**Department of Computer Science and Engineering**

**MASTER LAB MANUAL**

**R20**

**Computer Networks Lab**

**III B.TECH, I-SEM**



**KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES**

**Vinjanampadu, Guntur District- 522017 (A. P.)**

|  |  |  |  |
| --- | --- | --- | --- |
| Document NO:  KITS/CSE/LAB MANUAL/NP | Date of issue: | Compiled by | Authorized by  Prof.R.RAMESH  HOD  DEPT. OF CSE |
|  | Date of revision | Verified by |

**Department of Computer Science Engineering**

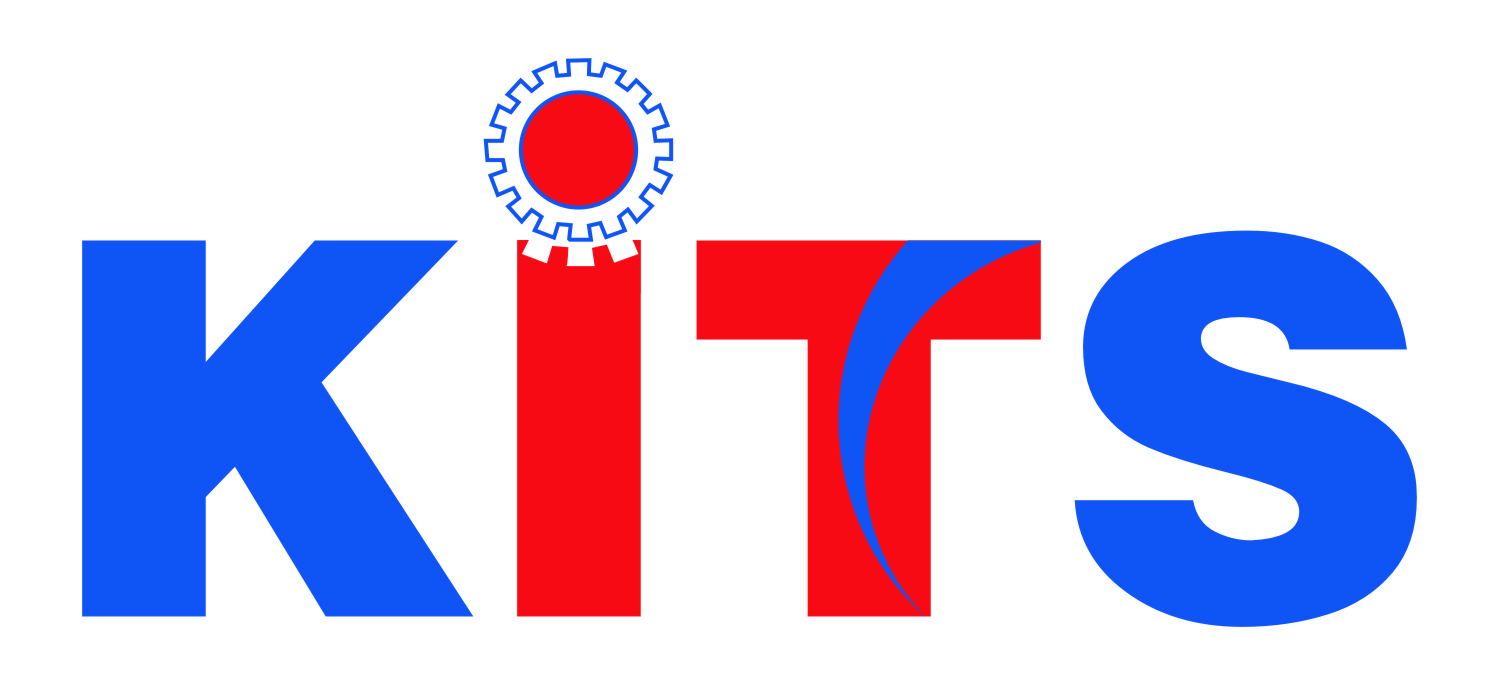
**LAB MANUAL**

**R20**

**Computer Networks Lab**

**[III B.TECH, SEM-I]**

|  |  |  |
| --- | --- | --- |
| **INDEX** | | |
| **S.No** | **Contents** | **Page. No** |
| 1 | Institute Vision & Mission | 3 |
| 2 | Department Vision & Mission | 3 |
| 3 | Program Educational Objectives & Program Outcomes | 4-5 |
| 4 | Program Specific Outcomes | 6 |
| 5 | Syllabus | 7 |
| 6 | Course Outcomes | 8 |
| 7 | List of Experiments | 9 |
| 8 | Course Outcomes of associated course | 10 |
| 9 | Experiment Mapping with Course Outcomes | 10 |
|  | **Experiments** |  |
| 10 | Implement the data link layer framing methods such as character stuffing and bit stuffing. | 11-19 |
| 11 | Write a C program to develop a DNS client server to resolve the given hostname. | 20 |
| 12 | Implement on a data set of characters the three CRC polynomials – CRC-12, CRC-16 and CRC-CCIP. | 21 |
| 13 | Implement Dijkstra’s algorithm to compute the Shortest path in a graph.Take an example subnet graph with weights | 22 |
| 14 | Indicating delay between nodes. Now obtain Routing table art each node using distance vector routing algorithm | 23 |
| 15 | Take an example subnet of hosts. Obtain broadcast tree for it. | 24 |
| 16 | Write a client-server application for chat using UDP | 25 |
| 17 | Implement programs using raw sockets (like packet capturing and filtering) | 26 |
| 18 | Write a C program to perform sliding window protocol. | 27 |
| 19 | Get the MAC or Physical address of the system using Address Resolution Protocol. | 28 |
| 20 | Simulate the Implementing Routing Protocols using border gateway protocol(BGP) | 29 |
| 21 | Simulate the OPEN SHORTEST PATH FIRST routing protocol based on the cost assigned to the path. | 30 |
|  |  |  |
| ADDITIONAL EXPERIMENTS | | |
| 23 | Implementation of CRC | 31 |
| 24 | Implementation of Character Stuffing | 32 |
| 25 | Viva Questions and Answers | 33-34 |

**KKR & KSR INSTITUTE OF TECHNOLOGY & SCIENCES**

**(Approved by AICTE, New Delhi, Affiliated to JNTU Kakinada, Approved by NBA & NAAC)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**INSTITUTE VISION**

|  |
| --- |
| To become a knowledge centre for technical education and also to become the top engineering college in the sunrise state of Andhra Pradesh.  **INSTITUTE MISSION** |
| 1. To incorporate benchmarked teaching and learning pedagogies in curriculum. 2. To ensure all round development of students through judicious blend of curricular, co-curricular and extra-curricular activities. 3. To support cross-cultural exchange of knowledge between industry and academy. 4. To provide higher/continued education and research opportunities to the employees of the institution. |

**DEPARTMENT VISION**

To commit itself to continuously improve its educational environment in order to develop graduates with the strong academic and technical backgrounds needed to achieve distinction and discipline.

**DEPARTMENT MISSION**

To provide a strong theoretical and practical education in a congenial environment so as to enable the students to fulfill their educational and industrial needs.

**PROGRAM EDUCATIONAL OBJECTIVES OF CSE DEPARTMENT**

|  |
| --- |
