NETWORK AND SYSTEMS PROGRAMMING

ENCT 386

Lecture	**: 3**	**Year : III**
Tutorial	**: 2**	**Part : II**
Practical	**: 1**	

Course Objectives:

The objective of this course is to provide knowledge of networking and system-level programming concepts. It focuses on socket programming, process and thread management, and the use of system calls for efficient resource handling. Upon completion, students will be able to develop concurrent, scalable, and high-performance network applications suitable for real-world computing environments.

1 System Programming Fundamentals (4 hours)

1.1 Overview of system programming and its role in networking
1.2 User space versus kernel space
1.3 System calls and standard libraries
1.4 File descriptors and low-level I/O
1.5 Linux/Unix system architecture overview

2 Process Management and IPC (7 hours)

2.1 Process creation and control (Fork, exec, wait)
2.2 Process states and scheduling concepts
2.3 Signals and signal handling
2.4 Inter-process communication: Pipes and FIFOs, message queues, shared memory, semaphores

3 Threads and Concurrent Programming (7 hours)

3.1 Thread models and lifecycle
3.2 POSIX threads (Pthreads)
3.3 Thread synchronization: Mutexes, condition variables
3.4 Deadlocks, race conditions, and avoidance
3.5 Multithreading versus multiprocessing (Performance considerations)

4 Socket Programming and Network APIs (10 hours)

4.1 Socket interface and API
4.2 TCP socket programming (Client-server model)
4.3 UDP socket programming
4.4 Blocking versus non-blocking sockets
4.5 I/O multiplexing (Select, poll basics)

4.6 Error handling and debugging techniques

5 Advanced Network Programming (7 hours)

5.1 Concurrent server design: Iterative versus concurrent servers, multi-process and multi-threaded servers
5.2 Event-driven programming: Select, poll, epoll
5.3 High-performance server design principles
5.4 Concepts of remote procedure call (RPC)
5.5 Introduction to RESTful APIs and microservices
5.6 Secure communication basics (SSL/TLS overview)

6 System-Level I/O and Performance Optimization (5 hours)

6.1 Advanced file I/O (Read/write optimization)
6.2 Non-blocking I/O and asynchronous I/O concepts
6.3 Memory-mapped files (Mmap)
6.4 File locking and concurrency
6.5 Performance tuning and profiling basics

7 Network Application Development and Case Studies (5 hours)

7.1 Design of scalable client-server systems
7.2 Development of network applications: Chat servers, file transfer systems
7.3 Debugging and monitoring tools (Netstat, ss, tcpdump, wireshark)
7.4 Case studies of real-world systems (Web servers, distributed apps)

Tutorial (30 hours)

1. Implement file operations using low-level system calls (Open, read, write, close) and compare with standard I/O
2. Demonstrate file descriptor manipulation using dup()/dup2() for I/O redirection
3. Write a program using fork() and exec() to create and replace processes
4. Implement IPC using pipes and shared memory, comparing their behavior
5. Create multiple threads using POSIX threads (pthreads) and observe execution behavior
6. Solve the producer-consumer problem using mutexes and condition variables
7. Develop a TCP client-server application (Echo service)
8. Implement a UDP communication program and compare with TCP behavior.
9. Build a multi-client concurrent server using threads or processes
10. Implement I/O multiplexing using select() or poll() for handling multiple clients
11. Demonstrate memory-mapped file (mmap) usage and compare with standard file I/O
12. Implement file locking mechanisms to handle concurrent file access
13. Develop a multi-client chat application using sockets
14. Capture and analyze traffic of your application using tools like Wireshark/tcpdump

15. Design and implement a scalable network application (chat/file server) incorporating: Concurrency (threads/processes), socket programming, basic performance optimization

