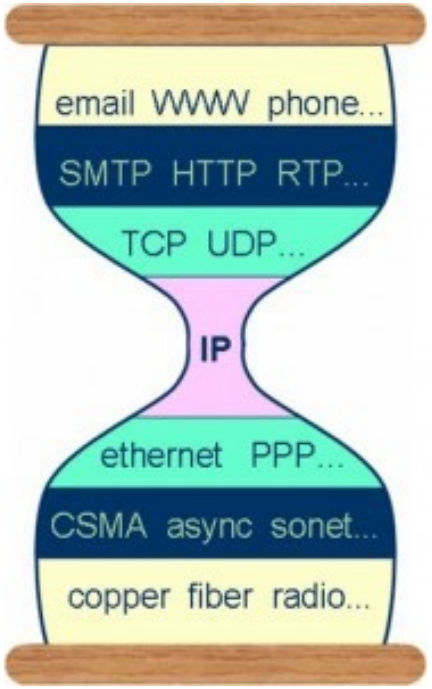
TECHNISCHE HOCHSCHULE DEGGENDORF

Datacenter Network Programming

Computer Networks – A review

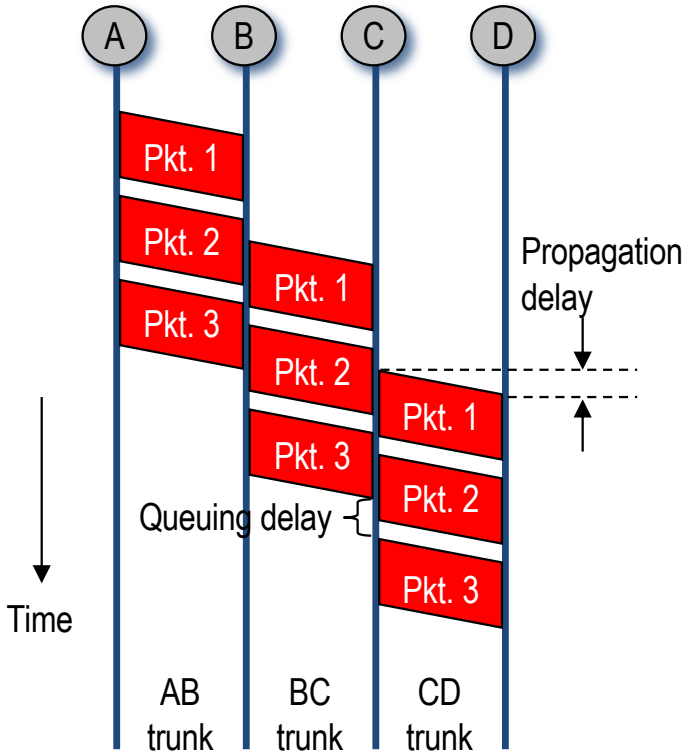
Review of Computer Networks 1

- Best-effort packet delivery
- Protocol layering for modularity
  - Internet hourglass design
- Relationships between layers
  - Naming and addressing
  - Directories and routing
- Scalability
  - Through hierarchy and indirection



Best-Effort Packet Switching

- Packet switching
  - Divide data into packets
  - Packets travel separately
  - Enables statistical multiplexing
- Best-effort delivery
  - Packets may be lost, delayed, out-of-order
  - Simplify network design and failure handling
  - Build timely, ordered, reliability delivery on top

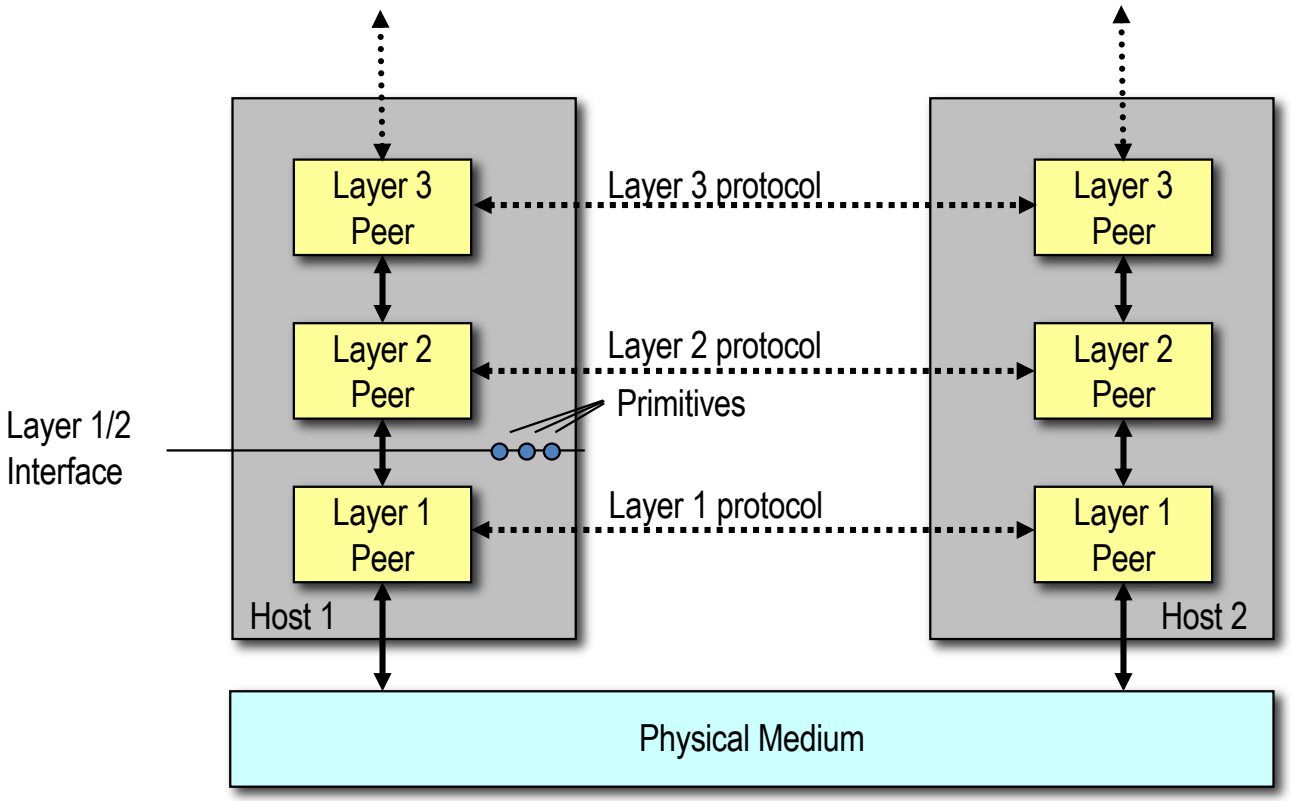


Layered Network Architecture

- Most networks are organized in *layers*.
- Each layer offers *services* to the higher layers.
- The service *primitives* are accessed via an *interface*.
- *Peers* are the entities of corresponding layers on different machines.

- Peers communicate (virtually) using *protocols*.
- The protocol defines the *meaning* of information exchanged.
- A set of layers and protocols is called a *network architecture*.
- The list of protocols used by a system is called the *protocol stack*.
- *Messages* are created at the highest layer.
- Each layer adds a *header* with control information in front of the *payload*.

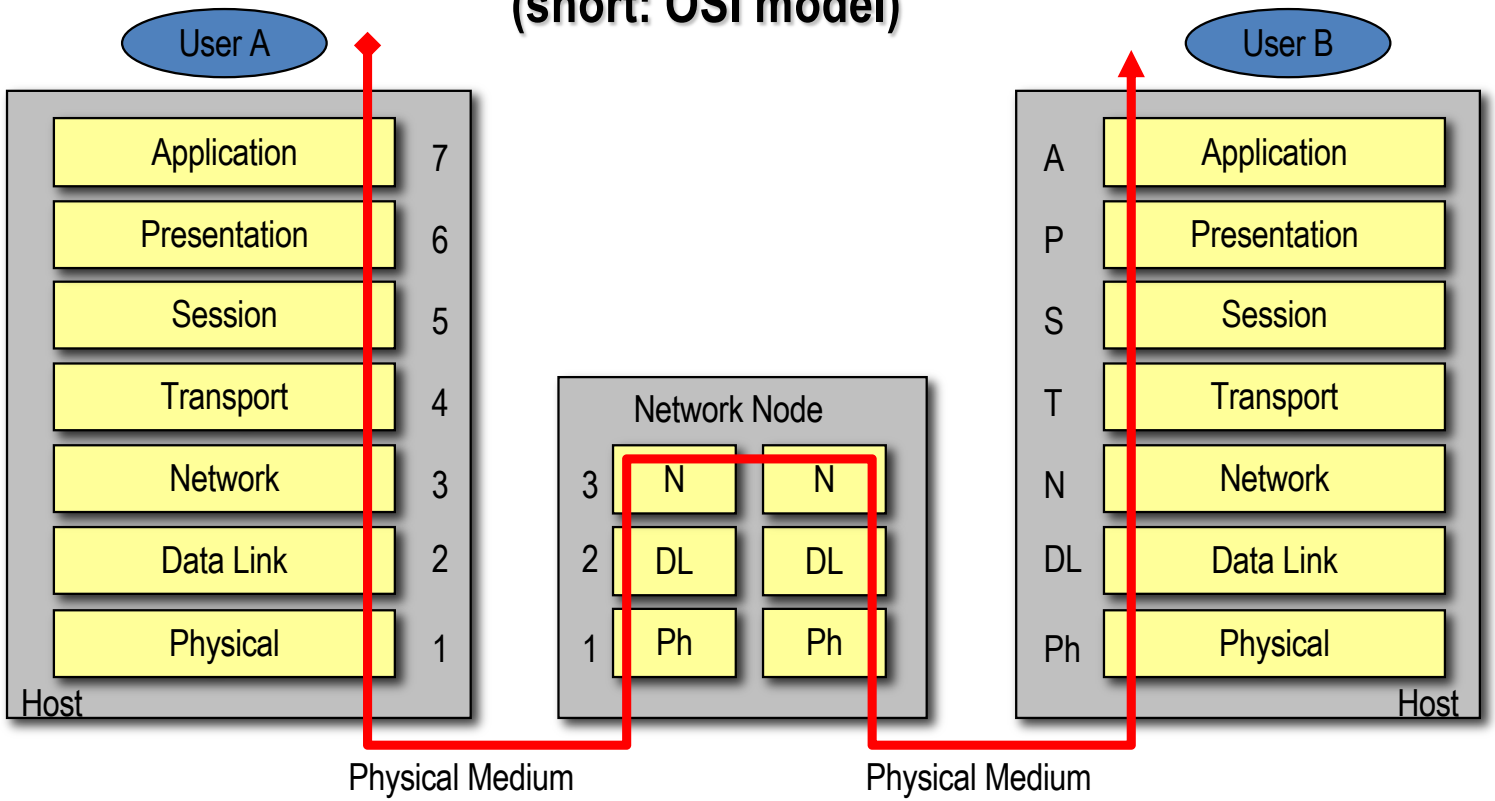
Layered Network Architecture



- **This allows for a reduced design complexity**
- **Protocols and transmission systems can be easily exchanged**

The ISO/OSI Model

- **In 1983 the International Standards Organization (ISO) developed the *Open Systems Interconnection Reference Model* (short: OSI model)**



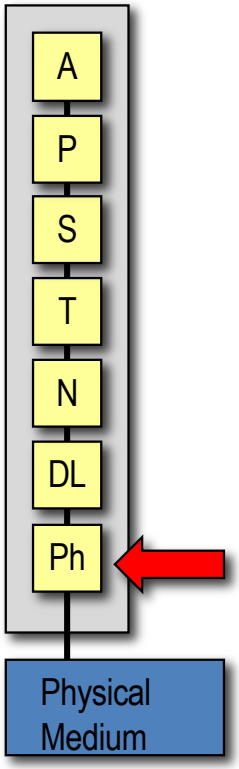
The ISO/OSI Model

Physical Layer:

- **Transmitting raw bits over a physical medium**
- **Agreement about mechanical, electrical and logical interfaces**

“How many volts should be used to represent a 1?”

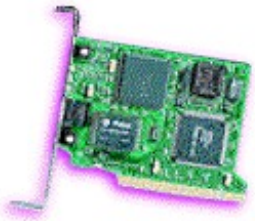
“How many pins should the connector have?”