| **PEO 1:**  Domain Knowledge: Have a strong foundation in areas like mathematics, science and engineering fundamentals so as to enable them to solve and analyze engineering problems and prepare them to careers, R&D and studies of higher level. |
| **PEO 2:**  Professional Employment: Have an ability to analyze and understand the requirements of software, technical specifications required and provide novel engineering solutions to the problems associated with hardware and software. |
| **PEO 3:**  Higher Degrees: Have exposure to cutting edge technologies thereby making them to achieve excellence in the areas of their studies. |
| **PEO 4:**  Engineering Citizenship: Work in teams on multi-disciplinary projects with effective communication skills and leadership qualities. |
| **PEO 5:**  Lifelong Learning: Have a successful career wherein they strike a balance between ethical values and commercial values. |

|  |
| --- |
| **PROGRAM OUTCOMES (PO’S)** |
| **1. Engineering knowledge:**  Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems. |
| **2. Problem analysis:**  Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences. |
| **3. Design/development of solutions:**  Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations. |
| **4. Conduct investigations of complex problems:**  Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions. |
| **5. Modern tool usage:**  Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations. |
| **6. The engineer and society:**  Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice. |
| **7. Environment and sustainability:**  Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development. |
| **8. Ethics:**  Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice. |
| **9. Individual and team work**:  Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings. |
| **10. Communication**:  Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions. |
| **11. Project management and finance**:  Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments. |
| **12. Life-long learning**:  Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change. |
| **PROGRAM SPECIFIC OUTCOME (PSO’S)** |
| **PSO1: Application Development**  Able to develop the business solutions through Latest Software   Techniques and tools for real time Applications. |
| **PSO2: Professional and Leadership**  Able to practice the profession with ethical leadership as an entrepreneur through participation in various events like Ideathon, Hackathon, project expos and workshops. |
| **PSO3: Computing Paradigms**  Ability to identify the evolutionary changes in computing   using Data Sciences, Apps, Cloud computing and IoT. |

**Course Objectives**:

•Understand and apply different network commands

• Analyze different networking functions and features for implementing optimal solutions

• Apply different networking concepts for implementing network solution Implement different network protocols

**COURSE OUTCOMES OF ASSOCIATED COURSE (Computer Networks)**

C321.1 Understand the concepts of LAN technologies and OSI, TCP/IP models and implement basic commands, system calls. (L2 Understand, L3 Apply)

C321.2 Identify the types of media, modulation and multiplexing techniques. (L2 Understand)

C321.3 Discuss various error control techniques and data link protocols. (L2 Understand)

C321.4 Analyze various MAC layer Protocols. (L4 Analyze)

C321.5 Understand routing and congestion control algorithms. (L2 Understand)

C321.6 Design applications using internet protocols. (L6 Create)

**List of Programs**

1) Implement the data link layer framing methods such as character stuffing and bit stuffing.

2) Write a C program to develop a DNS client server to resolve the given hostname.

