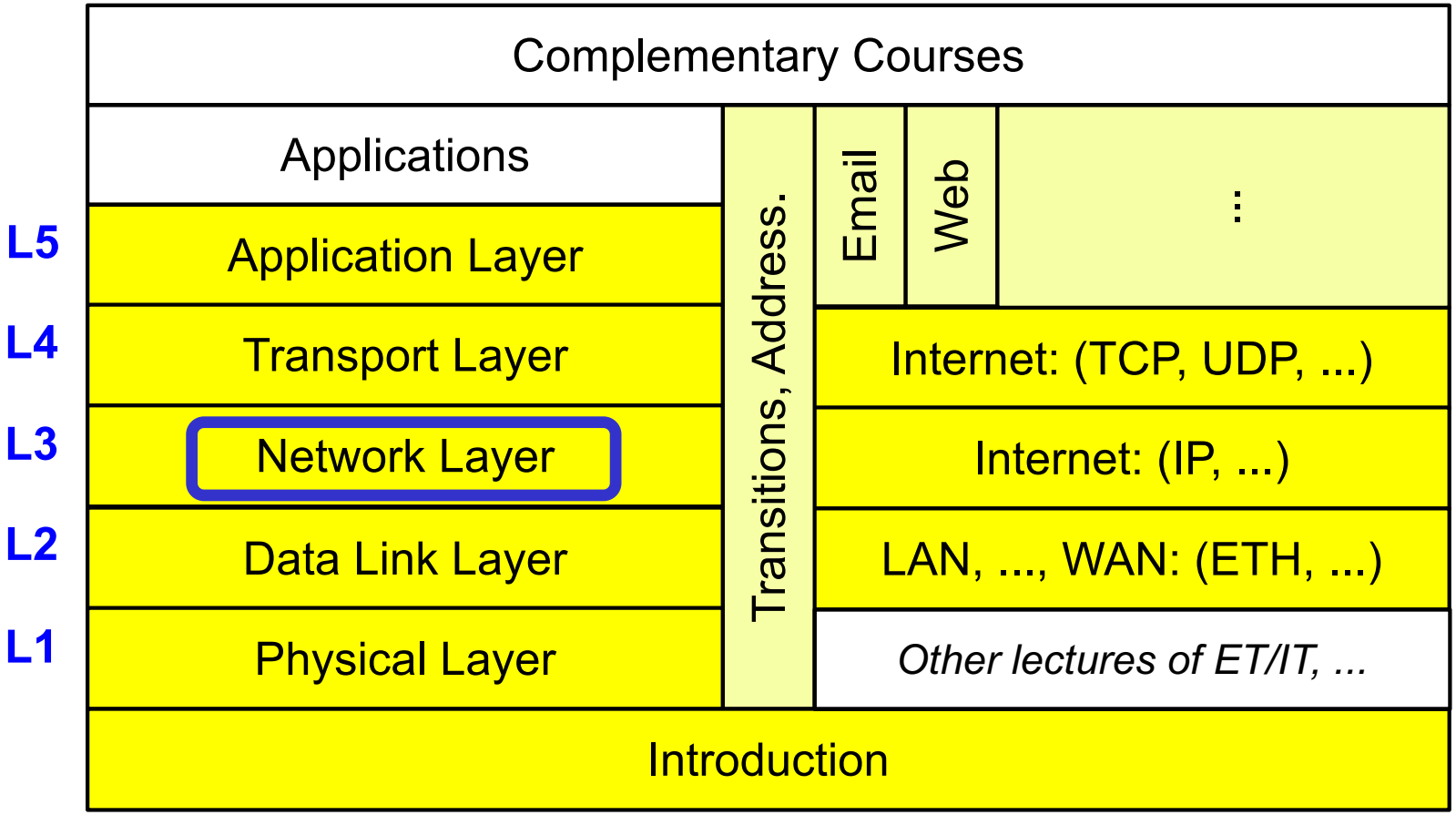
Computer Networks I

Network Layer

Prof. Dr.-Ing. **Lars Wolf**

IBR, TU Braunschweig
Mühlenpfordtstr. 23, D-38106 Braunschweig, Germany,
Email: wolf@ibr.cs.tu-bs.de

Scope


Complementary Courses
Applications
Application Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer
L5
L4
L3
L2
L1
Transitions, Address.
Email
Web
...
Internet: (TCP, UDP, ...)
Internet: (IP, ...)
LAN, ..., WAN: (ETH, ...)
Other lectures of ET/IT, ...
Introduction

Overview

1. Functions (in) the Network Layer

2. Switching Approaches
   2.1 Circuit Switching
   2.2 Message Switching
   2.3 Packet Switching
   2.4 Virtual Circuit Switching

3. Services: Concepts
   3.1 Service: Connection Oriented Communication
   3.2 Service: Connectionless Communication
   3.3 Services: Comparison of Concepts

Overview

4. Routing: Foundations

5. UNICAST (Point-to-Point) Routing: NON-ADAPTIVE
   5.1 Non-Adaptive Shortest Path Routing
   5.2 Non-Adaptive Flooding
   5.3 Non-Adaptive Flow-Based Routing (left out due to time constraints)

6. UNICAST (Point-to-Point) Routing: ADAPTIVE
   6.1 Adaptive Centralized Routing
   6.2 Adaptive Isolated Routing – Backward Learning
   6.3 Adaptive Distributed – Distance-Vector Routing
   6.4 Adaptive Distributed – Link State Routing

7. Addressing

Overview

In Computer Networks 2:

Unicast Routing: Diverse Enhancements (Multipath, Hierarchical Routing)