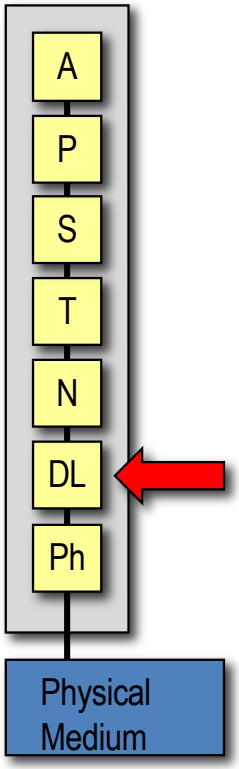


The ISO/OSI Model

Data Link Layer:

- Breaking the input stream into *data frames*
- Enable reliable connection through *channel coding* and *error correction*
- *Flow control* (sometimes)
- Broadcast networks with shared channels: Medium Access Control (*MAC*) sublayer
- In most systems a single host adapter card provides both the data link layer and the physical layer





The ISO/OSI Model

- MAC address (e.g., 00-15-C5-49-04-A9)
  - Numerical address used within a link
  - Unique, hard-coded in the adapter when it is built
  - Flat name space of 48 bits
- Hierarchical allocation
  - Blocks: assigned to vendors (e.g., Dell) by the IEEE
  - Adapters: assigned by the vendor from its block
- Broadcast address (i.e., FF-FF-FF-FF-FF-FF)
  - Send the frame to all adapters

The ISO/OSI Model

- Why not just IP-@?
- Links can support any network protocol
  - Not just for IP (e.g., IPX, Appletalk, X.25, …)
  - Different addresses on different kinds of links
- An adapter may move to a new location
  - So, cannot simply assign a static IP address
  - Instead, must reconfigure the adapter's IP address
- Must identify the adapter during bootstrap
  - Need to talk to the adapter to assign it an IP address

The ISO/OSI Model

- Dynamic Host Configuration Protocol (DHCP)
  - Broadcast “I need an IP address, please!”
  - Response “You can have IP address 1.2.3.4.”

The ISO/OSI Model

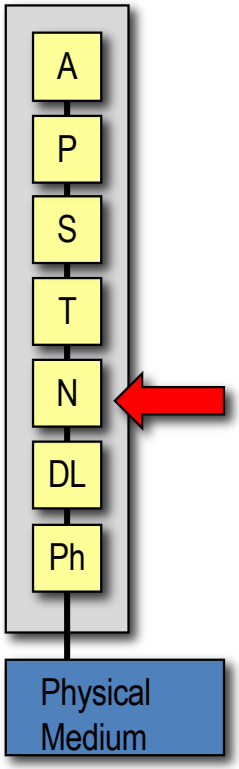
Data Link Layer: Who are you?

- Address Resolution Protocol (ARP)
  - Broadcast “who has IP address 1.2.3.6?”
  - Response “0C-C4-11-6F-E3-98 has 1.2.3.6!”

The ISO/OSI Model

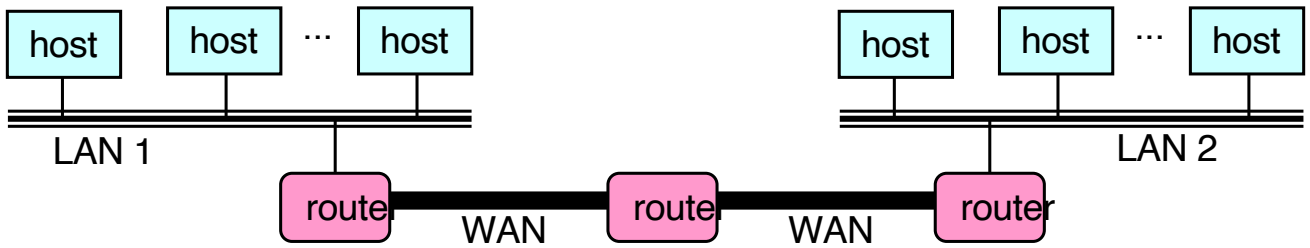
Network Layer:

- End-to-end *routing* among nodes within a packet-switched network
- Unique network-wide *addressing*
- *Interconnection* of heterogeneous networks
- *Accounting* / *Billing*
- *Flow Control*, *Segmentation*, *Multiplexing*, *Buffering*



The ISO/OSI Model

- Main challenges
  - Scalability
  - Autonomy

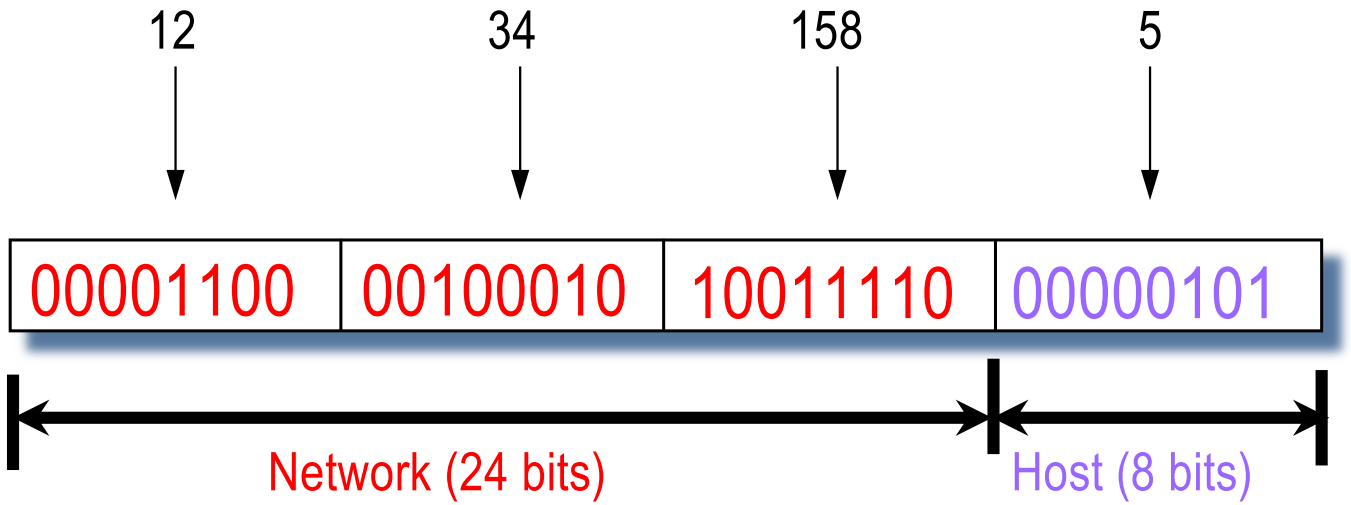


Autonomous systems

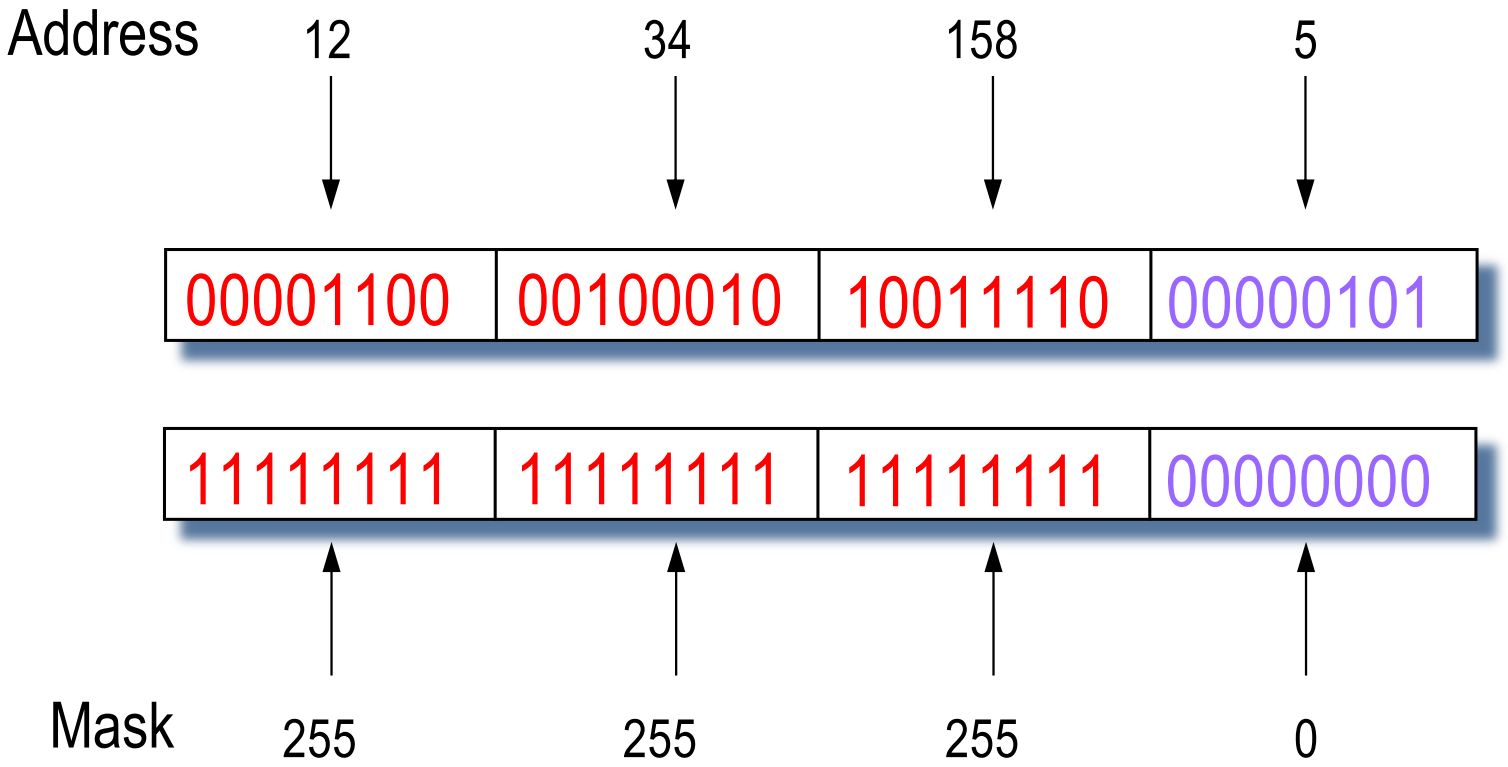
- Internet
  - Network of networks
  - Every computer is connected to an AS
- Autonomous System
  - (group) of networks with unified routing policy
    - IP address space plus list of connecting AS
    - Announced using BGP
  - Connect to several other AS
  - Post Office analogy
    - Packets hop rom AS to AS until destination AS, which delivers it internally

The ISO/OSI Model

- Network and host portions (left and right)
- 12.34.158.0/24 is a 24-bit **prefix** with 2^8 addresses