3) Implement on a data set of characters the three CRC polynomials – CRC-12, CRC-16 and CRC-CCIP.

4) Implement Dijkstra’s algorithm to compute the Shortest path in a graph.

5) Take an example subnet graph with weights indicating delay between nodes. Now obtain Routing table art each node using distance vector routing algorithm

6) Take an example subnet of hosts. Obtain broadcast tree for it.

7) Write a client-server application for chat using UDP

8) Implement programs using raw sockets (like packet capturing and filtering)

9) Write a C program to perform sliding window protocol.

10) Get the MAC or Physical address of the system using Address Resolution Protocol.

11) Simulate the Implementing Routing Protocols using border gateway protocol(BGP)

12) Simulate the OPEN SHORTEST PATH FIRST routing protocol based on the cost assigned to the path.

**Associated Theory Course: Computer Neteorks**

**III Year – II Semester L T P C**

**4 2 0 3**

**Course Objectives:**

The main objectives are

• Study the basic taxonomy and terminology of the computer networking and enumerate the layers of OSI model and TCP/IP model

• Study data link layer concepts, design issues, and protocols

• Gain core knowledge of Network layer routing protocols and IP addressing

• Study Session layer design issues, Transport layer services, and protocols

• Acquire knowledge of Application layer and Presentation layer paradigms and protocols

**Course Outcomes:**

At the end of the course, the students will be able to:

CO-1: Illustrate the OSI and TCP/IP reference model

CO-2: Analyze MAC layer protocols and LAN technologies

CO-3: Summarize various Routing algorithms and Congestion control principles.

CO-4 : Describe Transport layer protocols.

CO-5: Develop application layer protocols

**UNIT I**

**Introduction:** History and development of computer networks, Basic Network Architectures: OSI reference model, TCP/IP reference model, and Networks topologies, types of networks (LAN, MAN, WAN)

**Physical layer:** Different types of transmission media Guided and unguided, Multiplexing methods : TDM, FDM

**UNIT II**

**Data Link Layer**: Design Issues and services: framing, error control, flow control, medium access control. Error & Flow control mechanisms: sliding window protocols: stop and wait, Go back N and selective repeat. **MAC Sub Layer:** MAC protocols: Aloha, slotted aloha, CSMA, CSMA/CD, CSMA/CA, polling, token passing, scheduling.

**UNIT III**

**Network Layer:** Network Layer Services, packet switching, Network Layer Performance, IPv4 addresses, Forwarding of IP packets, Internet Protocol (IP), IPv6 Protocol and addressing, Transition from IPv4 to IPv6, Mobile IP.

**Routing Algorithms:** Least Cost Routing, Distance vector Routing, Link- State Routing, and Hierarchical Routing. Congestion control: : Approaches to Congestion Control, Traffic-Aware Routing, Traffic Throttling, Load shedding, traffic shaping.

**Internet Control Protocols:** ARP, RARP, ICMP and DHCP.

**UNIT IV**

**Transport Layer**: The Transport Service-Services Provided to the Upper Layers, Transport Service Primitives, Elements of Transport Protocols –Addressing, Connection Establishment, Connection Release, Error Control and Flow Control, Congestion control-Desirable Bandwidth allocation, Regulating the sending rate, The Internet Transport Protocols: Introduction to UDP, Remote procedure call, Real-Time transport protocols, Introduction to TCP, The TCP Service Model, The TCP Protocol, The TCP Segment Header, TCP Connection Establishment, TCP Connection Release, SCTP

**UNIT V**

**Application Layer**: Domain Name Space (DNS), SNMP, Electronic mail: MIME, SMTP, IMAP, WWW, FTP, HTTP

**Text Books:**

1) Computer Networks, Andrew S. Tanenbaum, David J. Wetherall, Pearson Education India; 5 edition, 2013

2) Data Communication and Networking ,Behrouz A. Forouzan, McGraw Hill, 5th Edition, 2012

**Reference Books:**

1) Computer Networks: A Systems Approach, LL Peterson, BS Davie, Morgan-Kauffman, 5th Edition, 2011.

2) Computer Networking: A Top-Down Approach JF Kurose, KW Ross, Addison-Wesley, 5th Edition, 2009

3) Data and Computer Communications, William Stallings, Pearson, 8th Edition, 2007

**e-Resources:**

1)https://nptel.ac.in/courses/106/105/106105183/

**List of Experiments**

|  |  |
| --- | --- |
| **Experiment No.** | **Name of the Experiment** |
| 1 | Implement the data link layer framing methods such as character stuffing and bit stuffing. |
| 2 | Write a C program to develop a DNS client server to resolve the given hostname |
| 3 | Implement on a data set of characters the three CRC polynomials – CRC-12, CRC-16 and CRC-CCIP. |
| 4 | Implement Dijkstra’s algorithm to compute the Shortest path in a graph. |
| 5 | Take an example subnet graph with weights indicating delay between nodes. Now obtain Routing table art each node using distance vector routing algorithm |
| 6 | Take an example subnet of hosts. Obtain broadcast tree for it. |
| 7 | Write a client-server application for chat using UDP |
| 8 | Implement programs using raw sockets (like packet capturing and filtering) |
| 9 | Write a C program to perform sliding window protocol. |
| 10 | Get the MAC or Physical address of the system using Address Resolution Protocol |
| 11 | Simulate the Implementing Routing Protocols using border gateway protocol(BGP) |
| 12 | Simulate the OPEN SHORTEST PATH FIRST routing protocol based on the cost assigned to the path. |
| **Additional Experiments** | |
| 13 | Implementation of Character Stuffing |
| 14 | Implementation of CRC (Cyclic Redundancy Check) |