Broadcast Routing

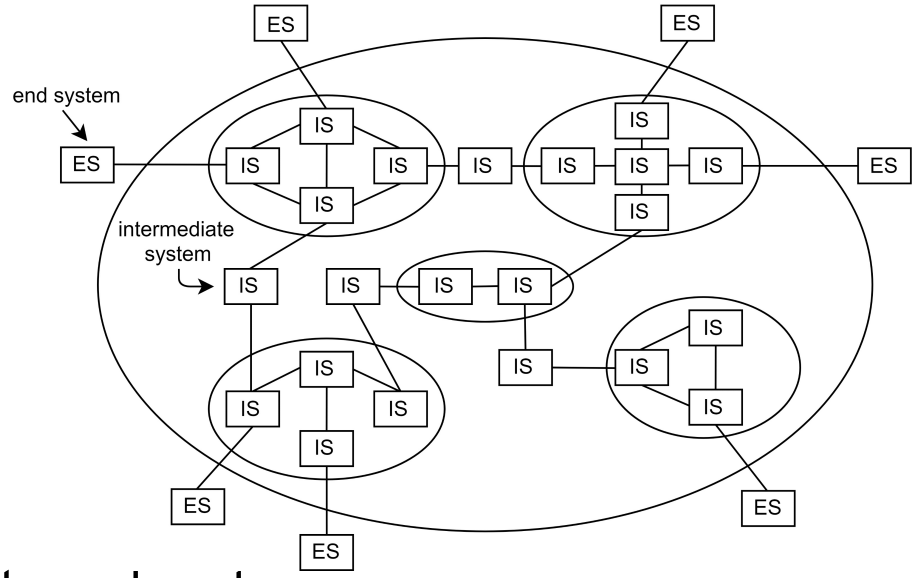
Multicast Routing

Congestion Control

1 Functions of the Network Layer

L2 allows for:

- communication between neighbors
- setup of networks with links between switches



BUT:

- potentially uses broadcast
- leads to large tables in switches
- crossing L2 technologies is difficult

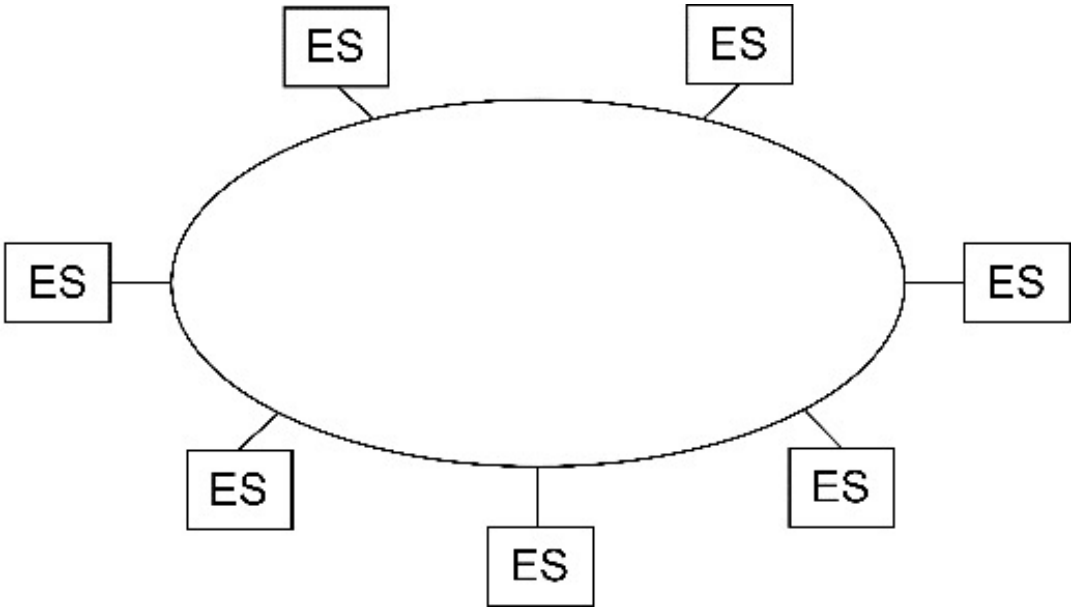
Goals: Data transfer from end system to end system

- several hops,
- (heterogeneous) subnetworks
- compensate for differences (end systems, link layer) during transmission

Relevance of the interface: switching vs. transport service

- L1 up to L1,L2+L3: organization: carrier
- from L4 onward: user/customer/company

Functions of the Network Layer



The provided services are

- standardized for end systems
- independent from network technology
- independent from number, type and topology of the subnetworks

SUBNETWORKS (IS 7498):

A multiple of one or several intermediary systems that

- provide switching functionalities
- through which open end systems can establish network connections

Functions of the Network Layer

Primary tasks

- data transmissions based on
  - datagram
  - or virtual circuits
- routing
- internetworking
  - to provide transitions between networks
- addressing
- fragmentation and reassembling

Functions of the Network Layer

Secondary tasks,
often for specific types of service only:

- congestion control
- Quality of Service (QoS)
  - example: bandwidth, delay, error rate
  - negotiate costs vs. quality of service to be provided

- multiplexing of network connections
- error detection and correction
- flow control
- maintaining the transmission sequence

Functions of the Network Layer

Required knowledge

- subnetwork topology
- address / localization of the end system
- network status (utilization,...)
- packet / data stream communication requirements (Quality of Service)