Practical **(15 hours)**

1. System calls and file operations in Linux
2. Process creation using fork/exec and process control
3. IPC using pipes, shared memory, and message queues
4. Signal handling programs
5. Thread creation and synchronization using Pthreads
6. TCP client-server socket programming
7. UDP-based communication programs
8. Multi-client concurrent server (thread/process-based)
9. I/O multiplexing using select/poll
10. Packet capture and analysis using Wireshark
11. Mini-project: Design and implement a scalable network application

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	4	5
2	7	9
3	7	9
4	10	14
5	7	9
6	5	7
7	5	7
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Forouzan, B. A. (2017). Data communications and networking. McGraw-Hill Education.
2. Kurose, J. F., Ross, K. W. (2021). Computer networking: A top-down approach. Pearson.
3. Stevens, W. R. (2013). Advanced programming in the UNIX environment. Addison-Wesley.
4. Stevens, W. R. (2003). UNIX network programming. Addison-Wesley.
5. Kernighan, B. W., Pike, R. (1984). The UNIX programming environment (Latest Edition). Prentice Hall.

Next Generation Networking with IPv6

ENCT 387

Lecture : 3 **Year : III**
Tutorial : 2 **Part : II**
Practical : 1

Course Objectives:

The objective of this course is to develop a solid understanding of internet protocols with a focus on IPv6 and next-generation networking concepts. It aims to equip students with knowledge of IPv6 addressing, routing, security and transition mechanisms from IPv4 to IPv6. The course also emphasizes practical skills in configuring, analyzing and deploying IPv6-based networks and services. Ultimately, it prepares students to design and manage modern network infrastructures in real-world environments.

1 Fundamentals of Internet and Networking Protocols (3 hours)

1.1 Evolution of the Internet and networking paradigms
1.2 OSI versus TCP/IP model
1.3 IPv4 addressing overview
1.4 IPv4 addressing review and limitations
1.5 Need and importance of next generating IP addressing

2 Next Generation Internet Protocol (12 hours)

2.1 Introduction to IPv6
  2.1.1 History and need for IPv6
  2.1.2 IPv6 header format
  2.1.3 Features of IPv6
  2.1.4 IPv6 addressing format and types
  2.1.5 Extension headers
2.2 ICMPv6
  2.2.1 Features
  2.2.2 General message format
  2.2.3 ICMP error and informational message types
  2.2.4 Neighbor discovery
  2.2.5 Path MTU discovery

3 Security and Quality of Service in IPv6 (6 hours)

3.1 Network threats and vulnerabilities in IPv6
3.2 Security techniques
3.3 IPSEC framework
3.4 Basic firewall and filtering concepts

3.5 Zero trust networking
3.6 QoS concept in IPv6
3.7 QoS in multimedia and real-time applications

4 Routing with IPv6 (6 hours)

4.1 Routing in the internet and CIDR
4.2 Unicast and multicast routing concepts
4.3 Unidirectional link routing
4.4 Routing protocols: RIPng, OSPFv3 and BGP (Basic concepts for IPv6)
4.5 Multicast routing (PIM-SM overview)

5 IPv4/IPv6 Transition Mechanisms (6 hours)

5.1 Tunneling: Automatic tunneling; Configured tunneling
5.2 Dual stack approach
5.3 Translation mechanisms
5.4 Migration strategies for enterprises and ISPs
5.5 Case studies of global IPv6 transition

6 IPv6 Deployment and Network Design (5 hours)

6.1 IPv6 deployment challenges and risks
6.2 Enterprise and ISP deployment strategies
6.3 IPv6 addressing plan design
6.4 IPv6 DNS (AAAA records)
6.5 IPv6 enabled proxy, web and mail servers

7 Advanced Networking Applications (7 hours)

7.1 Fundamentals of SDN
7.2 Introduction to IBN and NDN
7.3 IPv6 as the foundations of 5G network
7.4 IPv6 integration SDN/IBN/NDN

Tutorial (30 hours)