**Course Outcomes of Associated Course Related To Lab:**

C326.1 Understand the concepts of LAN technologies and OSI, TCP/IP models and implement basic commands, system calls. (L2 Understand, L3 Apply)

C326.2 Discuss various error control techniques and data link protocols. (L2 Understand)

C326.3 Understand routing and congestion control algorithms. (L2 Understand)

C326.4 Design applications using internet protocols. (L6 Create)

**Mapping Of Co’s With Lab Experiments:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EXPERIMENT | C326.1 | C326.2 | C326.3 | C326.4 |
| EX1 | 3 |  |  |  |
| EX2 |  |  |  | 3 |
| EX3 |  |  |  | 3 |
| EX4 | 3 |  |  |  |
| EX5 | 3 |  |  |  |
| EX6 | 3 |  |  |  |
| EX7 |  |  |  | 3 |
| EX8 |  |  | 3 |  |
| EX9 |  |  |  | 3 |
| EX10 |  |  |  | 3 |
| EX11 |  |  |  | 3 |
| EX12 |  |  |  | 3 |
| EX13 |  | 3 |  |  |
| EX14 |  | 3 |  |  |

Level of Mapping: 1 – Slightly 2 – Moderate 3 – Highly

**1) Implement the data link layer framing methods such as character stuffing and bit stuffing**

**DESCRIPTION**

**Data Link Layer Functions**

Concerned with reliable, error-free and efficient communication

between *adjacent* machines in the network through the following functions:

#### Data Framing:

The term “frame” refers to a small block of data used in a specific network. The data link layer groups raw data bits to/from the physical layer into discrete frames with error detection/correction code bits added. Framing methods:

* + - Character count.
    - Starting and ending characters, with character stuffing.

Starting and ending flags with bit stuffing.

* + - Physical layer coding violations.

#### Error Detection/Correction:

* + - Error Detection:
      * Include enough redundant information in each frame to allow the receiver to deduce that an error has occurred, but not which error and to request a retransmission.
      * Uses error-detecting codes.
    - Error Correction:
      * Include redundant information in the transmitted frame to enable the receiver not only to deduce that an error has occurred but also correct the error.
      * Uses error-correcting codes.

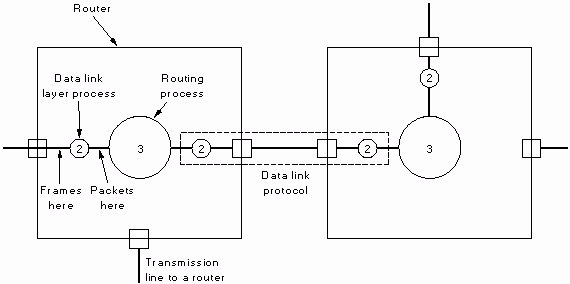
#### Services to the network layer:

* + - Unacknowledged connectionless service:
      * Independent frames sent without having the destination acknowledge them.
      * Suitable for real-time data such as speech and video where transmission speed is more important than absolute reliability.
      * Utilized in most LANS.
    - Acknowledged connectionless service:
      * Each frame sent is acknowledged by the receiver.
      * Acknowledgment at the layer level is not essential but provides more efficiency than acknowledgment at higher layers (transport) which is done only for the whole message.
      * A lost acknowledgment may cause a frame to be sent and received several times.
    - Acknowledged connection-oriented service:
      * The sender and receiver establish a connection before any data transmission.
      * The message is broken into numbered frames.
      * The data link guarantees that each frame sent is received exactly once and in the right order.

#### Flow control

Protocols to control the rate the sender transmits frames at a rate acceptable to the receiver, and the ability to retransmit lost or damaged frames. This insures that slow receivers are not swamped by fast senders and further aids error detection/correction.

* + - Several flow control protocols exist, but all essentially require a form of feedback to make the sender aware of whether the receiver can keep up.
      * Stop-and-wait Protocols:
        + A positive acknowledgment frame is send by the receiver to indicate that the frame has been received and to indicate being ready for the next frame.
        + Positive Acknowledgment with Retransmission (PAR); uses timeouts
      * Sliding Window Protocols:
        + Data frames and acknowledgement frames are mixed in both directions.
        + Frames sent contain sequence numbers
        + Timeouts used to initiate retransmission of lost frames.