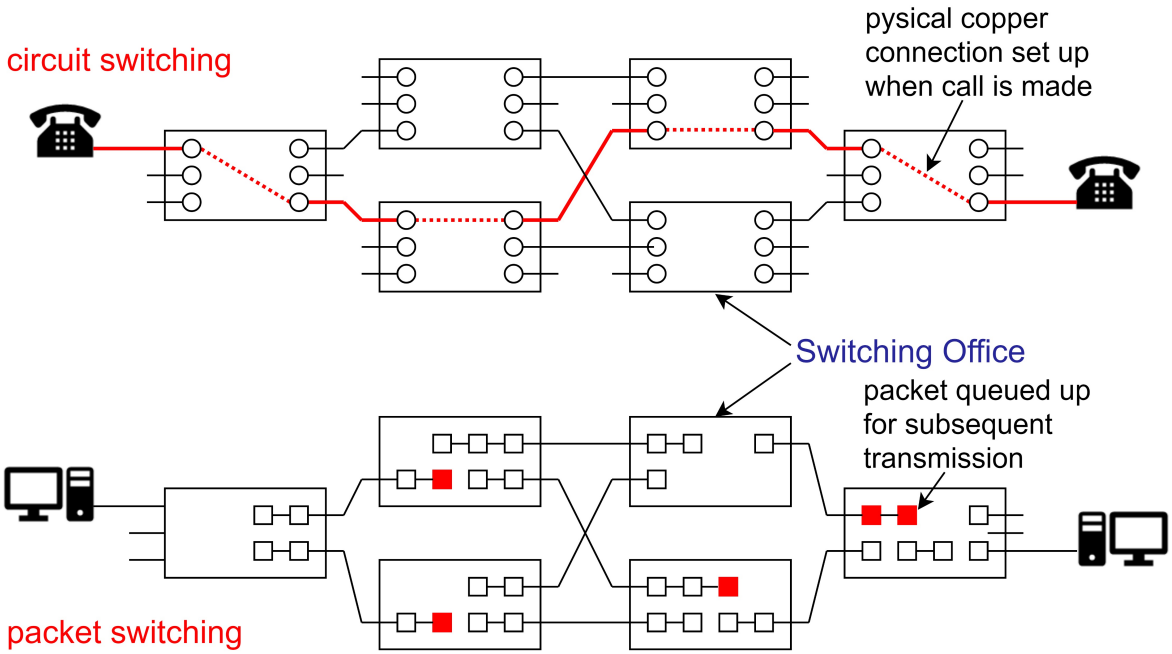
Examples

- Internet protocol IP (TCP/IP,..)

Nomenclature:

Layer	Data Entity
Transport	…
Network	Packet
Data Link	Frame
Physical	Bit/Byte (bit stream)

2 Switching Approaches



Circuit switching
- switching a physical connection

Message switching
- message is stored and passed on by one hop

Packet switching
- store-and-forward, but transmission packets are limited in size

Switching by virtual circuit
- packets (or cells) over a pre-defined path

2.1 Circuit Switching

Principle

- **dedicated path** from src to destination for entire duration of *call*
  - connections between switching centers (frequency spectrum, dedicated ports)

Implementation examples

- historically: on switching boards
  - mechanical positioning of the dialers
  - setting coupling points in circuits

Properties

- connection has to be setup before transmission
  - establishing a connection takes time
- fixed allocation of bandwidth → no congestion during transfer
- no processing of data at intermediate nodes
  - constant and short delay
- information delivery is sequential (by nature)
- resource allocation too rigid (possibly waste of resources)
  - no support for transmission of bursty data
    → potential resource underutilization

2.2 Message Switching

Principle

- all data to be sent is treated as a "message"
- "store and forward" network:
  in each node the message is handled as follows:
  1. receive message
  2. check message and handle potential errors
  3. store message, and
  4. forward message (as a whole to the next node)

Example

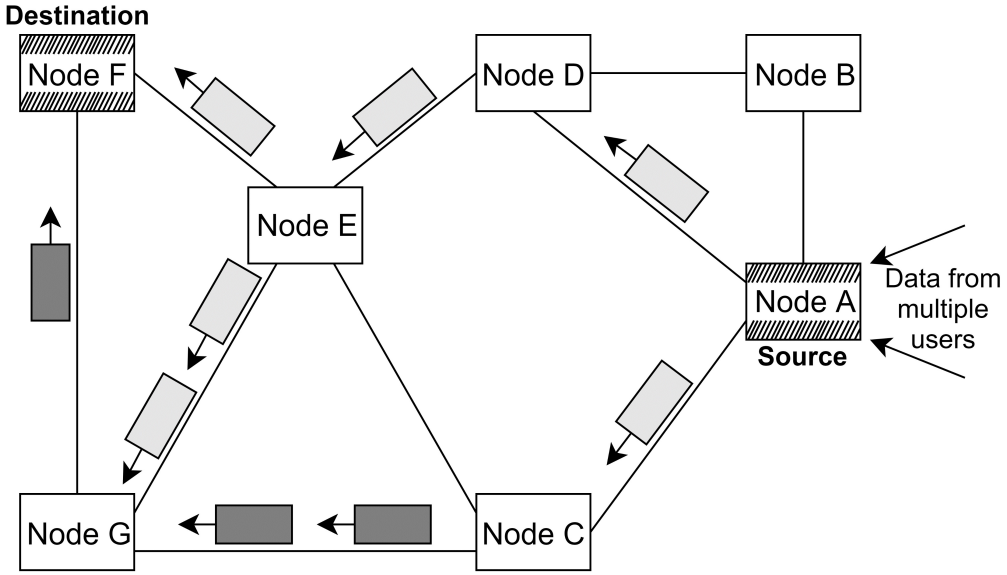
- early telegram service

Properties

- high memory requirements at each node (switching centers),
  - because message may be of any size
  - usually stored on secondary repository (hard disk)
- node may be used to its full capacity over a longer period of time by one message,
  - i. e. better if packets are of limited size (packet switching)

2.3 Packet Switching

Example:
Internet



Principle

- packets of limited size
- dynamic determination of route for every packet
- no dedicated path from source to destination

Properties

- no connect phase
- dynamic allocation of bandwidth
  - suitable for bursty traffic
  - flexible, provides for resource sharing and good utilization
- congestion possible
- bandwidth reservation difficult, QoS provisioning limited
- variable end-to-end delay
  - due to queuing at intermediate nodes (and varying routes)
- information delivery may neither be sequential nor reliable

2.4 Virtual Circuit Switching

Principle:

- setup path from source to destination for entire duration of call
- using state information in nodes, but no physical connection
- connection setup → define the data path
- data packets (as in packet switching), but all follow ONE path
  - packets may carry only the address of the network entry point
    - not the complete destination address, e.g., ATM: VPI/VCI

Examples:

- ATM (Asynchronous Transfer Mode)
  - PVC (permanent virtual circuit): established "manually“
  - SVC (switched virtual circuit): established using signaling protocols
- Internet Integrated Services
  - state established via signaling protocol (RSVP)
  - full addresses are used

Properties

- all messages of a connection are routed over the same pre-defined data path, i.e., sequence is maintained
- it is easier to ensure Quality of Service

Comparison: Circuit and Packet Switching

Circuit switching:

- connection establishment can take a long time
- bandwidth is reserved
  - no danger of congestion
  - possibly poor bandwidth utilization (bursty traffic)
- continuous transmission time,
  because all data is transmitted over the same path
- price calculation classically based on duration of connection

Packet switching:

- connect phase not (absolutely) necessary
- dynamic allocation of bandwidth
  - danger of congestion
  - optimized bandwidth utilization
- varying transmission time
  - because packets of a connection may use different paths
  - not suitable for isochronous data streams
- price calculation (classically) based on transfer volume