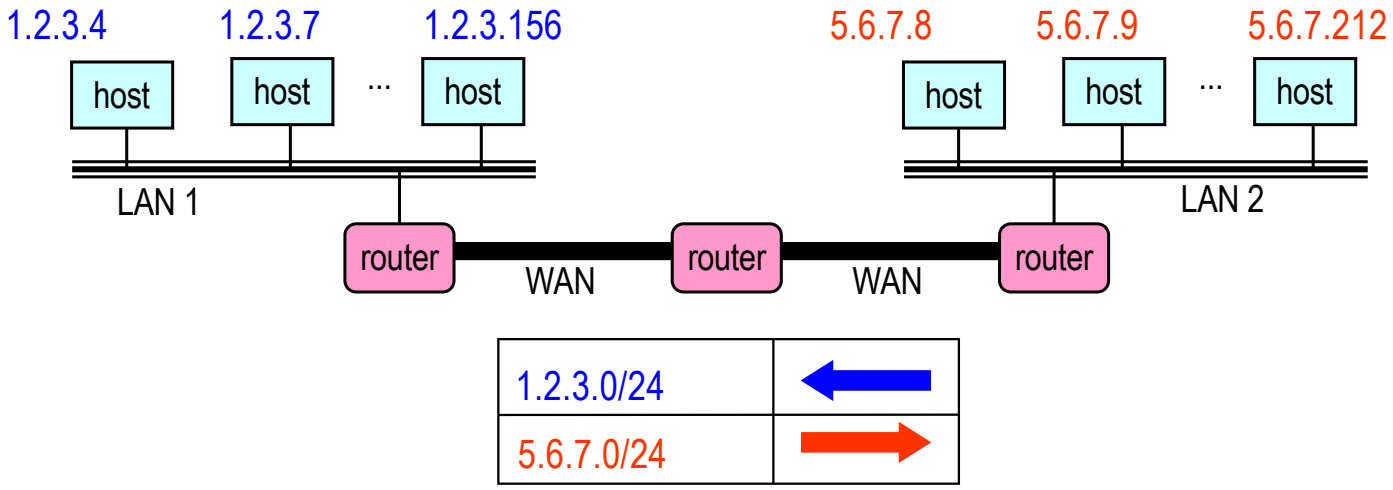


The ISO/OSI Model



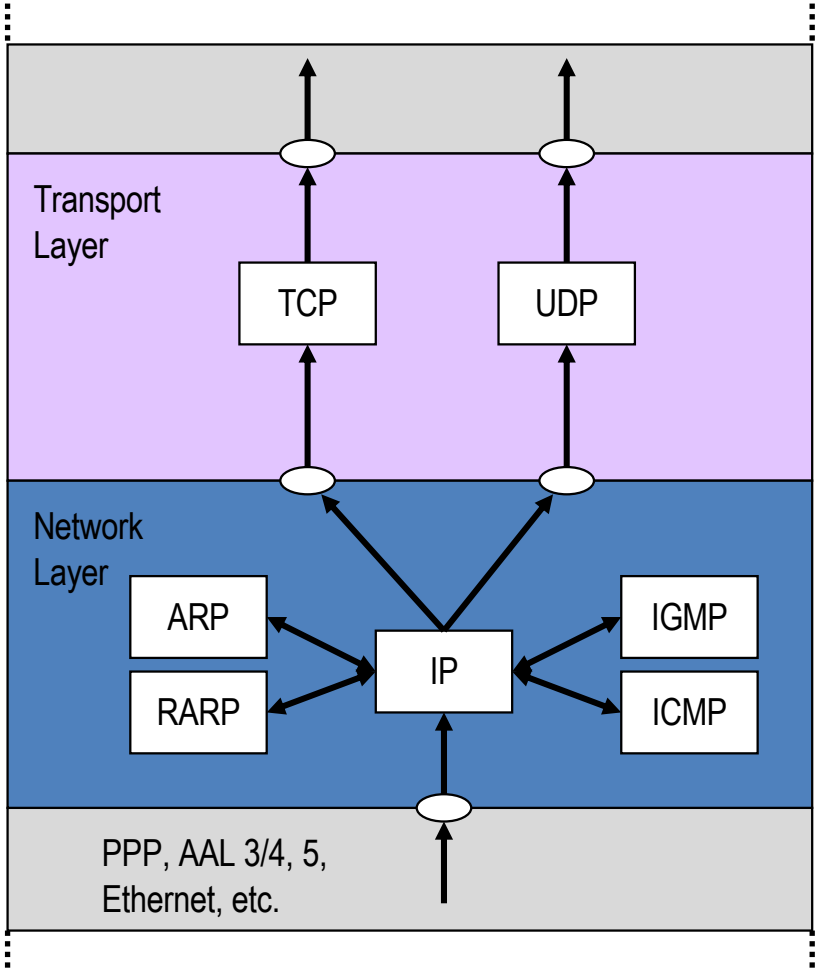
The ISO/OSI Model

- Number related hosts form a common subnet
  - 1.2.3.0/24 on the left LAN
  - 5.6.7.0/24 on the right LAN



forwarding table

The Internet Protocol (IP)



IP Protocol:
-Connectionless, no error detection/recovery
-Defining the datagram, which is the basic unit of transmission in the Internet
-Defining the Internet addressing scheme
-Moving data between the Network Access Layer and the Host-to-Host Transport Layer
-Routing datagrams to remote hosts
-Performing fragmentation and re-assembly of datagrams

ICMP:
Internet Control Message Protocol

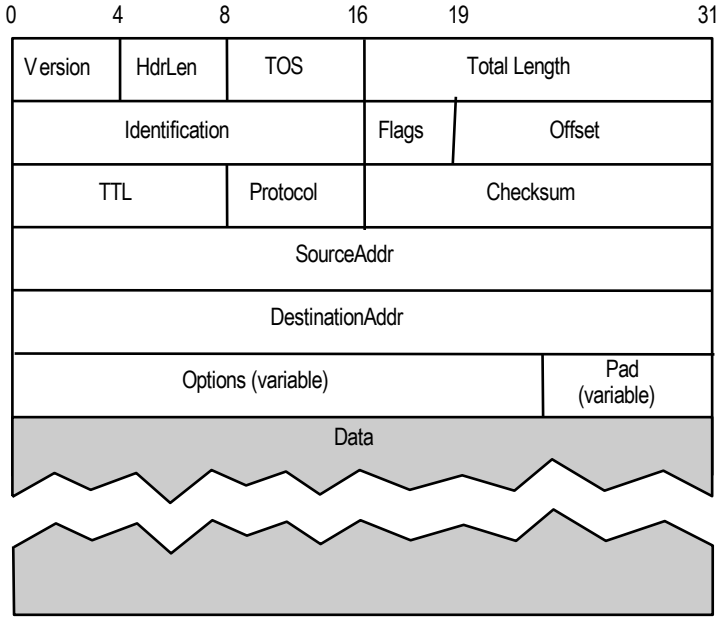
IGMP:
Internet Group Management Protocol

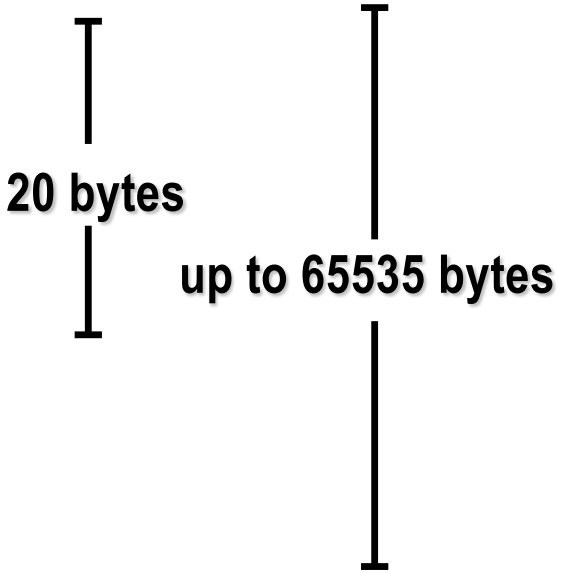
ARP:
Address Resolution Potocol

RARP:
Reverse Address Resolution Protocol

The Internet Protocol (IP)

IPv4 Header Structure:





- *Version*: 4 (IPv4), also 6 (IPv6)
- *HdrLen* (header length): in 32 bit words (5-15 = 20-60 Bytes)
- *TOS* (Type of Service): currently ignored by routers (-> DiffServ)
- *Total Length*: length of complete packet (max. 65.536)

The Internet Protocol (IP)

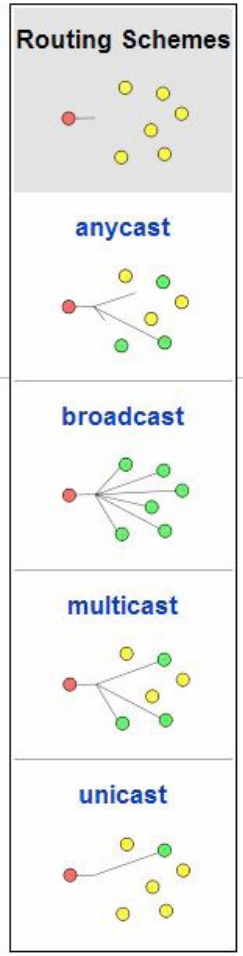
- *Identification*: Used for fragmentation
- *Flags*: Unused bit, *DF*: don ‘t fragment (fragment not allowed), *MF* : more fragments (packet not complete)
- *Fragment Offset*: Position of the fragment in the datagram
- *TTL* (time-to-live): Time in seconds (1-255), in practice: max. number of hops to pass
- *Protocol*: Defines the transport protocol used (RFC 1700), e.g. ICMP=1, IGMP=2, TCP=6, UDP=17
- *Header Checksum*: One ‘s complement arithmetic
- *Source Address*: Source network and host number
- *Destination Address*: Destination network and host number
- *Options*: Seldom use, optional mechanisms for security, routing, time-stamping, etc.
- *Pad*: The options field is padded out to a multiple of four Bytes

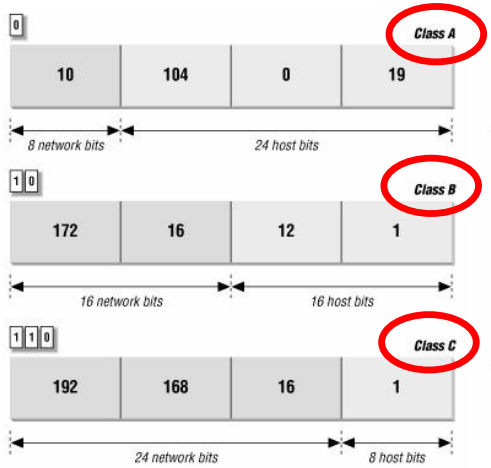
The Internet Protocol (IP)

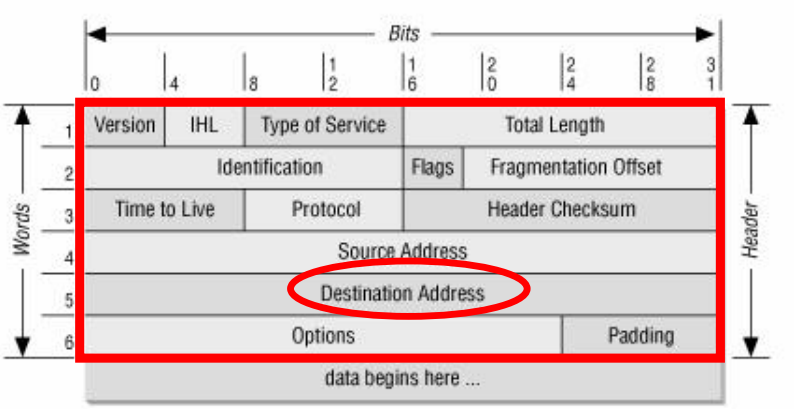
IPv4 Addressing Scheme:

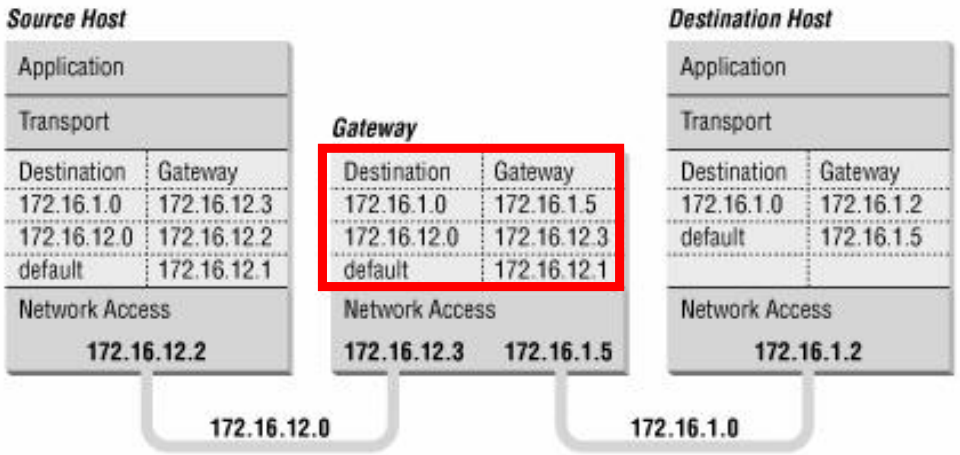
- **IPv4 addresses are 32-bit numbers assigned to each interface of a node**
- **Nodes with multiple interfaces are called ‚multi-homed ‘**
- **Addresses are globally unique identifiers**
- **They use a hierarchical structure: unicast addresses consist of *network* and *host* part**
- **Network numbers are assigned by the *NIC* (Network Information Center)**
- **Host numbers are assigned by the local administrator**