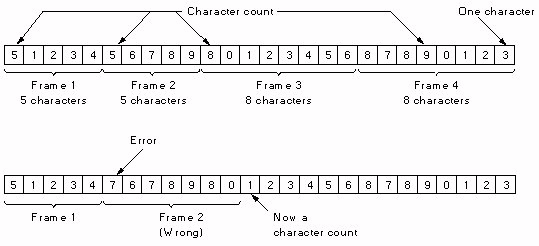


**Placement of The Data Link Protocol**

**Data Link Layer: Framing**

### Data Link Layer: Framing

1. **The character count method:**
   * The frame header includes the count of characters in the frame
   * A transmission error can cause an incorrect count causing the source and destination to get out of synchronization
   * Rarely used in actual data link protocols



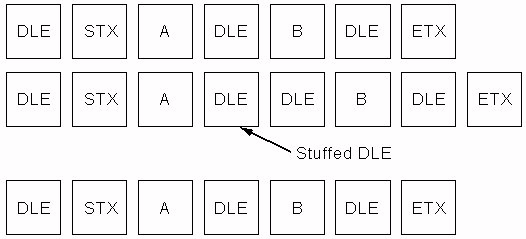
**A character stream with no errors**

**A character stream with errors**

### Using Starting and ending characters, with character stuffing:

* Each frame starts with the ASCII character sequence DLE (Data Link Escape) and STX (Start of TeXt) and ends with DLE ETX (End of TeXt)
* When binary data is transmitted where (DLE STX or DLE ETX) can occur in data, character stuffing is used (additional DLE is inserted in the data).
* Limited to 8-bit characters and ASCII.

**Network Layer Data at the sender**



**Data after character stuffing by the Data Link Layer at the sender**

**Network Layer Data at the Receiver**

### Bit-Oriented Using Start/End Flags:

1. Each frame begins and ends with 01111110
2. Bit stuffing: After each five consecutive ones in a data a zero is stuffed
3. Stuffed zero bits are removed by the data link layer at receiving end.

## The Original Data

PROCEDURES

**Step-1:** Print the menu as follows Character count -1 Character stuffing – 2 Bit stuffing – 3 Exit -

**Step-2:** Read the Choice

**Step-3:** If the choice is 1, do the following

* Read the Character sequence
* Use random( ) function for frame length If random() function returns 5 then

Take the first 4 characters from the character stream and print the frame (for example: if first 4 characters are a,b,c,d then the frame is 5 a b c d)

Repeat the above 2 steps until the character sequence ends

**Step-4:** If the choice is 2, do the following

* Read the data units with delimiters DLE STX and DLE ETX
* Scan the data for any occurrences of delimiters like DLE
* If DLE found in the sequence, then stuff the another DLE
* Repeat the above 2 step until data stream ends.

**Step-5:** If the choice is 3, do the following

* Read the bit stream
* Scan the bit stream for continuous 6 1’s
* If it is found then insert 0 after 5th 1
* Finally print the bit stream with delimiter 0 1 1 1 1 1 1 0 (i.e. 0 1 1 1 1 1 1 1 0 bit stream 0 1 1 1 1 1 1 0 )

**Step-6:** If the choice is 4 , then call exit() function

**2) Write a C program to develop a DNS client server to resolve the given hostname.**

### AIM: To write a C program to develop a DNS client server to resolve the given hostname.

**ALGORITHM:**

1. Create a new file. Enter the domain name and address in that file.
2. To establish the connection between client and server.
3. Compile and execute the program.
4. Enter the domain name as input.
5. The IP address corresponding to the domain name is display on the screen
6. Enter the IP address on the screen.
7. The domain name corresponding to the IP address is display on the screen.
8. Stop the program.

**3) Implement on a data set of characters the three CRC polynomials – CRC-12, CRC-16 and CRC-CCIP.**

**AIM :To write a program for implementing on a data set characters the three CRC polynomials – CSR12, CRC16 and CRC CCIP**

**DESCRIPTION**

## Data Link Layer: Error Detection/Correction

**.** Simplest error detection : Parity bits and checksum (sum of 1’s in Data).

**.** Error-detecting and -correcting codes:

* *m* data bits + *r* redundant bits added.
* *n* = *m* + *r* transmitted in frame.
* Only *2m* code words out of possible 2m*+r* words are legal.
* The Hamming distance --minimum number of positions any two legal code words differ-- of a code defines its error detection/correction ability.
* To detect *d* errors code Hamming distance = *d* + 1
* To correct *d* errors code Hamming distance = *2d* + 1
* Some codes are more suitable to correct burst errors rather than isolated errors.
* Polynomial codes: Cyclic Redundancy Check (CRC) Codes, are characterized by a generating polynomial G(X)

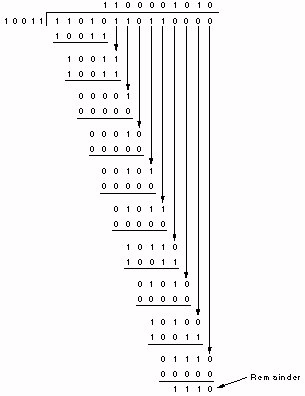
## Cyclic Redundancy Check (CRC)