1. Compare OSI vs TCP/IP models through layered packet flow analysis
2. Analyze limitations of IPv4 addressing and justify the need for IPv6 using numerical examples
3. Construct and analyze an IPv6 packet, including header fields and extension headers
4. Perform IPv6 addressing and subnetting exercises (Global, link-local, multicast)
5. Implement and observe neighbor discovery protocol and ICMPv6 message types
6. Demonstrate basic IPsec configuration concepts
7. Analyze QoS mechanisms in IPv6, especially for real-time/multimedia traffic

8. Configure and test IPv6 unicast routing (OSPFv3 basics) in a simulated environment
9. Comparison of IPv6 multicast routing (PIM-SM concept) with unicast routing
10. Implement dual stack configuration in a lab setup
11. Demonstrate tunneling mechanisms and analyze packet flow
12. Design an IPv6 addressing plan for an enterprise network
13. Configure and test IPv6 DNS (AAAA records) and basic web service
14. Analyze the role of IPv6 in SDN/5G/NDN architectures through a case study
15. Design and simulate a complete IPv6-enabled network, including: Addressing plan, routing (OSPFv3 or static), transition mechanism (Dual stack/tunneling), basic service deployment (DNS/Web)

Practical **(15 hours)**

1. Enabling and configuring IPv6 in Windows/Linux
2. IPv6 header and packet analysis using Wireshark
3. Neighbor Discovery and ICMPv6 analysis
4. IPv6 addressing and subnetting practice
5. Unicast routing using OSPFv3
6. Multicast routing using PIM-SM
7. IPv6 DNS, web, and proxy server configuration
8. Case Study: Design and deploy an IPv6-based network

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapters	Hours	Marks distribution*
2	12	16
3	6	8
4	6	8
5	6	8
6	5	8
1 and 7	10	12
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Davies, J. (2008). Understanding IPv6. Microsoft Press.
2. Hagen, S. (2014). IPv6 essentials. O’Reilly Media.
3. Thomas, S. A. (1996). IPng and the TCP/IP protocols. John Wiley & Sons.
4. Hersent, O., Gurle, D., & Petit, J.-P. (2000). IP telephony: Deploying VoIP protocols. Addison-Wesley.

ANALYSIS OF ALGORITHMS
ENCT 388

Lecture : 3 **Year : III**
Tutorial : 2 **Part : II**
Practical : 1

Course Objectives:
The objective of this course is to provide students with a strong foundation in the analysis of algorithm efficiency and computational complexity. It aims to develop understanding of key algorithm design paradigms, including divide-and-conquer, greedy methods, dynamic programming, and backtracking, and covers fundamental concepts of NP-completeness and approximation algorithms.

1 Introduction to Algorithm Analysis (5 hours)

1.1 Algorithm and its properties, RAM model, time and space complexity, detailed analysis of algorithms, Concept of Aggregate Analysis
1.2 Asymptotic notations (Big-O, Big-Ω and Big-Θ), their geometrical interpretations and examples
1.3 Concept of best case, average case and worst case performance of an algorithm
1.4 Modeling algorithms by recurrence relation
1.5 Solving recurrence relation for evaluating computational complexity
  1.5.1 Recursion tree method
  1.5.2 Substitution method
  1.5.3 Using masters theorem

2 Iterative and Numeric Algorithms (8 hours)

2.1 Algorithm for GCD and Fibonacci number
2.2 Sequential search
2.3 Review of bubble sort, selection sort, and insertion sort algorithms
2.4 Number theoretic notations
2.5 Euclid’s and Extended Euclid’s algorithms
2.6 Solving modular linear equations using Chinese remainder theorem
2.7 Fermat’s theorem
2.8 Miller-Rabin randomized primility test and algorithm

3 Divide and Conquer Algorithms (8 hours)

3.1 Binary search, min max finding algorithm
3.2 Analysis of sorting algorithms
  3.2.1 Merge sort

3.2.2 Heap sort
3.2.3 Quick sort
3.2.4 Randomized quick sort

3.3 Order statistics

3.3.1 Selection in expected linear time
3.3.2 Selection in worst case linear time

4 Greedy Algorithms (7 hours)

4.1 Basic concepts
4.2 Fractional knapsack problem
4.3 Job sequencing with deadlines
4.4 Analysis of minimum spanning trees related algorithms
4.5 Analysis of single source shortest path algorithm