IP Routing Overview









Routing versus Routing protocols *(static versus dynamic)*

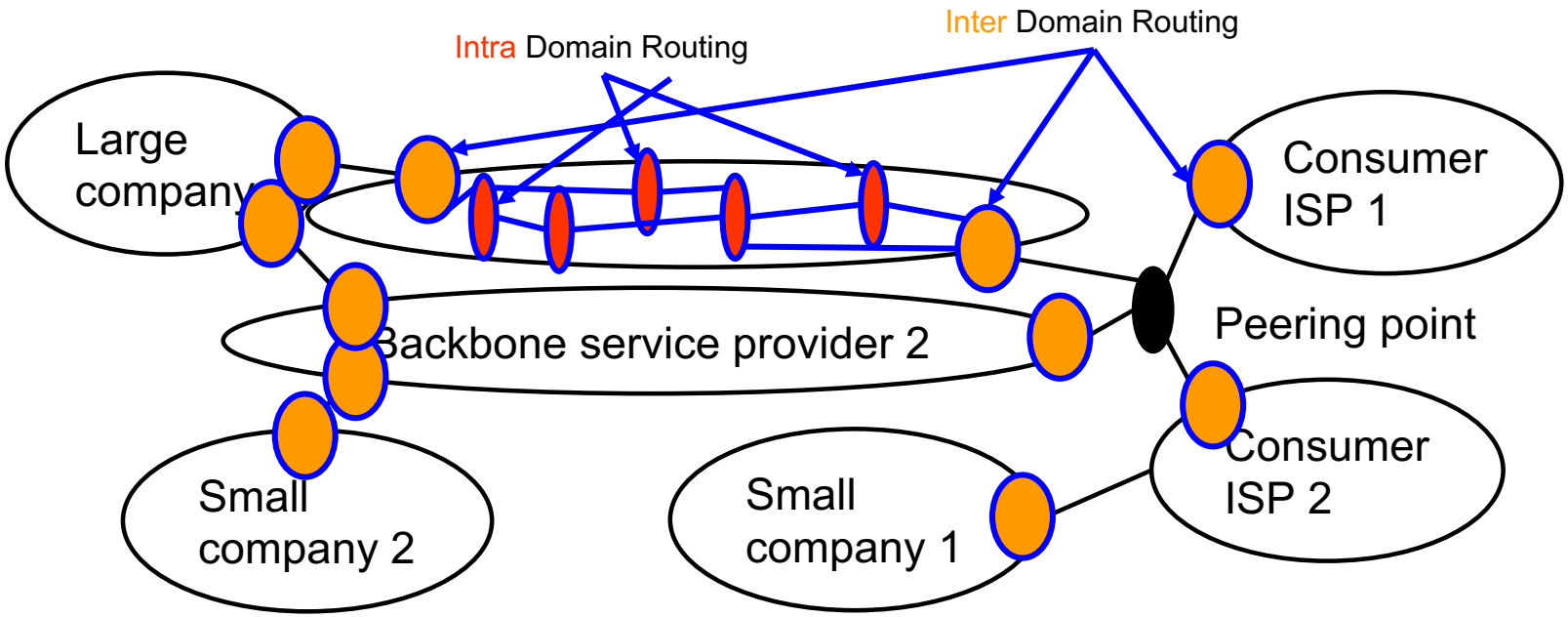
http://en.wikipedia.org/wiki/Routing

Tablebased IP Routing

http://www.unix.org.ua/orelly/networking/tcpip/ch01_05.htm

The Idea of Internet Routing

- Routing comprises:
  - Updating of routing tables according to routing algorithm
  - Exchange of routing information using routing protocol
  - Forwarding of data based on routing tables and addresses



Autonomous Systems in the IP World

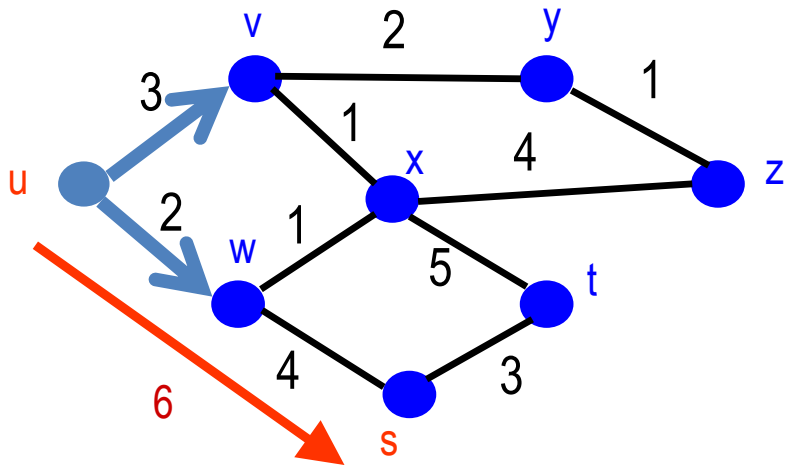
- Large organizations can own multiple networks that are under single administrative control
  - Forming autonomous system or routing domain
- Autonomous systems form yet another level of aggregating routing information
  - Give raise to inter- and intra-domain routing
- Inter-domain routing is hard
  - One organization might not be interested in carrying a competitor's traffic
  - Routing metrics of different domains cannot be compared
    - Only reachability can be expressed
  - Scalability: Currently, inter-domain routers have to know about 200,000 – 400,000 networks

Intra-domain Routing: OSPF

- The Internet's most prevalent intra-domain (= interior gateway) routing protocol: Open Shortest Path First (OSPF)
- Main properties:
  - Open, variety of routing distances, dynamic algorithm
  - Routing based on traffic type (e.g. real-time traffic uses different paths)
  - Load balancing: Also put some packets on the 2nd, 3rd best path
  - Hierarchical routing, some security in place, support tunneled routers in transit networks
- Essential operation: Compute shortest paths on graph abstraction of autonomous system
  - Link state algorithm

Example: Shortest-Path Routing

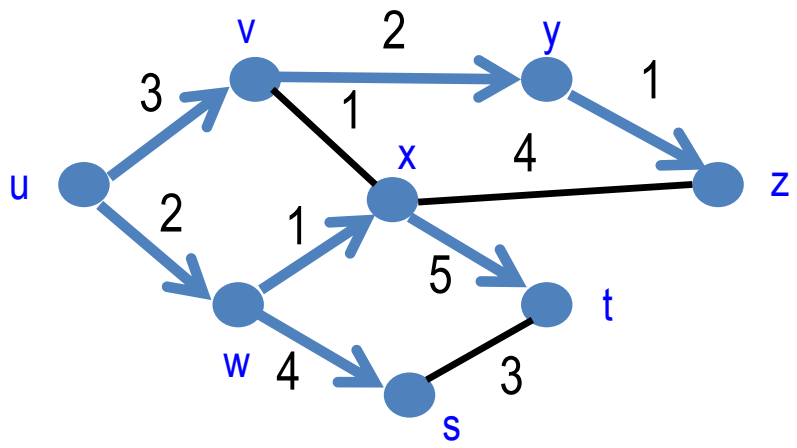
- Compute: *path costs* to all nodes
  - From a source u to all other nodes
  - Cost of the path through each link
  - Next hop along least-cost path to s



	link
v	(u,v)
w	(u,w)
x	(u,w)
y	(u,v)
z	(u,v)
s	(u,w)
t	(u,w)

Distributed Control Plane

- Link-state routing: OSPF, IS-IS → IGP (within single AS)
  - Find neighbors using periodic reachability protocol
  - Periodically flood its local (link, cost) pairs to all nodes
  - Each node computes shortest paths
  - Dijkstra's algorithm
  - Fill routing table



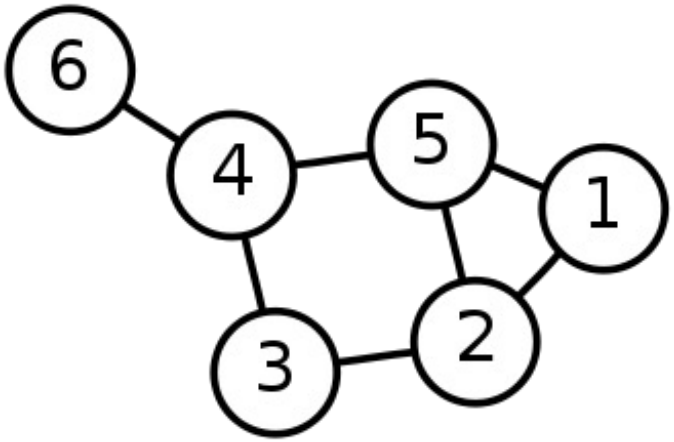
	link
v	(u,v)
w	(u,w)
x	(u,w)
y	(u,v)
z	(u,v)
s	(u,w)
t	(u,w)

Basic Ideas of Link State Routing

- Distributed, adaptive routing
- Algorithm:
  - Discovery of new neighbors
    - HELLO packet
  - Measurement of delay / cost to all neighbors
    - ECHO packet measures round trip time
  - Creation of link state packets containing all learned data
    - Sender and list of neighbors (including delay, age, ...)
    - Periodic or event triggered update (e.g. upon detecting new neighbors, line failure, ...)
  - Flooding of packet to all neighbors
    - Flooding, but with enhancements: Duplicate removal, deletion of old packets, ...
  - Each node independently calculates shortest path to all other routers
    - e.g. Dijkstra → Computing intensive, optimizations exist
    - https://www.youtube.com/watch?v=ZItmPVX_WIw

Single-Source Shortest Path Problem

- Single-Source Shortest Path Problem - The problem of finding shortest paths from a source vertex v to all other vertices in the graph.



Dijkstra's algorithm

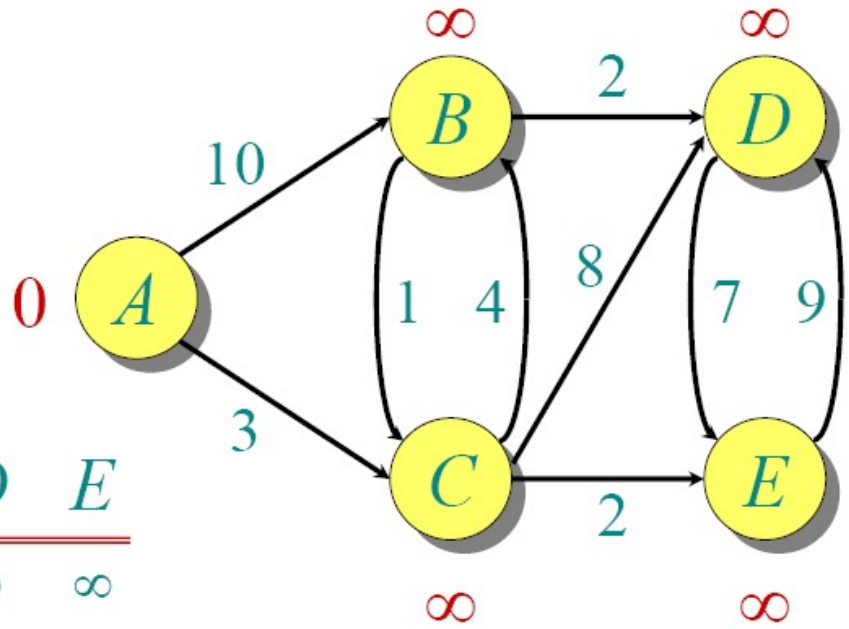
- Dijkstra's algorithm - is a solution to the single-source shortest path problem in graph theory.
- Works on both directed and undirected graphs. However, all edges must have nonnegative weights.
- Approach: Greedy
- Input: Weighted graph G={E,V} and source vertex v∈V, such that all edge weights are nonnegative
- Output: Lengths of shortest paths (or the shortest paths themselves) from a given source vertex v∈V to all other vertices

Dijkstra's algorithm - Pseudocode

```
dist[s] ←0                                  (distance to source vertex is zero)
for  all v ∈ V–{s}
        do  dist[v] ←∞                      (set all other distances to infinity)
S←∅                                         (S, the set of visited vertices is initially empty)
Q←V                                         (Q, the queue initially contains all vertices)
while Q ≠∅                                  (while the queue is not empty)
do   u ← mindistance(Q,dist)                (select the element of Q with the min. distance)
      S←SU{u}                               (add u to list of visited vertices)
       for all v ∈ neighbors[u]
             do  if   dist[v] > dist[u] + w(u, v)          (if new shortest path found)
                       then      d[v] ←d[u] + w(u, v)     (set new value of shortest path)
                   (if desired, add traceback code)
return dist
```

Dijkstra Animated Example

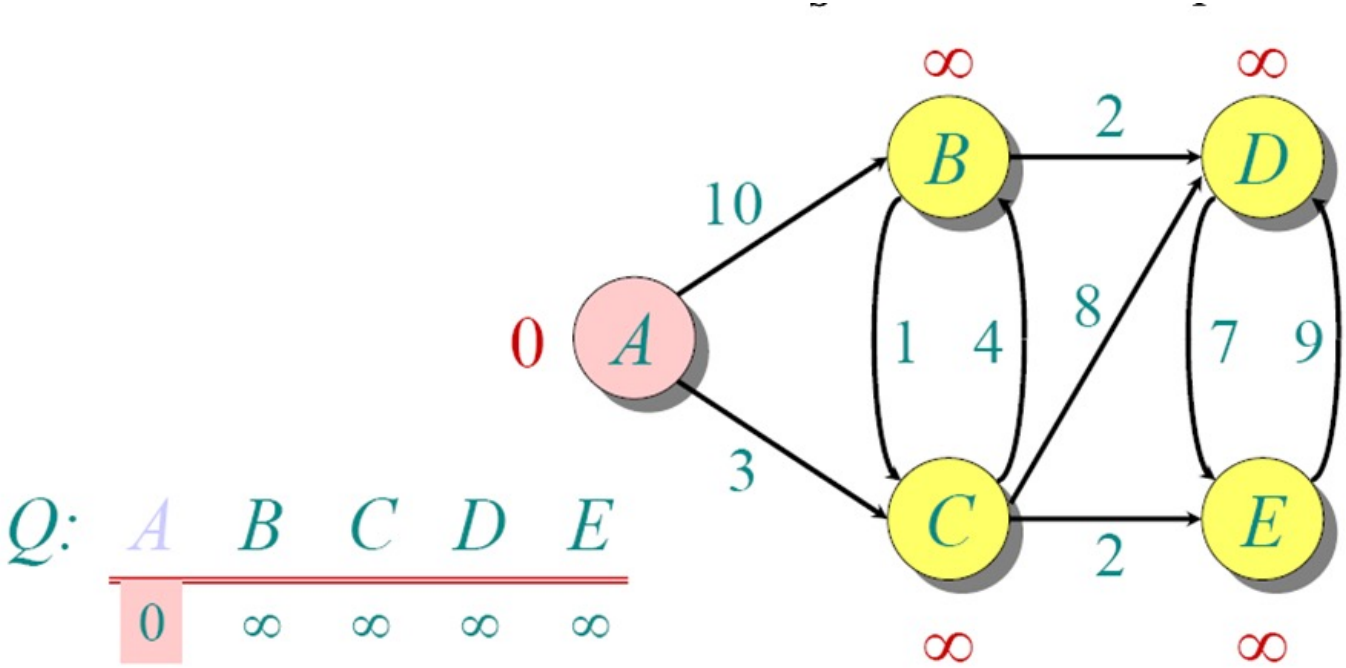
Initialize:



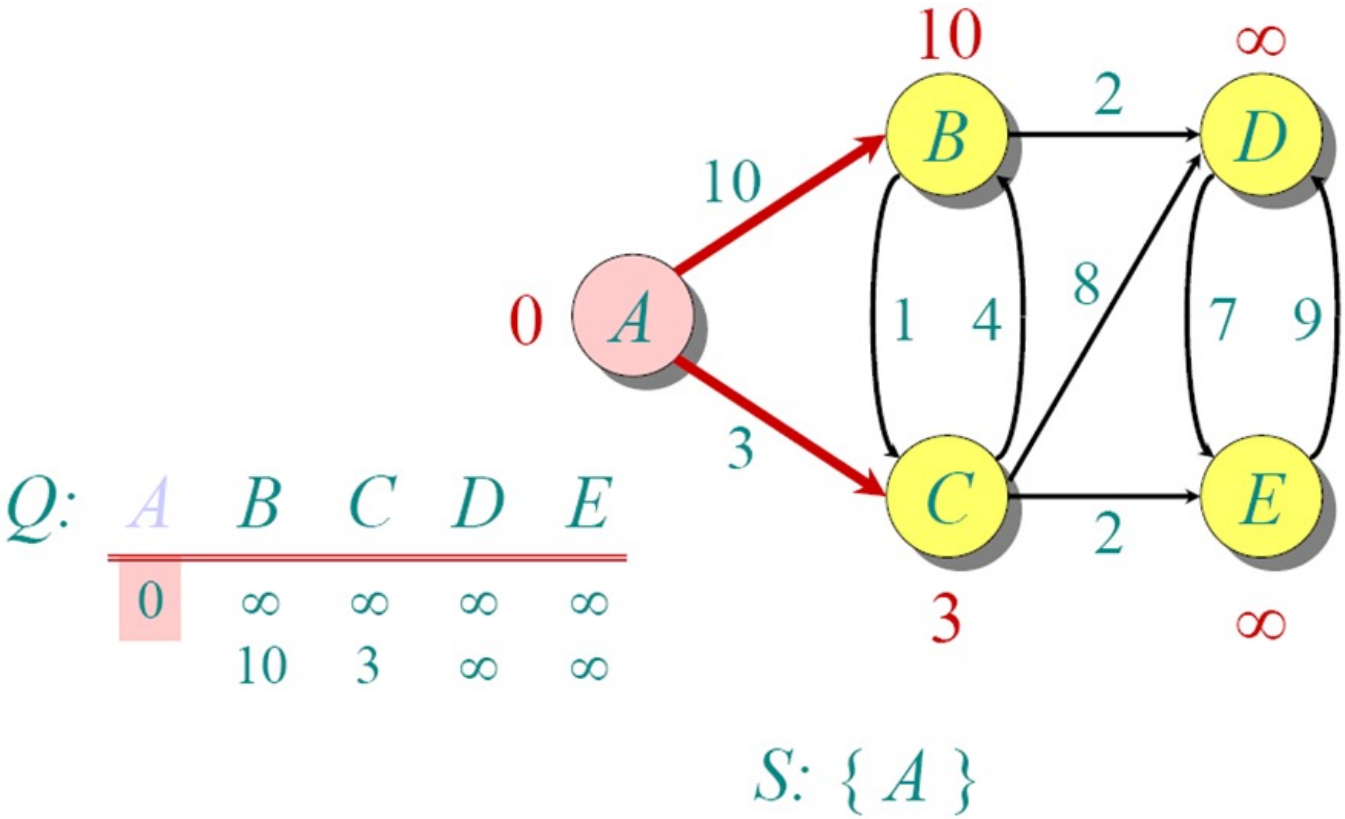
Q:	A	B	C	D	E
	0	∞	∞	∞	∞

S: {}

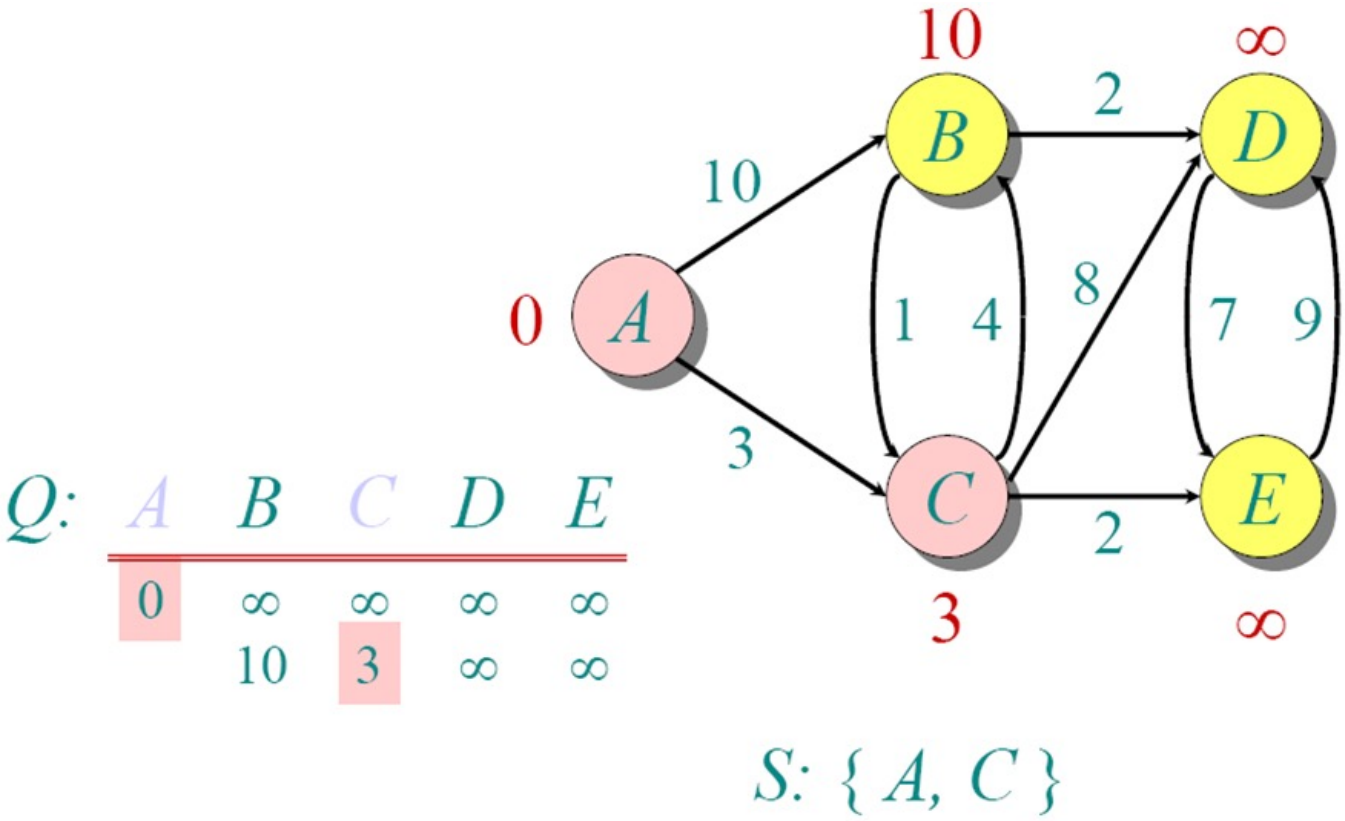
Dijkstra Animated Example



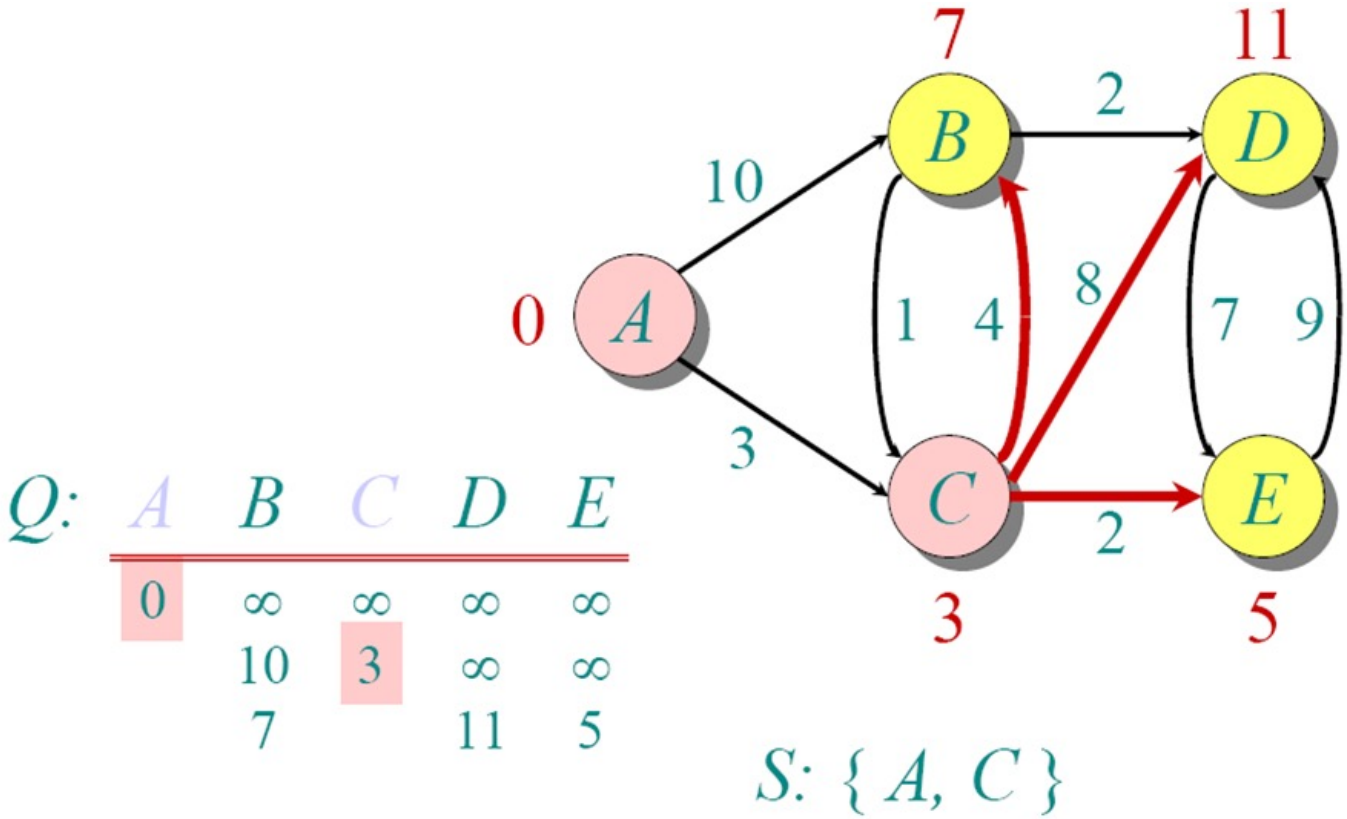
Dijkstra Animated Example



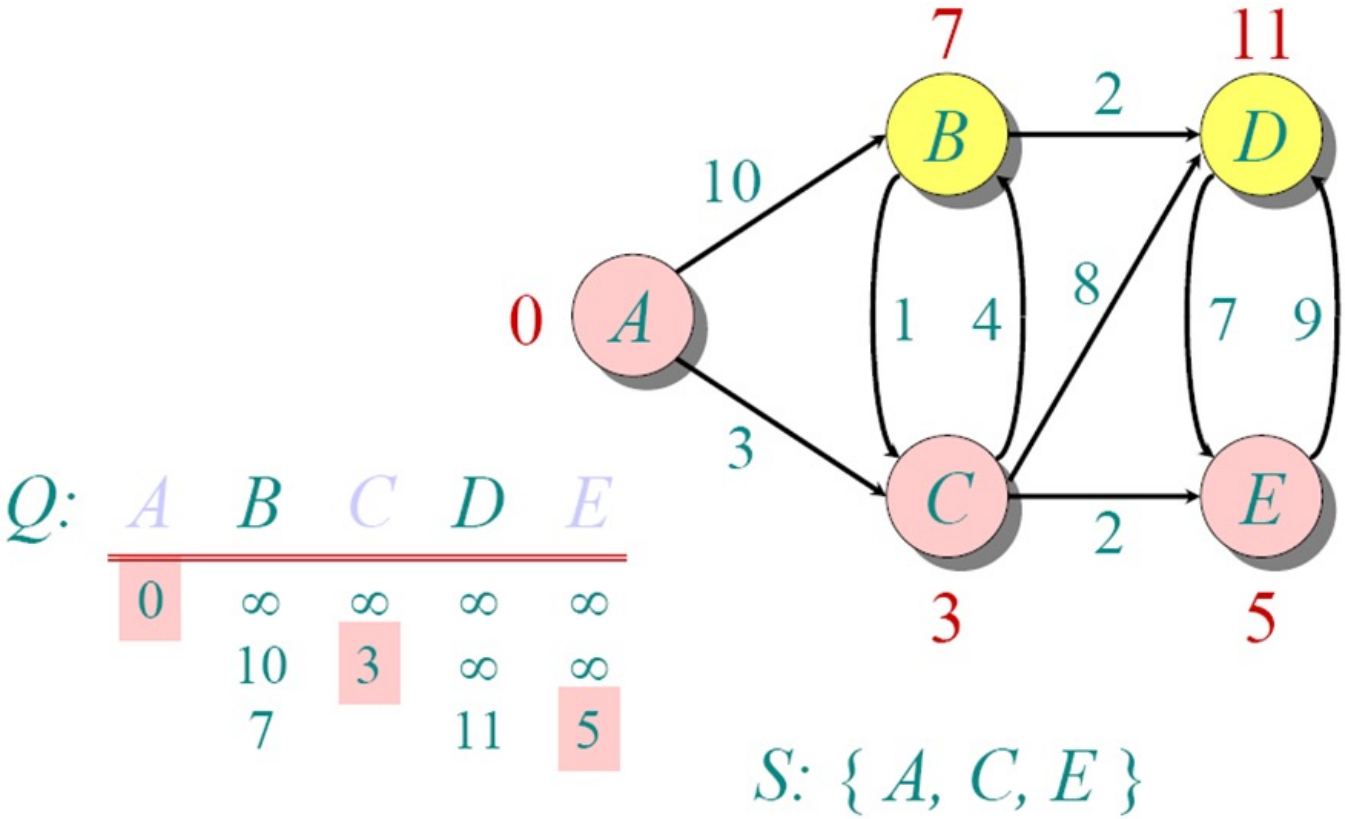
Dijkstra Animated Example



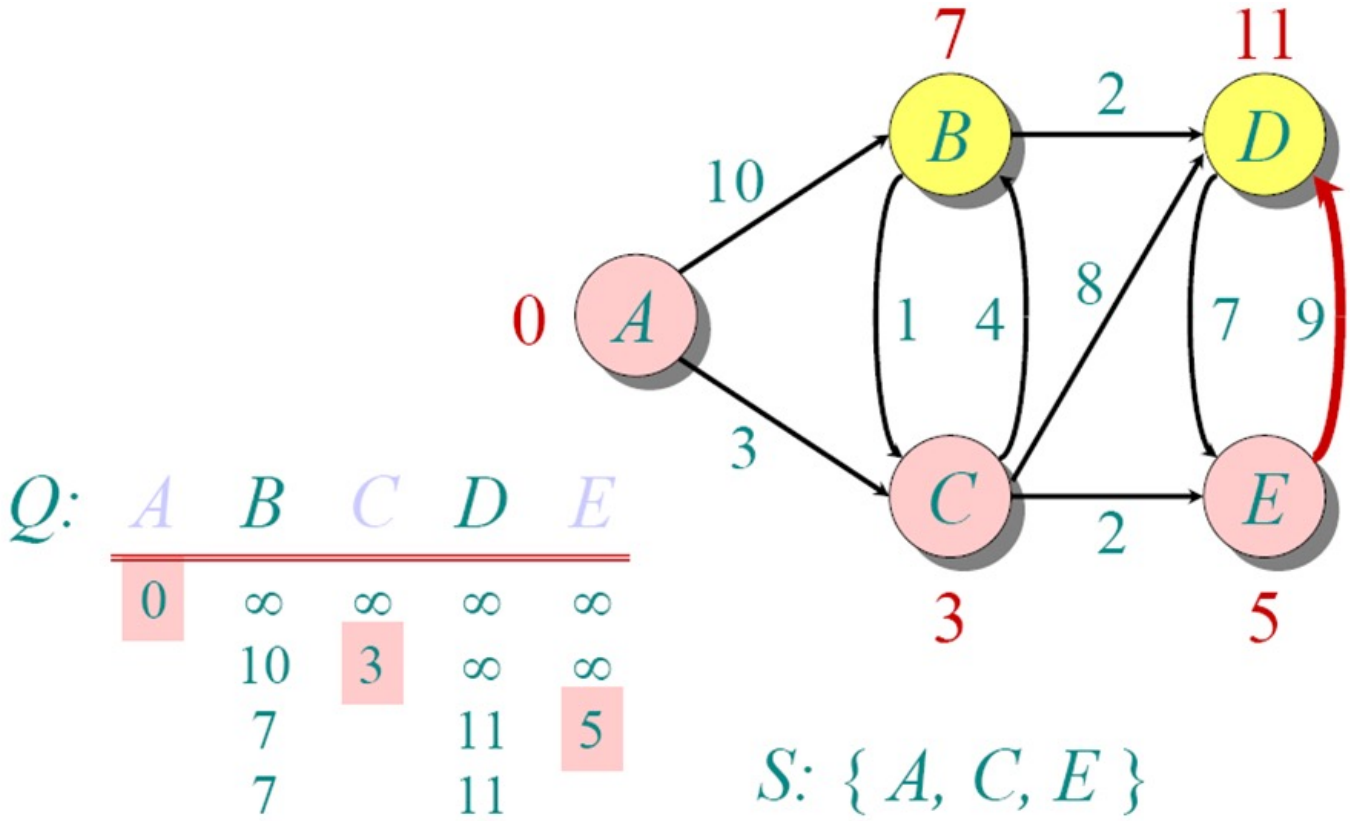
Dijkstra Animated Example



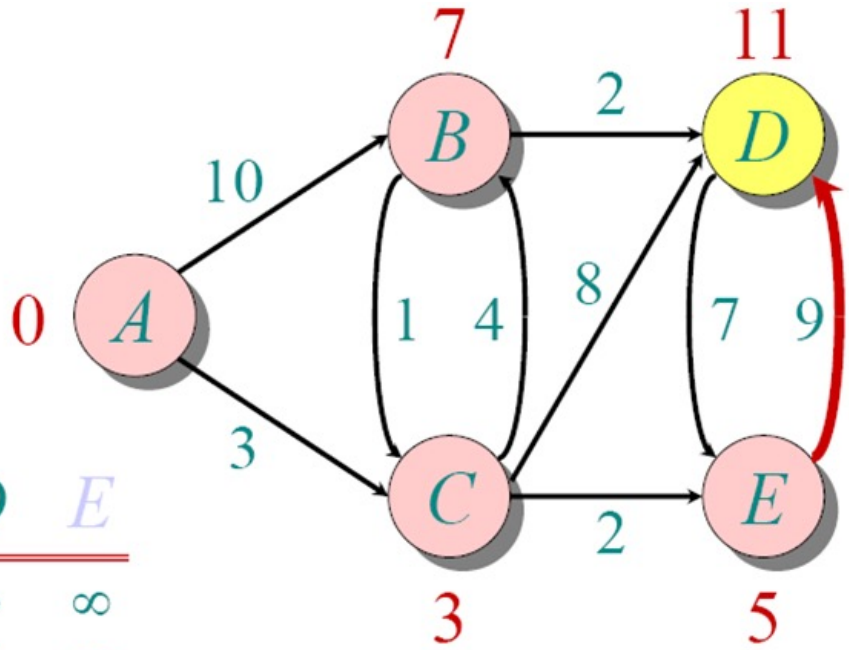
Dijkstra Animated Example



Dijkstra Animated Example



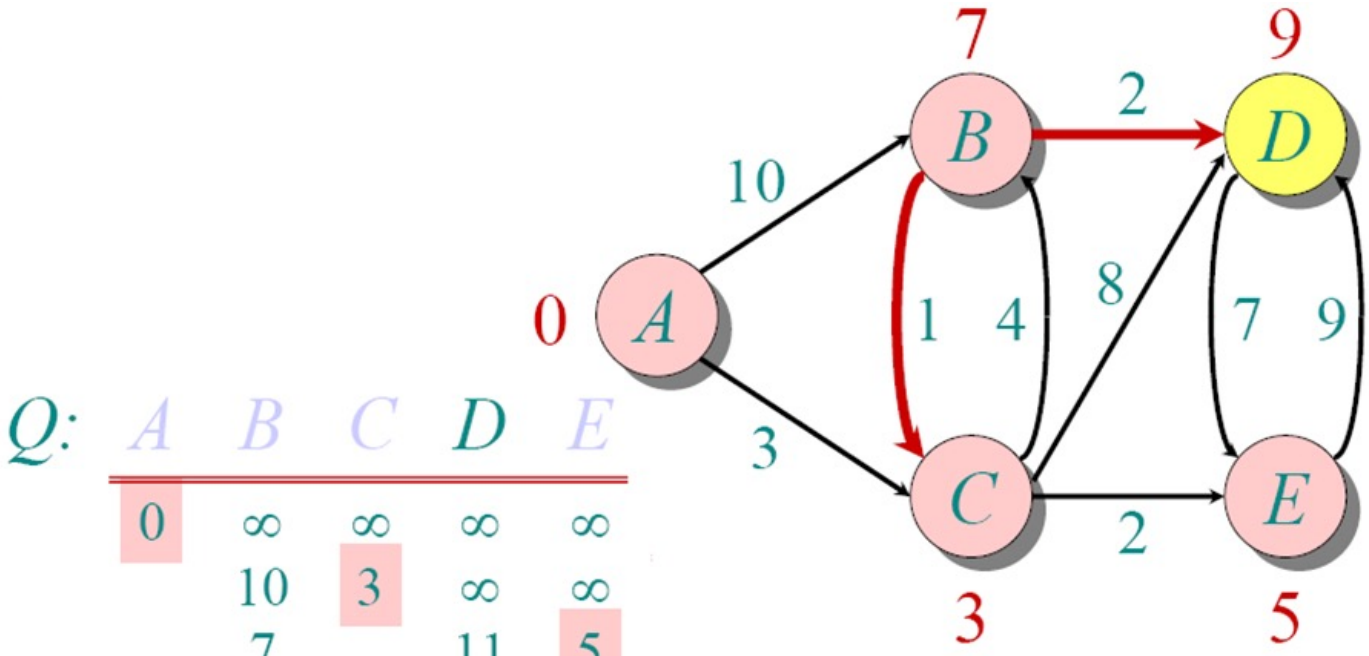
Dijkstra Animated Example



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞
		7		11	5
		7		11	

$S: \{ A, C, E, B \}$

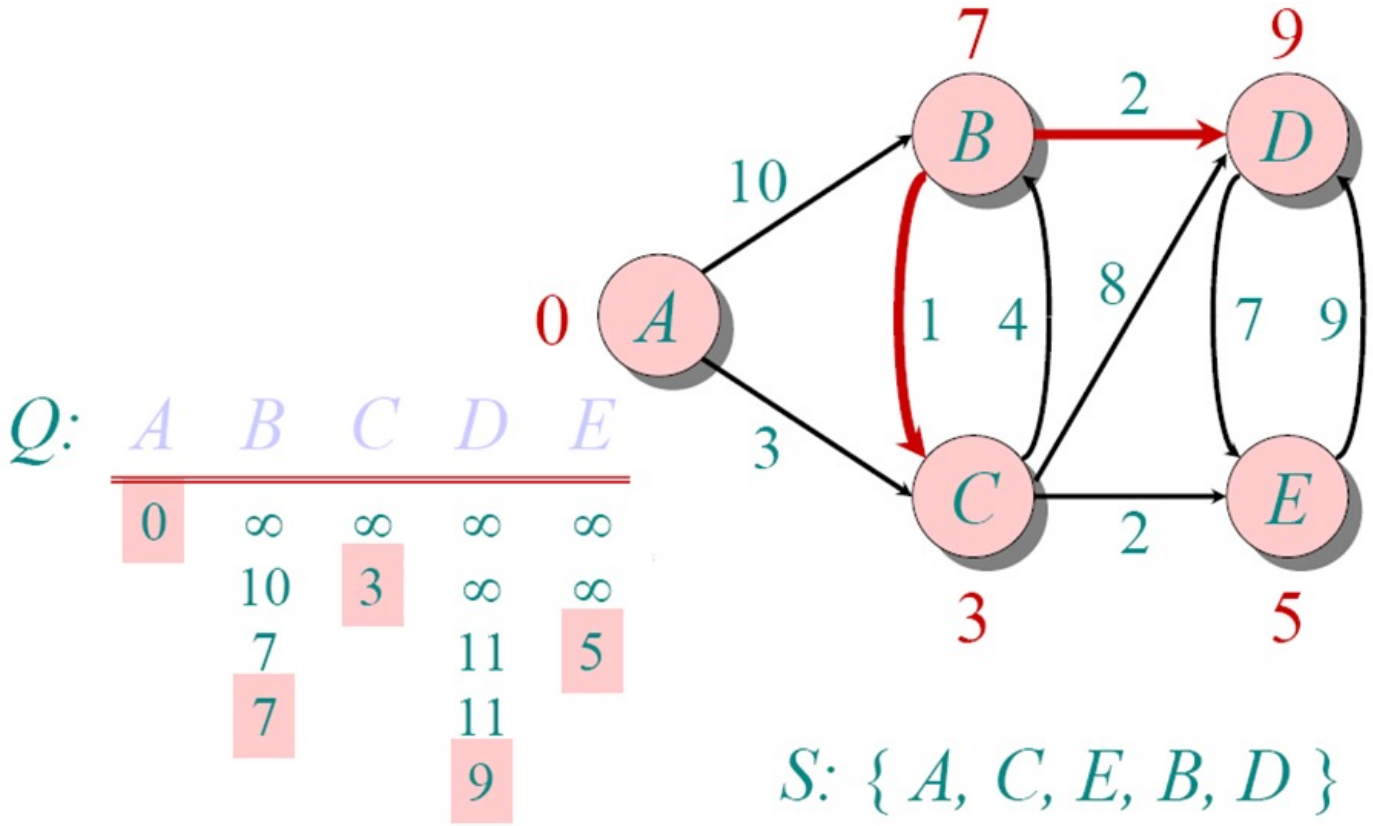
Dijkstra Animated Example



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞
		7		11	5
		7		11	
				9	

S: { A, C, E, B }

Dijkstra Animated Example



Implementations and Running Times

- The simplest implementation is to store vertices in an array or linked list. This will produce a running time of
  - O(|V|^2 + |E|)
- For sparse graphs, or graphs with very few edges and many nodes, it can be implemented more efficiently storing the graph in an adjacency list using a binary heap or priority queue. This will produce a running time of