* Based on polynomial arithmetic over finite field.
* View m-bit string a m-1a m-2 . . . a0 as a polynomial of degree m-1: M(x) = a m-1 x m-1 + a m-2 x m-2 + …. + a0
* Select a generating polynomial G(x) of degree r.
* Let R(x) be the remainder of xr M(x) / G(x)
* The code word T(x) of length m + r bit generated is then given by: T(x) = xr M(x) - R(x)
* Assume code word T(x) is transmitted, but T(x) + E(x) arrives at the receiver:
  + If E(x) = 0 then no transmission errors and T(x)/G(x) = 0
  + If E(x) ¹ 0 then transmission error(s) occurred and: [T(x) + E(x)] / G(x) ¹ 0

## Calculation of Polynomial Code (CRC) Checksum

1. For degree of generating polynomial G(x) = *r* , append *r* zero bits to low-order of frame. The frame now has *m+r* bits.
2. Divide the bit string corresponding to G(X) into the bit string xrM(x) mod(2)
3. Subtract the remainder R(x) from the bit string xrM(x) mod(2)

Frame: 1 1 0 1 0 1 1 0 1 1



Generator: 1 0 0 1 1

G(X) = X4 + X + 1

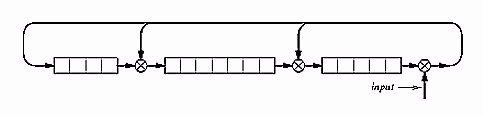
Message after appending four 0’s: 1 1 0 1 0 1 1 0 1 1 0 0 0 0

Remainder: 1110 Transmitted Frame:

1 1 0 1 0 1 1 0 1 1 1 1 1 0

## Hardware Computation of CRC

#### For G(x) = x16 + x12 + x5 + 1



**An Example Frame Format with CRC bits**



## Common CRC Generator Polynomials

* CRC-32: x32 + x 26 + x 23 + x22 + x16 + x12 + x11 +

x10 + x8 + x7 + x5 + x4 + x2 + x + 1 Used in FDDI, Ethernet.

* CRC-CCITT: x16 + X12 + x5 + 1

Used in HDLC.

* CRC-8: x8 + x2 + x + 1

Used in ATM.

#### PROCEDURES

Step-1: Read the frame

Step-2: Read the generator polynomial

Step-3: find out the degree of the generator polynomial

Step-4: Append the number of the zero’s to the frame that number is equal to the degree of the polynomial

Step-5: Find out number of digits in the generator polynomial

Step-6: Repeat the following until the number of digits are exhausted

Step-7: If the frame is starting with 1 , then exclusive-or the frame with generator

Step-8: Check whether the result obtained in step 7 is starting with 1, If so exclusive-or the remainder with the generator

Step-9: If the result obtained in step7 is starting with 0, then exclusive –or the remainder(result) with zeros. The number of zeroes must be equal to the length of the generator.

**4) Implement Dijkstra’s algorithm to compute the Shortest path in a graph.**

**AIM:**To write a program for implementation of Dijkstra's Algorithm to compute the shortest path through a graph

#### DESCRIPTION

Dijkstra's algorithm solves the single-source shortest-path problem when all edges have non-negative weights. It is a greedy algorithm and similar to Prim's algorithm. Algorithm starts at the source vertex, s, it grows a tree, T, that ultimately spans all vertices reachable from S. Vertices are added to T in order of distance i.e., first S, then the vertex closest to S, then the next closest, and so on. Following implementation assumes that graph G is represented by adjacency lists.

**DIJKSTRA (G, w, s)**

* 1. INITIALIZE SINGLE-SOURCE (G, s)
  2. S ← { } // S will ultimately contains vertices of final shortest-path weights from s
  3. Initialize priority queue Q i.e., Q ← V[G]
  4. while priority queue Q is not empty do
  5. *u* ← EXTRACT\_MIN(Q) // Pull out new vertex

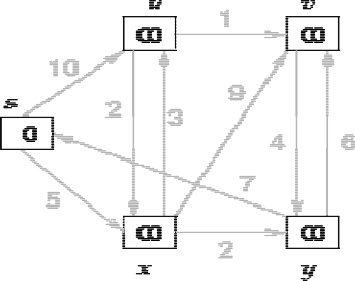
6. S ← S  {*u*}

// Perform relaxation for each vertex *v* adjacent to u

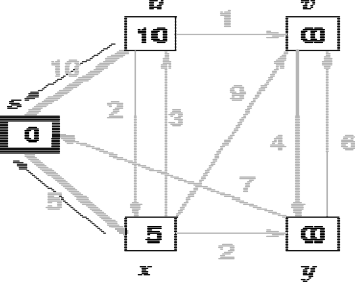
1. for each vertex v in Adj[u] do
2. Relax (*u*, *v*, *w*)

#### PROCEDURES

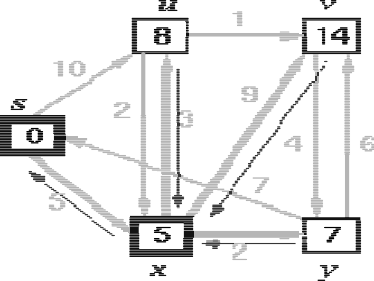
Step1. Given initial graph G=(V, E). All nodes have infinite cost except the source node, s, which has 0 cost.