5 Dynamic Programming (8 hours)

5.1 Basic concepts
5.2 All pair shortest path algorithm
5.3 Travelling salesperson problem
5.4 String editing
5.5 0/1 knapsack problem using dynamic programming
5.6 Matrix chain multiplication
5.7 Flow shop scheduling

6 Backtracking Techniques (4 hours)

6.1 Basic concepts
6.2 The N-Queen problem
6.3 Sum of subsets
6.4 Graph coloring
6.5 Hamiltonian cycles
6.6 0/1 knapsack problem using backtracking approach

7 NP-Hard and NP-Complete Problems (5 hours)

7.1 Basic concepts
7.2 Cook's theorem
7.3 NP-Hard graph problems
7.4 NP-Hard scheduling problems
7.5 NP-Hard code generation problems
7.6 Simplified NP-hard problems
7.7 Approximation algorithms: ε- approximation, polynomial time approximation scheme, probabilistically good algorithms
7.8 Vertex cover problem, subset sum problem

Tutorial **(30 hours)**

1. Algorithm analysis
2. Iterative and numeric algorithms
3. Divide and conquer algorithms
4. Greedy algorithms
5. Dynamic programming
6. Backtracking techniques
7. NP-Hard and NP-complete problems

Practical **(15 hours)**

1. Implementation and complexity analysis of iterative, numeric and recursive algorithms
2. Implementation and complexity analysis of greedy algorithms
3. Implementation and complexity analysis of algorithms involving divide and conquer strategy
4. Implementation and complexity analysis of algorithms based on dynamic programming
5. Implementation and complexity analysis of algorithms using backtracking concept

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	5	6
2	8	10
3	8	11
4	7	9
5	8	12
6	4	5
7	5	7
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Horowitz, E., Sahni, S., Rajasekaran, S. (2007). Fundamentals of computer algorithms. Universities Press.
2. Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. (2022). Introduction to algorithms. MIT Press.
3. Kleinberg, J., Tardos, É. (2006). Algorithm design. Pearson.
4. Skiena, S. S. (2020). The algorithm design manual. Springer.
5. Dasgupta, S., Papadimitriou, C. H., Vazirani, U. V. (2006). Algorithms. McGraw-Hill Education.

AUDIO PROCESSING
ENCT 389

Lecture	**: 3**	**Year : III**
Tutorial	**: 2**	**Part : II**
Practical	**: 1**	

Course Objectives:

The objective of this course is to develop competency in analyzing and processing audio signals in time and frequency domains. It covers fundamentals of acoustics, digital audio representation, spectral analysis and audio effects, along with practical skills in feature extraction and Music Information Retrieval (MIR). Hands-on use of Digital Audio Workstations (DAWs), MIDI, and plugins enables application of audio signal processing techniques in multimedia, speech processing, music technology and emerging digital audio applications.

1 Fundamental of Sound and Audio Signal (7 hours)

- 1.1 Physical acoustic: Wave propagation, frequency, amplitude, phase
- 1.2 Psycho acoustic: Pitch, loudness, timbre, dynamics and intensity
- 1.3 Fundamental of digital audio
- 1.4 Audio file format: WAV, AIFF, FLAC, ALAC, MP3
- 1.5 Practical audio quality: Sample rates (44.1khz, 48khz, 96khz) and bit depths

2 Time Domain Audio Processing (8 hours)

- 2.1 Discrete time audio signal, basic signal operations, system in time domain
- 2.2 Convolution and impulse response
- 2.3 Analog and digital filters in audio systems
- 2.4 Temporal feature extraction

