand the networking configuration of the computer that the student is working on.

Module 2: Application layer (2 weeks)	- Internet names, how DNS works. (1 hour) - Application layer protocols: HTTP, SMTP, SNMP, web applications. (3 hours) - Peer-to-peer applications. P2P file distribution. (1 hour) - Audio and video streaming. Adaptive streaming. Understand challenges of streaming over best effort IP. (1 hour)	**Kurose and Ross**: Chap 2 (sec. 2.1-2.5), Sec. 7.1.3	- Install and configure some network applications, eg. Apache, Bind (DNS), etc. - Understand various header fields and their usage in different application layer protocols using Wireshark packet capture.
Module 3: Linux Network Programming (1 week)	- Introduction to socket programming in Linux. Understand how to build a simple client-server application using TCP/UDP sockets. (2 hours) - Techniques for concurrent I/O: one-thread-per-connection (synchronous) and event-driven (asynchronous) server architectures. (1 hour)	**Kurose and Ross**: Sec 2.7. **OSTEP:** Chapter 33	- Socket programming: write a simple client-server program using TCP and UDP sockets. - Modify server to handle multiple clients concurrently.

Module 4: Transport Layer (2 wks)	- Importance of the transport layer; end-to-end argument. Transport layer protocols: basics of TCP and UDP, process-to-process delivery, multiplexing, port numbers, header structure. (2 hours) - Reliable transmission of packets over an unreliable network: sequence numbers, ACKs, timeout, retransmissions. Stop and wait, Go-back-N and sliding window. - TCP reliability. TCP connection setup and teardown. (2 hours) - Flow control and congestion control at the transport layer. Differences between the two. (1 hour) - TCP congestion control: Slow start; congestion avoidance using loss-based and delay-based control. Effect on performance. (1 hour)	**Kurose & Ross**: Chap. 3 (sec. 3.1-3.6)	- Measure TCP throughput between two hosts in a network using tools like iperf. Modify TCP configuration parameters. Use tc or similar to control bandwidth, delay, loss. Observe impact on measured throughput. - Experiment with multiple applications running concurrently to generate congestion. - (Optional) Observe the behavior of congestion control protocols in ns-2/ns-3, change various network parameters and observe evolution of the TCP congestion window.

Module 5: The IP Layer (2 wks)	**[A] Network architecture and performance** - Network topology; Router architecture: queueing and switching. (1 hour) - Performance analysis of a single link: Basics of queuing theory, M/M/1 queues, traffic characteristics, performance measures. (1 hour) - Network-level performance analysis using network of queues model: performance in terms of throughput, delay, jitter, and drop rates. Bottleneck analysis, link capacity planning. (2 hours) **[B] IP Protocol** - Need for an Internet address, and its design. Basics of IP: hierarchical IP addressing, IPv4 and IPv6, structure of IP datagram, IP forwarding, IP datagram fragmentation. (1 hour) - Static and dynamic address allocation, private IP addresses, NATs, DMZ, firewalls. (1 hour)	**Kurose & Ross**: Chap. 4 (Sec. 4.1, 4.3, 4.4 **R. Jain**: Sec 30.1, 30.2, 31.2, 32.3	- Use tools like ping and traceroute to explore various Internet paths to popular servers. - Understand IP address allocation (whether static or dynamic, whether public or private) in student's network. - Use NS-2/NS-3 to simulate a mesh of at least 6 nodes and evaluate the performance under various conditions