****

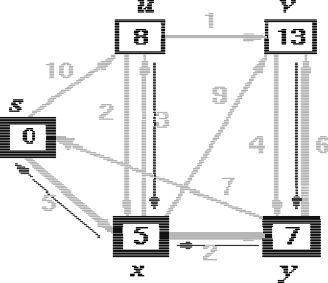
Step 2. First we choose the node, which is closest to the source node, s. We initialize d[s] to 0. Add it to S. Relax all nodes adjacent to source, s. Update predecessor (see red arrow in diagram below) for all nodes updated.



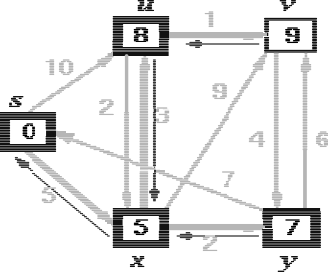
Step 3. Choose the closest node, x. Relax all nodes adjacent to node x. Update predecessors for nodes u, v and y (again notice red arrows in diagram below).



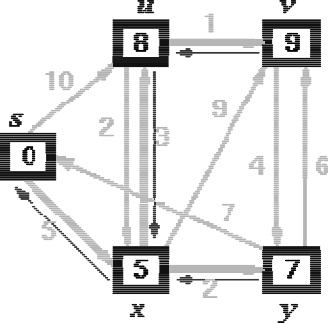
Step 4. Now, node y is the closest node, so add it to S. Relax node v and adjust its predecessor (red arrows remember!).



Step 5. Now we have node u that is closest. Choose this node and adjust its neighbor node v.



Step 6. Finally, add node v. The predecessor list now defines the shortest path from each node to the source node, s.



5) Take an example subnet graph with weights indicating delay between nodes. Now obtain Routing table art each node using distance vector routing algorithm

#### DESCRIPTION

Routing algorithms are a part of the network layer software whose responsibility is to decide what destination output line should be selected for successful journey completion of the packet from the source machine to destination machine. The network layer of ISO-OSI reference model is responsible for getting packets from the source to the destination. It uses routing algorithms to effectively use all communication lines and routers present in the communication subnet and decide which lines to use for forwarding incoming packets.

### Classification of Routing Algorithms

Routing algorithms are of classified on the basis of routing decisions made by them. They are either non adaptive or adaptive. Non adaptive algorithms do not change routing decisions on the basis of measurements of current traffic and topology, where as adaptive routing algorithms do change routing decisions on basis of measurements of current traffic and topology. Non adaptive algorithms are also called static algorithms and adaptive algorithms are also called dynamic algorithms.

### Types of Routing Algorithms Shortest Path Routing

Shortest path routing is a static (non adaptive) routing algorithm. In this algorithm a graph of the communication subnet is created with each node representing a router and each edge representing a communication line (link). Finding the shortest path between two nodes is done on the basis of some type of measurements. These measurements are called metrics. Some of the metrics used are number of hops, physical distance, mean queuing and transmission delays. Metrics like transmission delays and queue length are measured hourly for standard test packets sent out on each line of the communication subnet. Shortest path can be calculated on the basis of any one of the criteria or a combination of criteria. The most simple and widely used algorithm for finding the shortest path is the Dijkstra’s algorithm.

### Distance Vector Routing

Most computer networks in operation today use dynamic routing algorithms rather than static routing algorithms. One of the widely used dynamic algorithms is Distance Vector routing, which is also known as Bellman Ford algorithm. A table maintained by a router is called a vector. Vectors contain information on how to get to the destination using the best possible path to get to it. It also contains an entry for each router in the subnet. Each of the routing tables is updated continuously. This can be done by exchanging information with the neighboring routers. For measuring the optimal distance between each node, metrics are used, similar to ones described above in the classification section of the article. The optimal path is followed till the packet reaches the destination machine.

***Distance vector algorithms*** use the Bellman-Ford algorithm. This approach assigns a number, the cost, to each of the links between each node in the network. Nodes will send information from point A to point B via the path that results in the lowest total cost (i.e. the sum of the costs of the links between the nodes used). The algorithm operates in a very simple manner. When a node first starts, it only knows of its immediate neighbours, and the direct cost involved in reaching them. (This information, the list of destinations, the total cost to each, and the next hop to send data to get there, makes up the routing table, or distance table.) Each node, on a regular basis, sends to each neighbour its own current idea of the total cost to get to all the destinations it knows of. The neighbouring node(s) examine this information, and compare it to what they already 'know'; anything which represents an improvement on what they already have, they insert in their own routing table(s). Over time, all the nodes in the network will discover the best next hop for all destinations, and the best total cost.

When one of the nodes involved goes down, those nodes which used it as their next hop for certain destinations discard those entries, and create new routing-table information. They then pass this information to all adjacent nodes, which then repeat the process. Eventually all the nodes in the network receive the updated information, and will then discover new paths to all the destinations which they can still "reach".