3 Services

Concepts

- Connection-oriented vs. connectionless communication

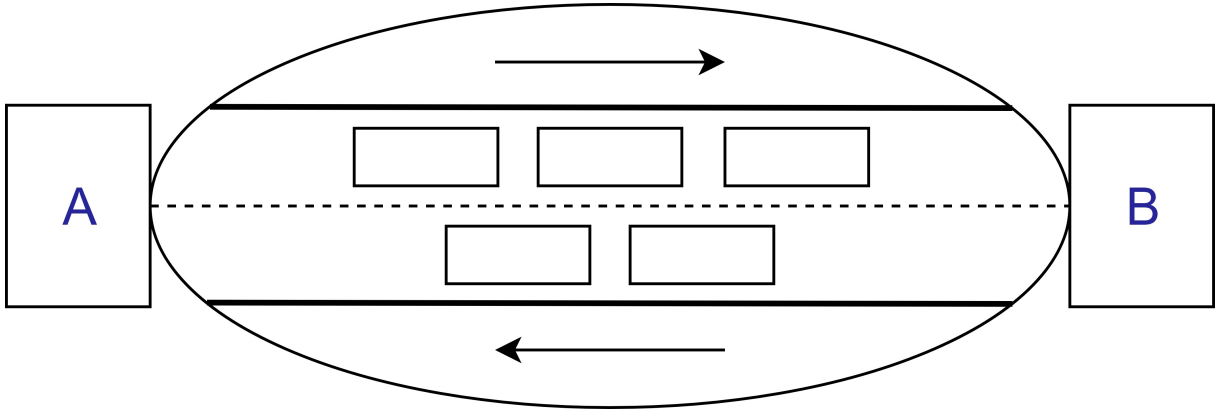
Connection-oriented

- goal: error free communication channel
- error control in L3
  - flow control, ...
- usually duplex communication
- more favorable for real-time communications
- typical approach of telephone and telecomm. companies:
  - X.25, ATM, various mobile systems

Connectionless

- unreliable communication
- hardly any error control in L3: left to L4 or higher layers
  - sequence not ensured, ...
- simplex communication
- more favorable for simple data communication:
  - SEND-PACKET, RECEIVE-PACKET
- Internet community: IP

3.1 Service: Connection-Oriented Communication



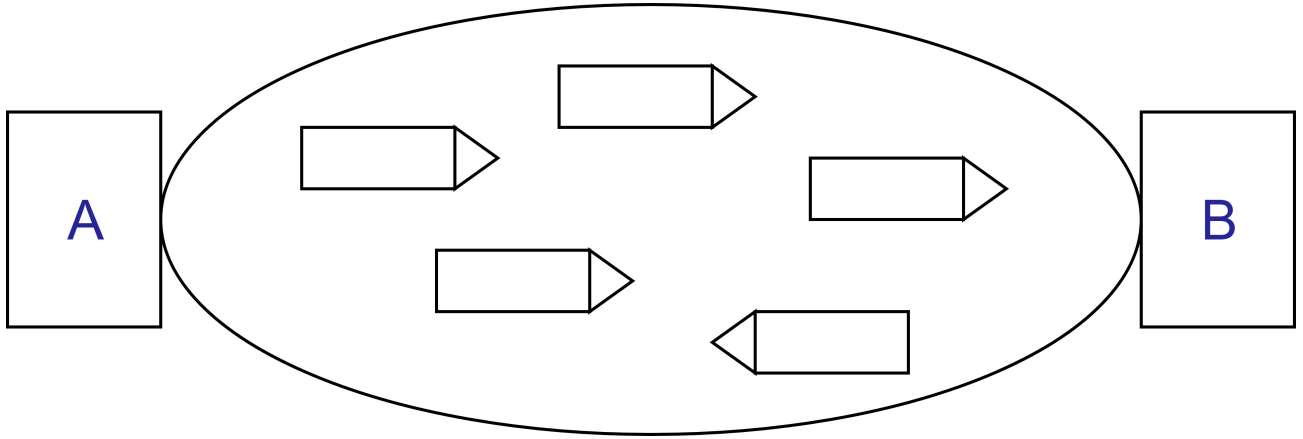
Properties:

- 3-phase interaction
  - connect
  - data transfer
  - disconnect
- (allows for) QUALITY OF SERVICE NEGOTIATION
  - e.g., throughput, error probability, delay
- (typically) RELIABLE COMMUNICATION in both directions
  - no loss, no duplicates, no modification
  - ensures maintenance of the correct sequence of transmitted data
- FLOW CONTROL
- relatively complex protocols

Classical example:

- telephone service

3.2 Service: Connectionless Communication



Properties:

- network transmits packets as ISOLATED UNITS (datagram)
- UNRELIABLE COMMUNICATION:
  - loss, duplication, modification, sequence errors possible
- no flow control
- comparatively SIMPLE PROTOCOLS

Classical example:

- mail delivery service

3.3 Services: Comparison of Concepts

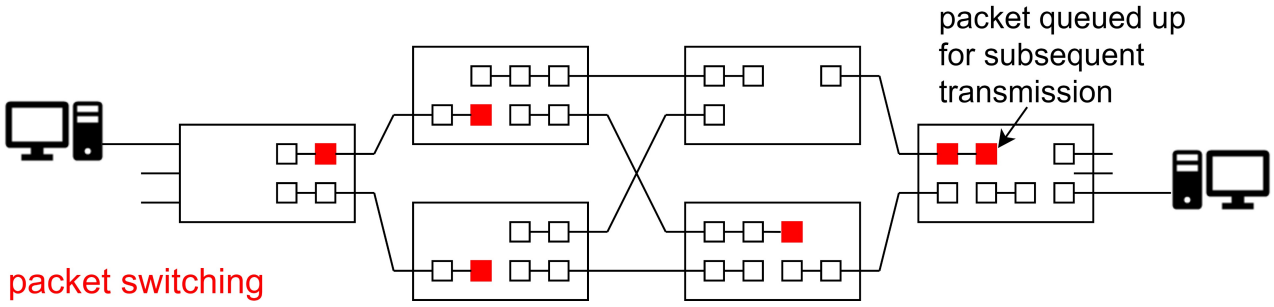
Arguments pro connection-oriented service:

- simple, powerful paradigm
- allows for simplification of the upper layers (L4 - L7)
- simplifies task of end systems
- for some applications efficiency in time is more important than error-free transmission
  - e. g. realtime applications, digital voice transmission
- suitable for a wide range of applications

Arguments pro connectionless service:

- high flexibility and low complexity
- avoids high costs for connects and disconnects for transaction-oriented applications
- easier to optimize the network load
- compatibility and costs: IP as common protocol
- "END-TO-END ARGUMENTS" (Saltzer et al.):
  - reliable communication requires error control within the application
  - and: error control in one layer can replace the error control in the layer underneath it

4 Routing: Foundations



Task:

- define the route of packets through the network
  - from source
  - to destination

ROUTING ALGORITHM:

- define on which **outgoing** line
  an **incoming** packet will be transmitted on

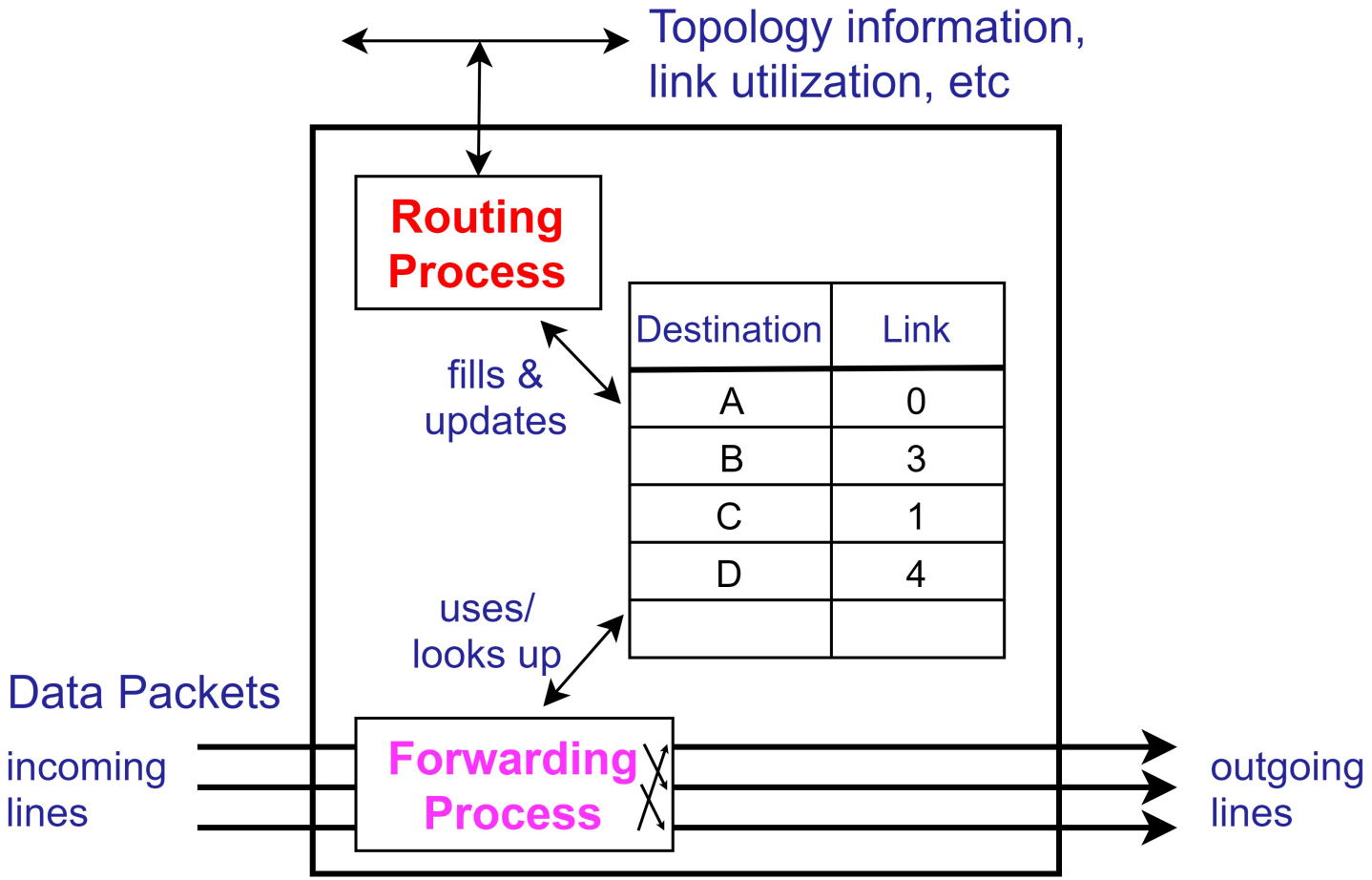
Route determination:

- datagram
  - individual decision for each packet
- virtual circuit
  - one decision for all packets of the same flow
  - routing only during connect (session routing)

Routing & Forwarding

Distinction can be made

- Routing: to take a decision which route to use
- Forwarding: to define what happens when a packet arrives



Desirable Properties of a Routing Algorithm

correctness

simplicity

robustness
- compensation for IS and link failures
- handling of topology and traffic changes

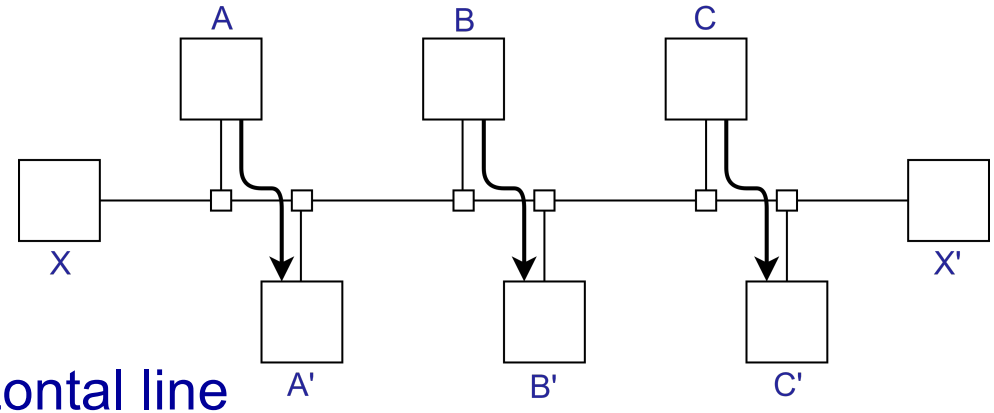
stability
- consistent results
- no volatile adaptations to new conditions

fairness
- among different sources compared to each other

optimality

Routing Algorithms: Conflicting Properties

Often conflicting:
fairness and optimization



Example:

- Communication among
  A → A', B → B', C → C'
  uses full capacity of horizontal line
- optimized throughput, but
- no fairness for X and X'
  - ➔ tradeoff between fairness and optimization

some different optimization criteria

- average packet delay
- total throughput
- individual delay
  - ➔ conflict

therefore often

- hop minimization per packet
  - it tends to reduce delays and decreases required bandwidth
  - also tends to increase throughput

Classes of Routing Algorithms

NON-ADAPTIVE ALGORITHMS

- current network state **not taken** into consideration
  - assume average values
  - all routes are defined off-line before the network is put into operation
  - no change during operation (static routing)
- WITH knowledge of the overall topology
  - spanning tree
  - flow-based routing
- WITHOUT knowledge of the overall topology
  - flooding

ADAPTIVE ALGORITHMS

- decisions are based on current network state
  - measurements / estimates of the topology and the traffic volume
- further sub-classification into
  - centralized algorithms
  - isolated algorithms
  - distributed algorithms

Enhancements (adaptive and non-adaptive algorithms)

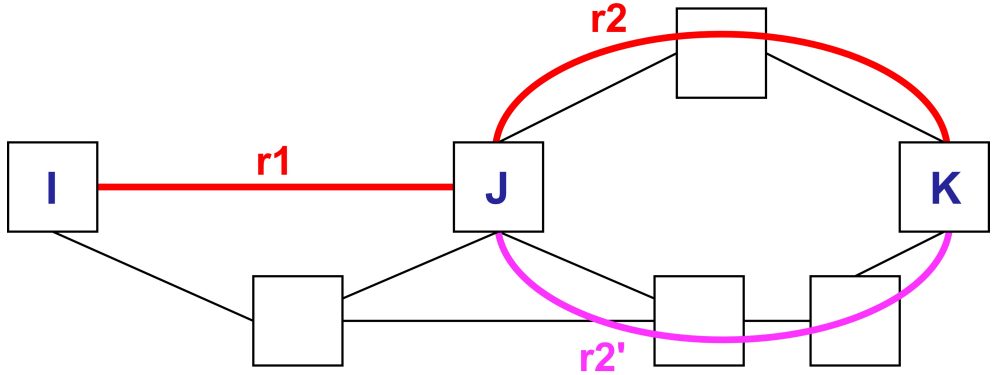
- multiple routing and hierarchical routing definition

Optimality Principle and Sink Tree

General statement about optimal routes:

If
- router **J** is on optimal path from router **I** to router **K**

Then
- the optimal path from router **J** to router **K** uses the same route

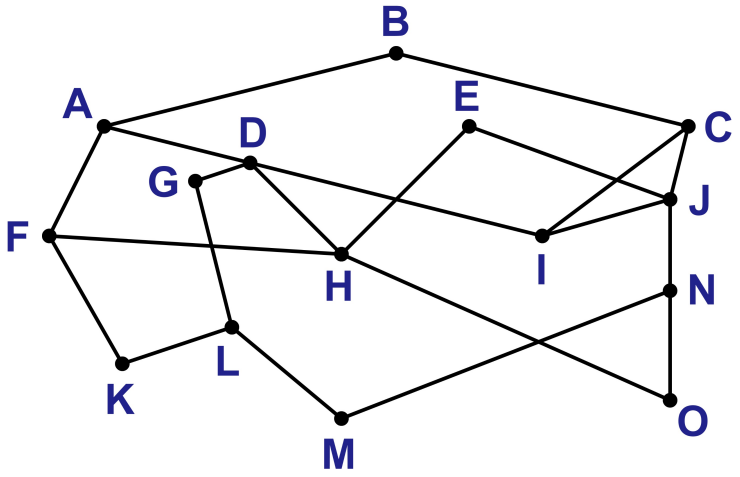


Example:
- r1: route from I to J
- r2: route from J to K
- if better route r2' from J to K would exist
  - then concatenation of r1 and r2' would improve route from I to K (contradiction)

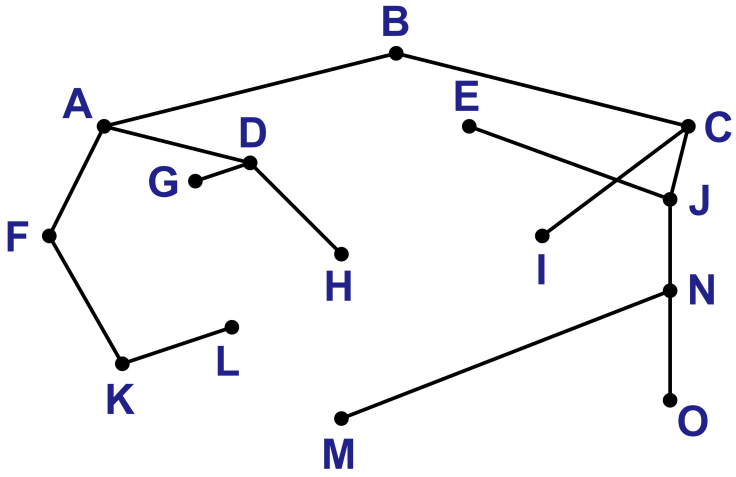
➔ set of optimal routes
- from all sources to a given destination

form a tree rooted at the destination: SINK TREE

Sink Tree: Example



Subnet



Sink Tree for Destination Node B

Comments:

- tree: no loops
  - each packet reaches its destination within finite and bounded number of hops
- not necessarily unique
  - other trees with same path lengths may exist

Goal of all routing algorithms

- discover and use the sink trees for all routers

Further comments:

- information about network topology necessary for sink tree computation
  - yet, sink tree provides benchmark for comparison of routing algorithms

Methodology & Metrics

Networks represented as graphs:
- node represents a router
- arc represents a communication line (link)

Compute the SHORTEST PATH between a given pair of routers

Different metrics for path “lengths” can be used
- can lead to different results
- sometimes even combined
  - (but this leads to computational problems)

Metrics for the "ideal" route, e.g., a "short" route
- number of hops
- geographical distance
- bandwidth
- average data volume
- cost of communication
- delay in queues
- ...