Module 6: Routing protocols and Internet architecture (2 wks)	-Routing protocols: Link state routing. Distance vector, count-to-infinity, routing convergence. (2 hours) - Understand the structure of the Internet: end user organizations and Internet service providers. Understand the difference between intra-domain and inter-domain routing. Intra-domain routing: OSPF. (2 hours) - Inter-domain routing: brief mention of BGP. Broadcast and multicast routing (1 hour)	**Kurose & Ross**: Chap 4 (Sec. 4.5 - 4.7)	- Configure a simple mesh network using computers in the lab, or using emulators like Mininet. Setup static routes to conform to the desired mesh topology. - Use web-based tools like whois to query Internet registries, and understand which IP addresses are allocated to the student's network. Use web-based looking glasses to understand the structure of the Internet. Find out which are the major ISPs, and what is the ISP of the student's network.
Module 7: Data Link Layer (2 wks)	- Mechanisms for error detection/recovery: Parity checks, CRC (1 hour) - Medium access protocols: Polling vs. contention-based (delay throughput tradeoff, etc): TDM, FDM, slotted Aloha, Aloha, CSMA/CD (3 hours) - Switched LANs: L2 addressing and ARP, Ethernet frame structure, learning switches. (2 hours)	**Kurose & Ross**: Chap. 5 (Sec. 5.1 - 5.4)	- Use Linux network tools like ethtool to observe and analyze link layer packet statistics and errors. - Use ns-2/ns-3 to simulate medium access protocols. Observe contention, collisions and packet loss in medium access protocols. Observe the working of error detection/recovery

			mechanisms in ns-2/ns-3.
Module 8: Wireless Networks (1 wk)	- Wireless physical layer: signal-to-noise ratio, bit error rate, modulation, multipath, interference (1 hour) - Wireless LANs: 802.11 architecture (access points, SSID, channels, beacons, scanning, association), Hidden terminal problem, 802.11 CSMA-CA protocol; summary of 802.11 variants; mobility between APs (2 hours)	**Kurose & Ross**: Chap 6 (Sec. 6.1, 6.2, 6.3)	- Use cellphone to measure WiFi signal strength (RSS) at various places in the campus. Draw a contour map with access points and RSS levels. Correlate with upload/download speed using tools like Measurement Lab speedtest. - (Optional) Understand the behavior of WiFi using ns-2/ns-3.

Detailed Contents for Desirable Learning Outcomes (optional, <= 3 modules):

Module (approx duration in wks)	**Topics**	**Pedagogy/ teaching suggestions**	**Lab / assignment / practice**
Module D1: SDN and data centre networking	Data centre network design and topology. Transport protocols optimized for data centres. Introduction to software defined networking.	**Kurose and Ross:** Chap. 5.6 Reference papers [8], [9]	Simulate various transport protocols optimized for data centres in ns-3. Practical exercises TBD
Module D2: Mobile data networks	Cellular Internet access. Mobility management and handovers. Impact of mobility on higher layers.	**Kurose and Ross:** Chap. 6.4-6.8	To be decided

Module D3: Streaming media protocols and services	Multimedia applications. Audio and video streaming over UDP, HTTP. Adaptive streaming. Voice over IP. Recovering from packet loss and jitter. QoS. Protocols for real time applications.	**Kurose and Ross:** Chap. 7	Implement a streaming audio/video server.

Suggested text books / Online lectures or tutorials:

1. J.F. Kurose and K.F. Ross, *Computer networking: a top-down approach*, 6th ed, Pearson, 2017. (*Low-cost Indian edition. Latest edition is high-cost, may be used if available.)*

Suggested reference books / Online resources:

1. R. Jain, *The art of computer systems performance analysis,* Wiley India, 1991.
2. [OSTEP] Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau, *Operating Systems: Three Easy Pieces,* (https://pages.cs.wisc.edu/~remzi/OSTEP/), 2018
3. A.S. Tanenbaum and D.J. Wetherall, *Computer Networks*, 5th ed., Pearson, 2013.
4. Larry Peterson and Bruce Davie, *Computer Networks: A Systems Approach*, 6th Ed., available at https://book.systemsapproach.org/
5. Nick Feamster, Jennifer Rexford, Ellen Zegura, “The Road to SDN”, *ACM Queue*, 2013.
6. Alizadeh et al., “Data Center TCP”, *ACM SIGCOMM 2010*.
7. *The Network Simulator - ns-2*, https://www.isi.edu/nsnam/ns/
8. E. Altman and T. Jimenez, *NS2 Simulator for Beginners*, 2003.
9. *ns-3 network simulator*. https://www.nsnam.org/