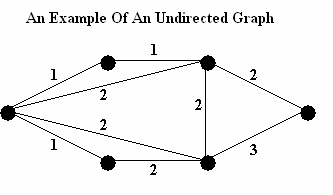
**6) Take an example subnet of hosts. Obtain broadcast tree for it.**

#### DESCRIPTION

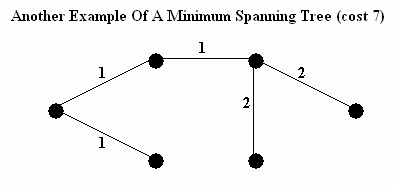
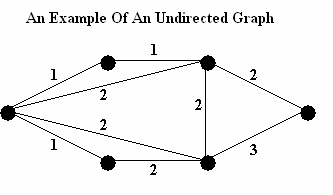
**Minimum Spanning Trees:**

Given a connected, undirected graph

G = <N,E>

where each edge has an associated 'length' (or 'weight').

We want a subset, T, of edges, E, such that the graph remains connected if only the edges in T are used, and the sum of the lengths of edges in T is as small as possible. Such a subgraph must be a tree, and is called a *Minimum Spanning Tree*. For the above graph, the following are both minimum spanning trees (cost 7).



**PROBLEM :** Devise an algorithm to find a minimum spanning tree.

#### Kruskal's Algorithm:

Greedy algorithm to find minimum spanning tree. Want to find set of edges T.

* Start with T = EMPTY SET
* Keep track of connected components of graph with edges T
* Initially components are single nodes
* At each stage, add the cheapest edge that connects two nodes not already connected

#### PROCEDURES

**Kruskal's Algorithm**

* Step 1 : Find the cheapest edge in the graph (if there is more than one, pick one at random). Mark it with any given colour, say red.
* Step 2 : Find the cheapest unmarked (uncoloured) edge in the graph that doesn't close a coloured or red circuit. Mark this edge red.
* Step 3 : Repeat Step 2 until you reach out to every vertex of the graph (or you have N ; 1 coloured edges, where N is the number of Vertices.) The red edges form the desired minimum spanning tree.

#### Prim's Algorithm

* Step 0

Pick any vertex as a starting vertex. (Call it S). Mark it with any given colour, say red.

* Step 1

Find the nearest neighbour of S (call it P1). Mark both P1 and the edge SP1 red. cheapest unmarked (uncoloured) edge in the graph that doesn't close a coloured circuit. Mark this edge with same colour of Step 1.

* Step 2

Find the nearest uncoloured neighbour to the red subgraph (i.e., the closest vertex to any red vertex). Mark it and the edge connecting the vertex to the red subgraph in red.

* Step 3

Repeat Step 2 until all vertices are marked red. The red subgraph is a minimum spanning tree.

* Interactive Prim's Algorithm

**7) Write a client-server application for chat using UDP**

### AIM: To write a client-server application for chat using UDP

**ALGORITHM: CLIENT**

1. Include necessary package in java
2. The client establishes a connection to the server.
3. The client accept the connection and to send the data from client to server and vice versa.
4. The client communicate the server to send the end of the message.
5. Stop the program.

### ALGORITHM: SERVER

1. Include necessary package in java
2. The server establishes a connection to the client.
3. The server accept the connection and to send the data from server to client and vice versa.
4. The server communicate the client to send the end of the message
5. Stop the program.

**8) Implement programs using raw sockets (like packet capturing and filtering)**

**ALGORITHM :**

1. Start the program and to include the necessary header files
2. To define the packet length
3. To declare the IP header structure using TCPheader
4. Using simple checksum process to check the process
5. Using TCP \IP communication protocol to execute the program
6. And using TCP\IP communication to enter the Source IP and port number and Target IP address and port number.
7. The Raw socket () is created and accept the Socket ( ) and Send to ( ), ACK
8. Stop the program

**11) Simulate the Implementing Routing Protocols using border gateway protocol(BGP)**

### AIM: To simulate the Implementing Routing Protocols using border gateway protocol(BGP)

**ALGORITHM:**

1. Read the no. of nodes n
2. Read the cost matrix for the path from each node to another node.
3. Initialize SOURCE to 1 and include 1
4. Compute D of a node which is the distance from source to that corresponding node.
5. Repeat step 6 to step 8 for n-l nodes.
6. Choose the node that has not been included whose distance is minimum and include that node.
7. For every other node not included compare the distance directly from the source with the distance to reach the node using the newly included node
8. Take the minimum value as the new distance.
9. Print all the nodes with shortest path cost from source node

12) Simulate the OPEN SHORTEST PATH FIRST routing protocol based on the cost assigned to the path

### AIM: To simulate the OPEN SHORTEST PATH FIRST routing protocol based on the cost assigned to the path.

**ALGORITHM:**

1. Read the no. of nodes n
2. Read the cost matrix for the path from each node to another node. 3.Initialize SOURCE to 1 and include 1
3. Compute D of a node which is the distance from source to that corresponding node.
4. Repeat step 6 to step 8 for n-l nodes.
5. Choose the node that has not been included whose distance is minimum and include that node.
6. For every other node not included compare the distance directly from the source with the distance to reach the node using the newly included node

8.Take the minimum value as the new distance.

9.Print all the nodes with shortest path cost from source node