5 Non-Adaptive Routing

Static Procedure

- network operator generates tables
- tables
  - are loaded when IS operation is initiated and
  - will not be changed any more

Characteristics

+ simple
+ good results **if** relatively consistent topology and traffic

- but:
  - poor performance if traffic volume or topologies change over time

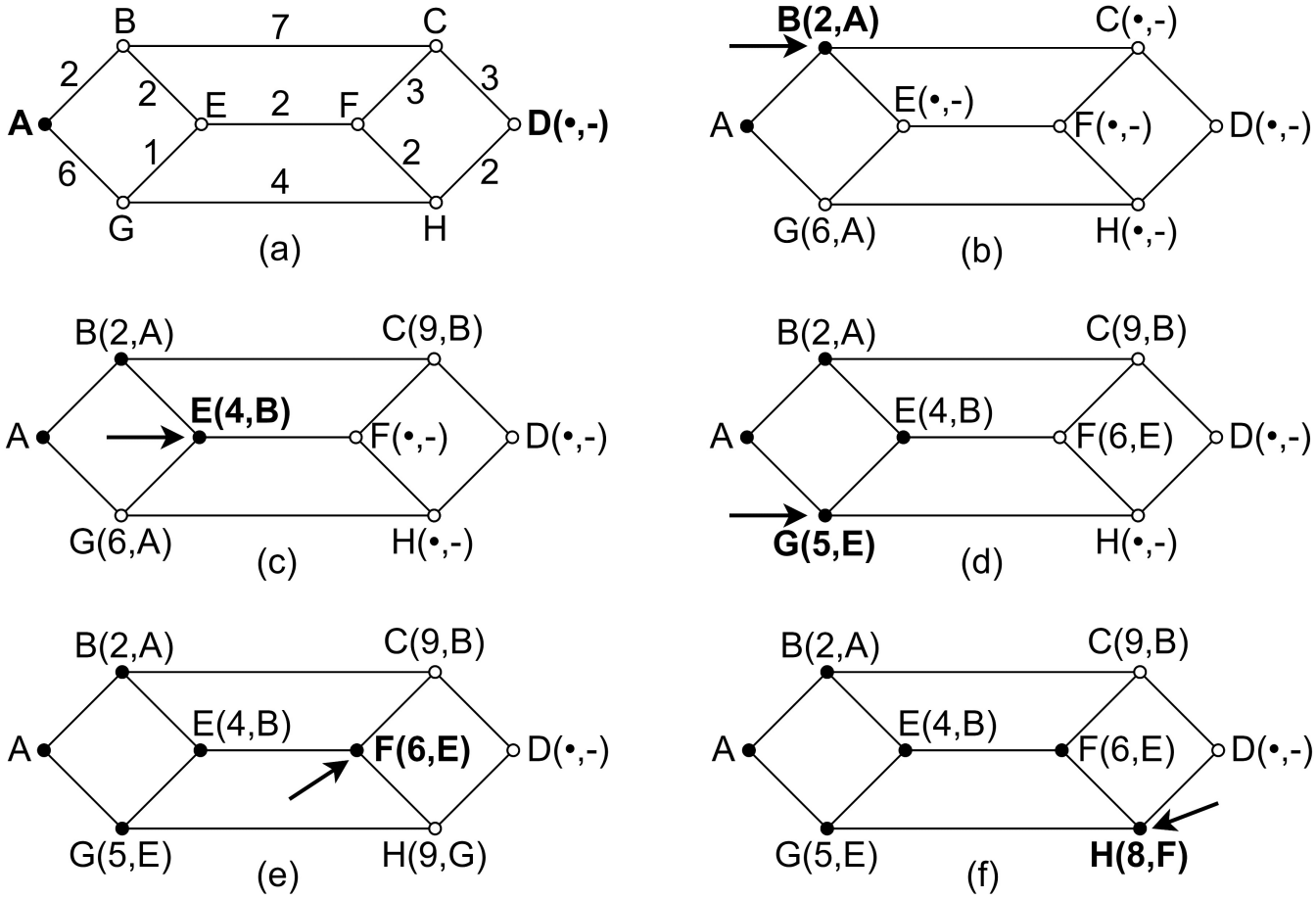
5.1 Non-Adaptive Shortest Path Routing

Spanning Tree and Optimized Route

- information about the entire network has to be available
  - i. e. can be used for comparison purposes / as a benchmark

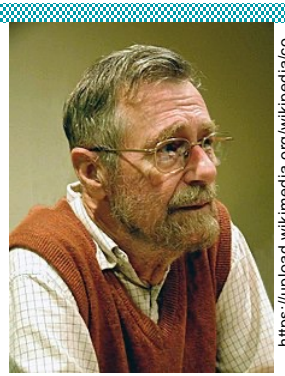
Example:

- link is labeled with distance / weight
- node is labeled with distance from source node along best known path (in parentheses)



Non-Adaptive Shortest Path Routing

Procedure: e. g. according to Edsger W. Dijkstra (1959)



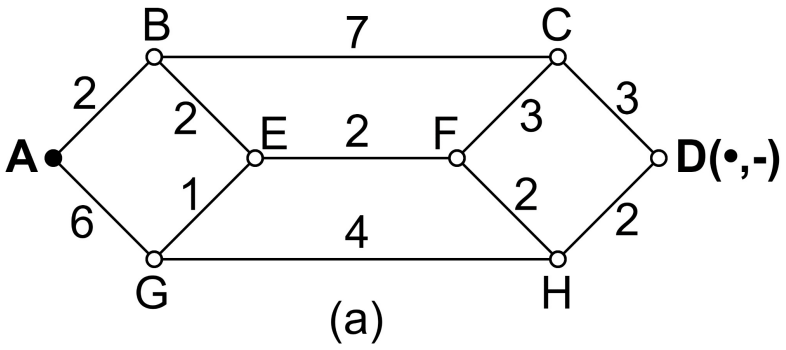
https://upload.wikimedia.org/wikipedia/commons/d/d9/Edsger_Wybe_Dijkstra.jpg