  - O((|E|+|V|) log |V|)

Inter-domain Routing: BGPv4

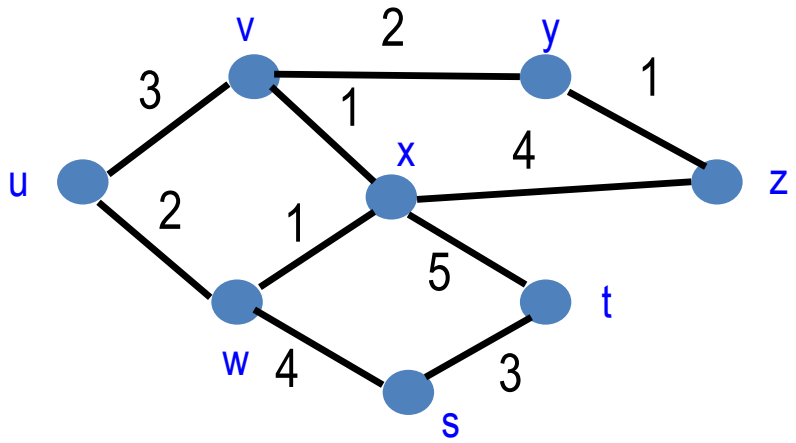
- Routing between domains: Border Gateway Protocols (BGP)
- BGP's perspective: Only autonomous systems and their connections
  - Routing complicated by politics, e.g. only route packets for paying customers, …
  - Legal constraints, e.g. traffic originating and ending in Canada must not leave Canada while in transit

Inter-domain Routing: BGPv4

- Basic operation: Distance vector (DV) protocol
  - Propagate information about reachable networks and distances one hop at a time
    - Each node calculates the distances between itself and all other nodes and stores this information as a table.
    - Each node sends its table to all neighboring nodes.
    - When a node receives distance tables from its neighbors, it calculates the shortest routes to all other nodes and updates its own table to reflect any changes.
    - Each router learns only next step to destination
    - Need TCP connections between routers to exchange the tables → Route reflector?
    - https://www.youtube.com/watch?v=1yt76zl5jcg
  - Optimizations in BGP:
    - Not only keep track of cost via a given neighbor, but store entire paths to destination ASs
      - > Path vector protocol
    - More robust, solves problems like count to infinity, i.e. can handle disconnected networks efficiently

Distributed Control Plane

- Distance-vector routing: RIP → IGP, BGP → Exterior Gateway Protocol (on border and exterior routers, ISPs)
  - Each node computes path cost based on each neighbors' path cost
  - Disseminate Distance Vectors (parts/all of routing table) to neighbors
  - Bellman-Ford algorithm

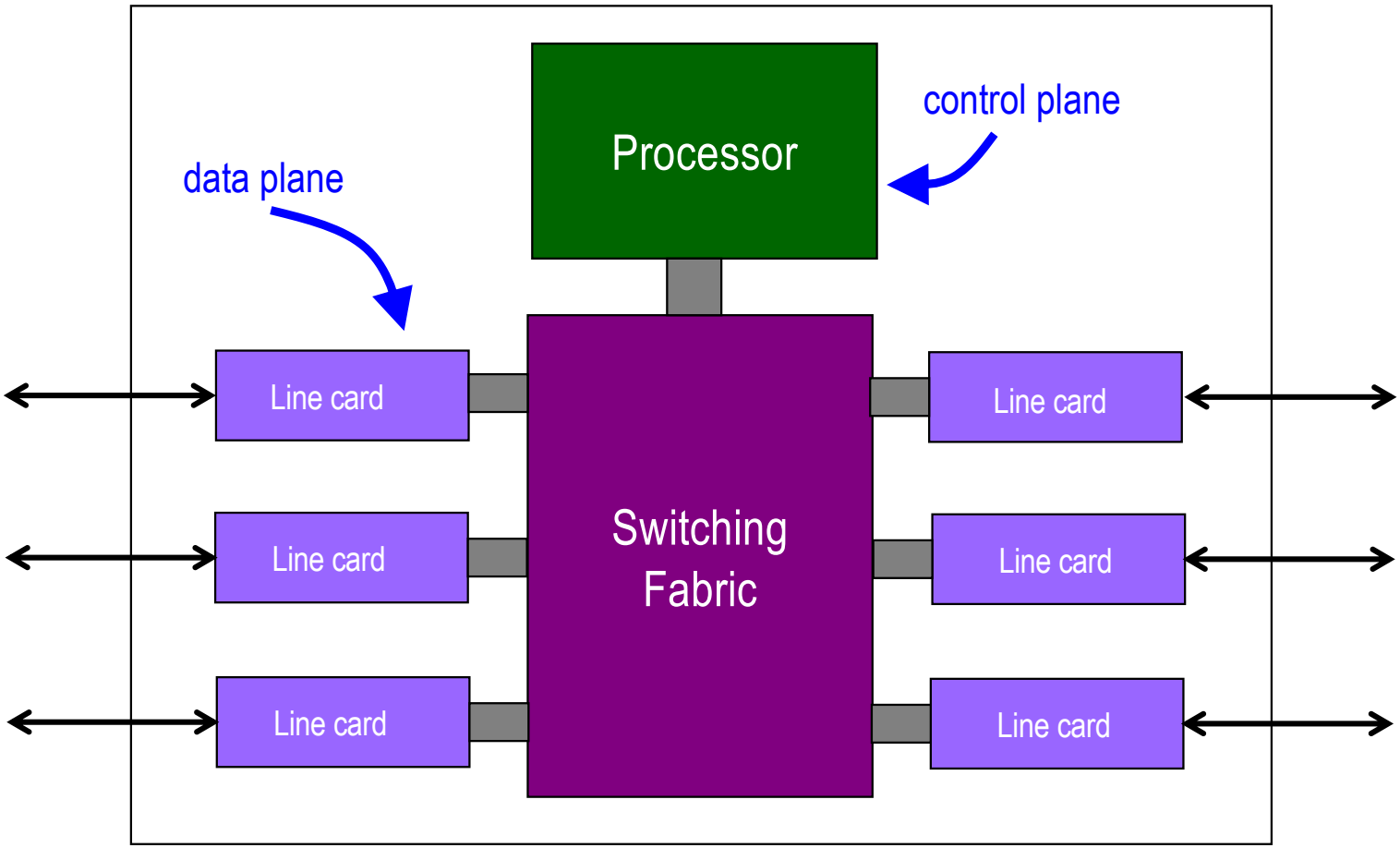


DV of v

$$d_u(z) = \min\{c(u,v) + d_v(z), \; c(u,w) + d_w(z)\}$$

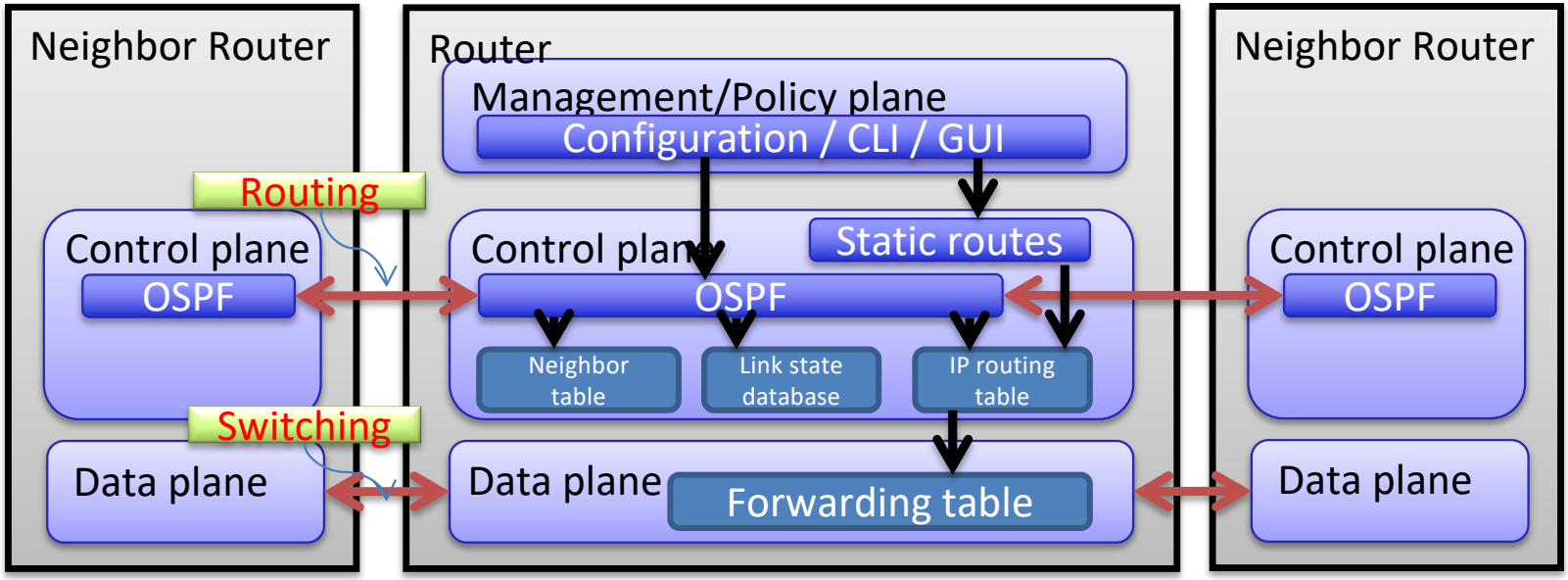
DV of w

Data and Control Planes



Introduction of Different Planes

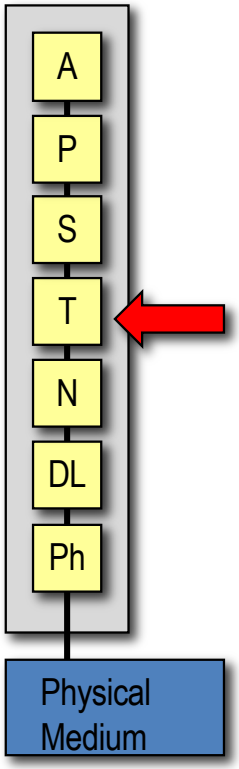
- Separation into Control and Data plane
  - Control plane: Decision where to forward together with neighbors
  - Data Plane: Fast Forwarding based on local state
  - Management Plane: Configuration Interface



The ISO/OSI Model

Transport Layer:

- End-to-end *protocol*
- provides *connection-oriented* and *connection-less* services
- *Flow control* between applications
- Reliable transmission *(Re-ordering, re-transmission)*