3 Spectral Representation (7 hours)

- 3.1 DFT and FFT: Derivation and computational complexity
- 3.2 Windowing function in audio
- 3.3 Short time Fourier transform: Frame size, hope size and overlap
- 3.4 Spectrograms: Reading and interpreting time-frequency representation,
- 3.5 Mel spectrogram: Reading and interpreting; Mel frequency representation
- 3.6 Constant Q transform and its relevance to music analysis

4 Frequency Domain and Non-Linear Effects (10 hours)

- 4.1 Sinusoidal model
- 4.2 Deterministic plus residual model
- 4.3 Deterministic plus stochastic model

4.4 Basis of non-linear effects
4.5 Dynamic range control: Envelop follow, compressor and limiting, expansion and gating, ADSR, modulation effect
4.6 Time segment process: Time scratching and pitch shifting

5 Music Information Retrieval (6 hours)

5.1 MIR overview: Types of music data, challenges, application
5.2 Spectral centroid and roll off
 5.2.1 Basics of spectral centroid and spectral roll
 5.2.2 Comparison between centroid and roll off
 5.2.3 Application (Timbre analysis, music/ speech classification)
5.3 Mel frequency cepstral coefficient (MFCC)
 5.3.1 Motivation for ceptral features
 5.3.2 Steps in MFCC
 5.3.3 Interpretation of coefficient
 5.3.4 Application (Speech recognition, music classification)
5.4 Chroma
 5.4.1 Concept of pitch classes (12- tone system) mapping
 5.4.2 Chromagram representation,
5.5 Onset and tempo and their applications

6 Audio system and DAW Integration (4 hours)

6.1 Digital audio workstation (DAW)
 6.1.1 Role in audio production
 6.1.2 History and evolution of DAW
6.2 DAW available in market, DAW interface, timeline and arrangement view
6.3 MIDI: Basics, MIDI message structure, VSTi basics,
6.4 Common plugin: Equalizer, compressor, reverb, delay

7 Recent Trend (3 hours)

7.1 Machine learning in audio
7.2 Audio foundation models for bioacoustic and healthcare
7.3 Spatial audio
7.4 Audio DSP ICs
7.5 Generative AI in music

Tutorial (30 hours)

1. Discuss lossless versus lossy compression in the context of audio
2. Derive the convolution sum for a linear time-invariant (LTI) system and explain its physical significance in audio signal processing.
3. Discuss different types of analog filter. Write an advantage of FIR and IIR filter in audio processing
4. Case study discussion
5. Explain the various steps involved in MFCC

6. State the Nyquist theorem and find the Nyquist rate for a signal with maximum frequency 8 kHz.
7. Explain the concept of a chromagram in music information retrieval
8. Explain binaural rendering using head-related transfer functions (HRTFs) and identify one application domain for dedicated audio DSP hardware
9. Music feature analyzer: Input (Audio file), output (Spectrogram, MFCC, tempo)
10. Mini DAW project: Record and edit audio using DAW, apply multiple effects, mix a short track
11. Music classifier: Genre basis classifier

Practical **(15 hours)**

1. Generate sinusoids; Downsample to observe aliasing
2. Measure file size, SNR, and spectral differences between WAV, MP3 (128k/320k), and FLAC
3. Temporal features extraction
4. Spectrogram and STFT
5. Mel spectrogram and CQT
6. MIR features: MFCC
7. DAW: Implement basic features
8. DAW integration and MIDI, VSTi implementation

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	7	9
2	8	11
3	7	9
4	10	13
5	6	8
6	4	6
7	3	4
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Christensen, M. G. (2019). Introduction to audio processing. Springer.
2. Smith, J. O. (2011). Spectral audio signal processing. W3K Publishing. Zölzer, U. (Ed.). (2011). DAFX: Digital audio effects. Wiley.
3. Müller, M. (2015). Fundamentals of music processing: Audio, analysis, algorithms, applications. Springer.
4. Lyons, R. G. (2011). Understanding digital signal processing. Pearson Education.