The ISO/OSI Model

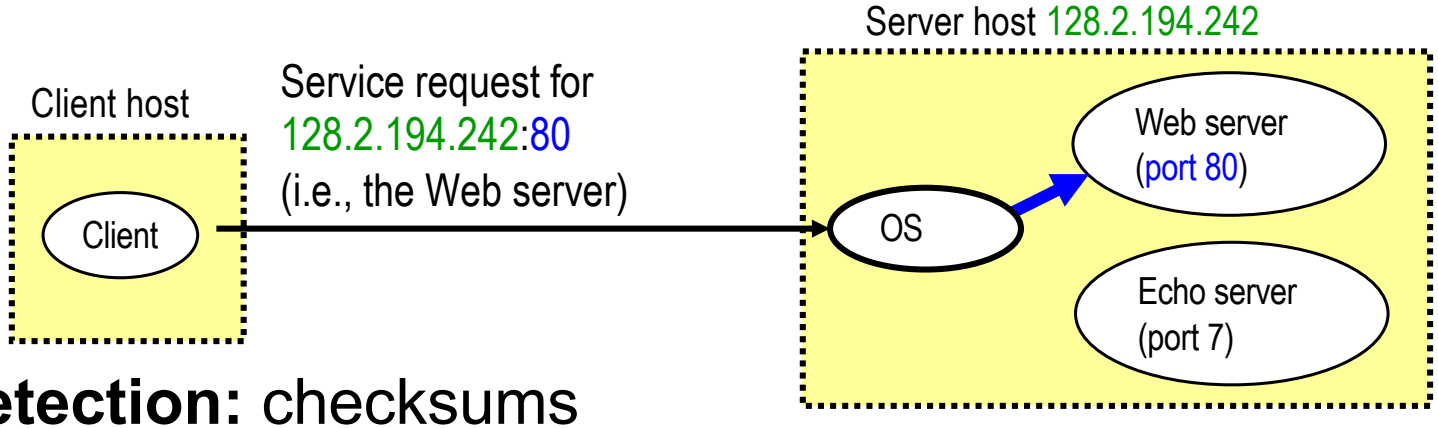
The *Transport Layer* provides application-to-application connectivity

Tasks:

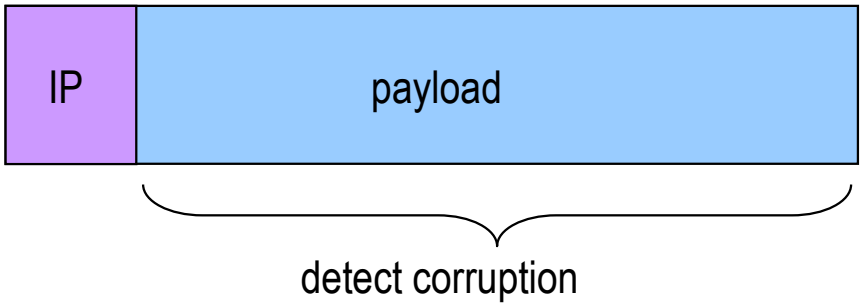
- ***Mapping*** **transport to network addresses**
- ***Selecting*** **appropriate network services**
- ***Set-up and tear-down*** **of connections**
- ***Multiplexing***
- ***Segmentation***
- ***Error detection*** **and** ***Error correction (Reliability)***
- ***Flow/congestion control***
- **Exchanging** ***control*** **data**
- ***Hiding*** **operating system details (of packet sending)**

Transport Layer: Two main issues

- **Demultiplexing:** port numbers

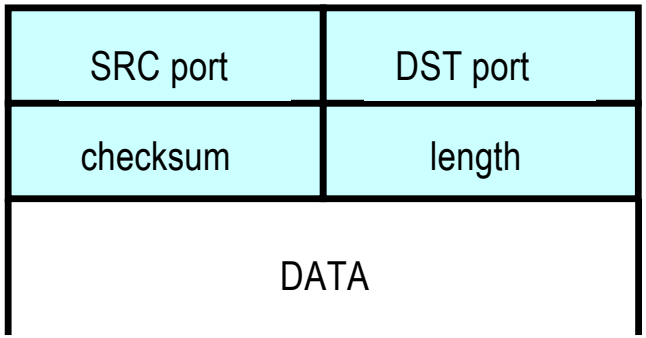


- **Error detection:** checksums



User Datagram Protocol (UDP)

- Datagram messaging service
  - Demultiplexing: port numbers
  - Detecting corruption: checksum
- Lightweight communication between processes
  - Send and receive messages
  - Avoid overhead of ordered, reliable delivery



Transmission Control Protocol (TCP)

- Stream-of-bytes service
  - Sends and receives a stream of bytes
- Reliable, in-order delivery
  - Corruption: checksums
  - Detect loss/reordering: sequence numbers
  - Reliable delivery: acknowledgments and retransmissions
- Connection oriented
  - Explicit set-up and tear-down of TCP connection
- Flow control
  - Prevent overflow of the receiver's buffer space
- Congestion control
  - Adapt to network congestion for the greater good

ApplicationLayer: HyperText Transfer Protocol (HTTP)

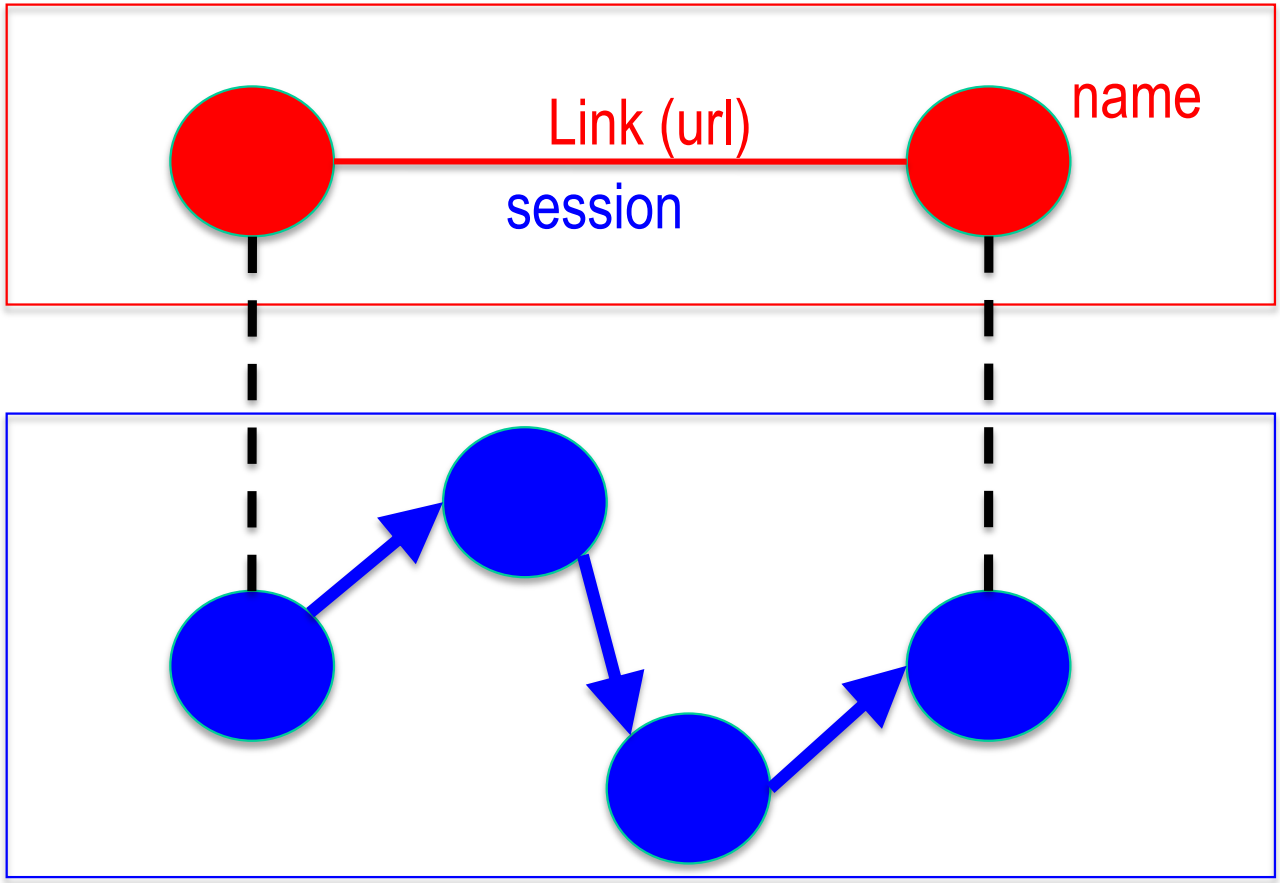
```
GET /courses/archive/HT2016/DVGC02/ HTTP/1.1
Host: www.cs.kau.se
User-Agent: Mozilla/4.03
CRLF
```

Request

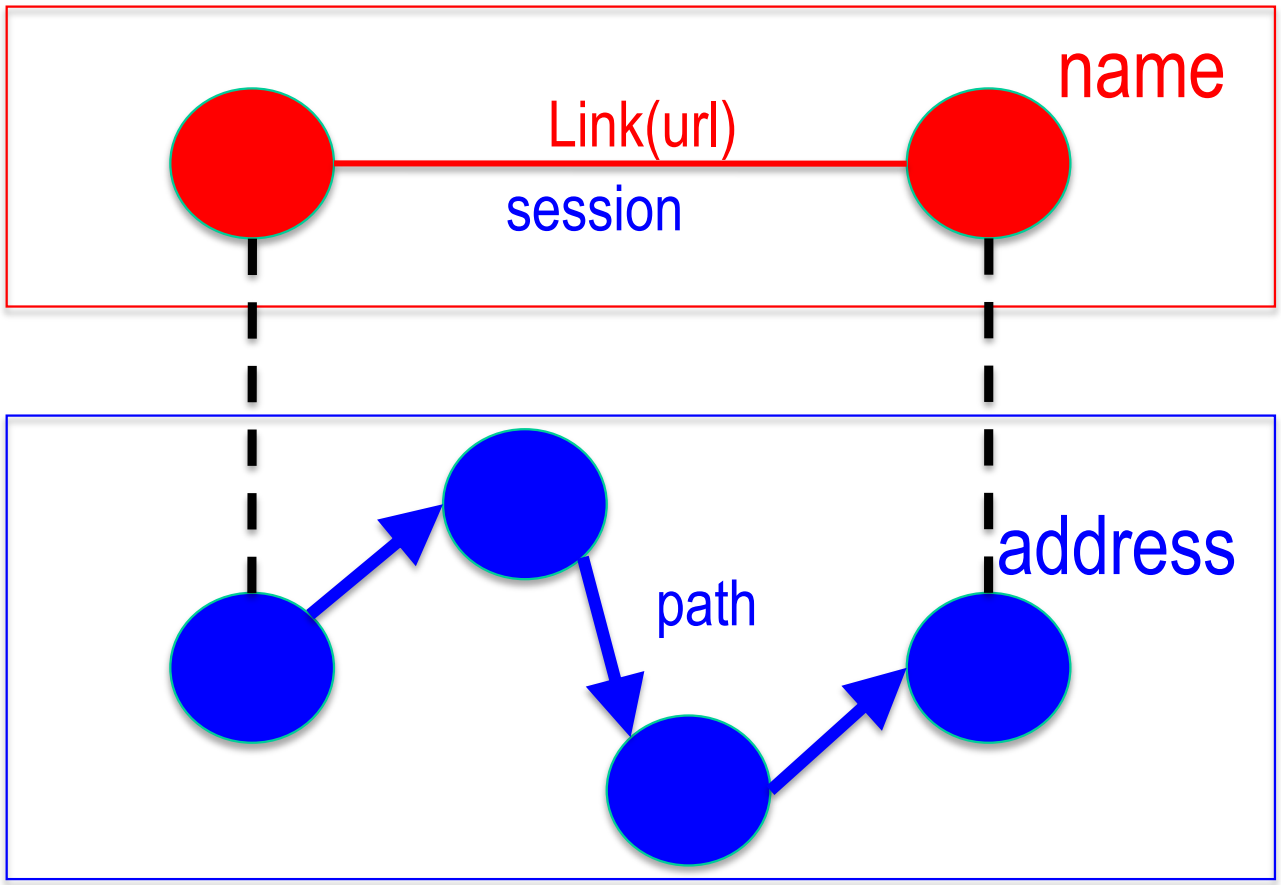
Response

```
HTTP/1.1 200 OK
Date: Mon, 6 Feb 2012 13:09:03 GMT
Server: Netscape-Enterprise/3.5.1
Last-Modified: Mon, 7 Feb  2011 11:12:23 GMT
Content-Length: 21
CRLF
Site under construction
```

Relationship between layers



Discovery: Mapping Name to Address



Routing: Mapping Link to Path