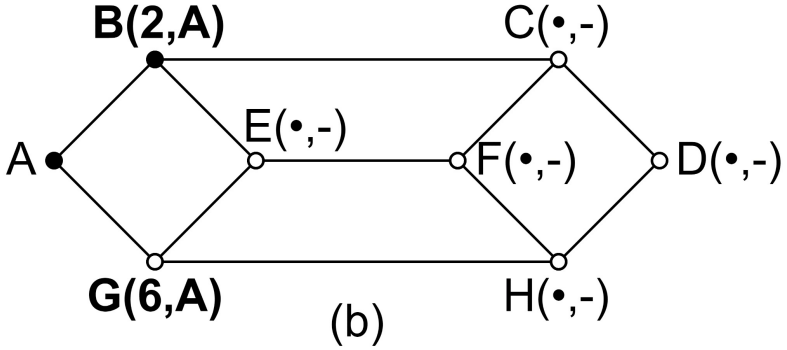
find the shortest path from A to D:

- labels may be permanent or tentative
- initially, no paths are known
  ➔ all nodes are labeled with infinity (TENTATIVE)
- discovery that label represents shortest possible path from source to node:
  ➔ label is made PERMANENT

1. Node A labeled as permanent (full black mark)
2. relabel all directly adjacent nodes with the distance to A (path length, nodes adjacent to source):
   - e.g. B(2,A) and G(6,A)
3. examine all tentatively labeled nodes; make the node with the smallest label permanent
   - e.g. B(2,A)
4. this node will be the new working node for the iterative procedure (i.e., continue with step 2.)

Non-Adaptive Shortest Path Routing (worksheet 1)





Example:

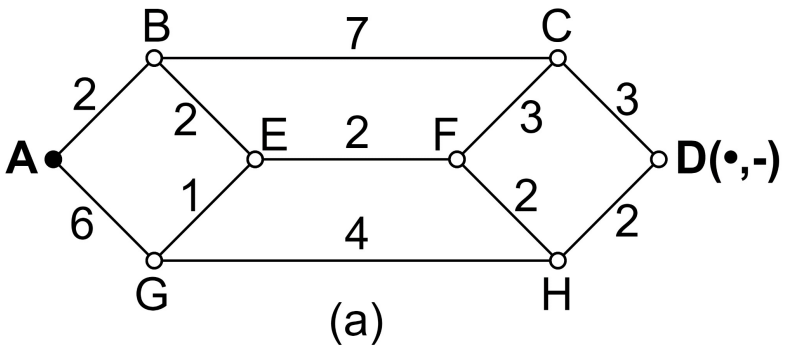
- link is labeled with distance
- node is labeled with distance from source along best known path

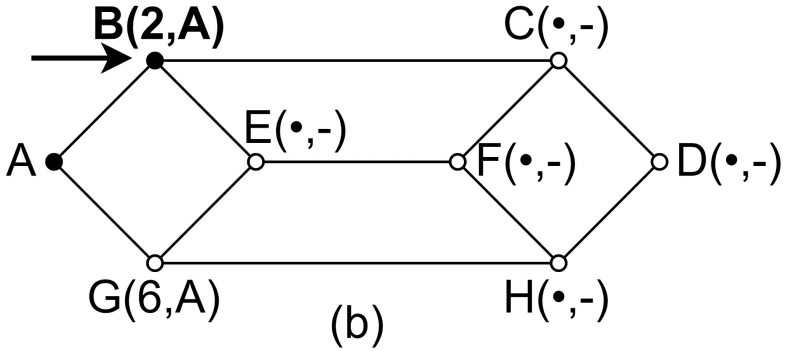
Procedure according to Dijkstra

find: the shortest path from A to D:

1. Node A labeled as permanent (black mark)
2. relabel all directly adjacent nodes with the distance to A
   - (path length, IS adjacent to the source):
   - e. g. B(2,A) and G(6,A)

Non-Adaptive Shortest Path Routing (worksheet 2)





Example:

- link is labeled with distance
- node is labeled with distance from source along best known path

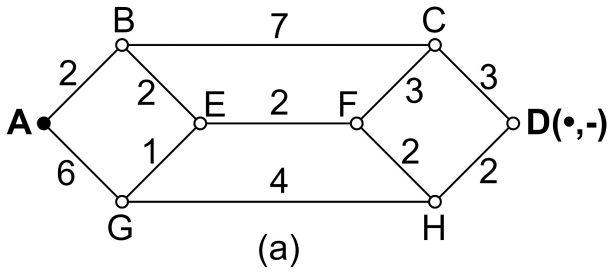
Procedure according Dijkstra

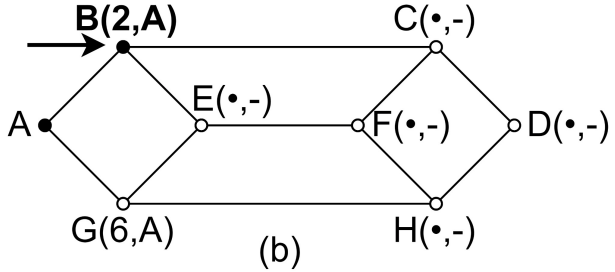
find: the shortest path from A to D:

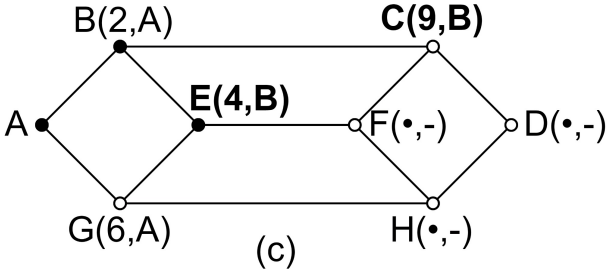
...

3. examine all tentatively labeled nodes
   - make the node with the smallest label permanent:
   - B(2,A)
4. this node will be the new working node for the iterative procedure
   - (i. e. continue with step 2)

Non-Adaptive Shortest Path Routing (worksheet 3)







Example:

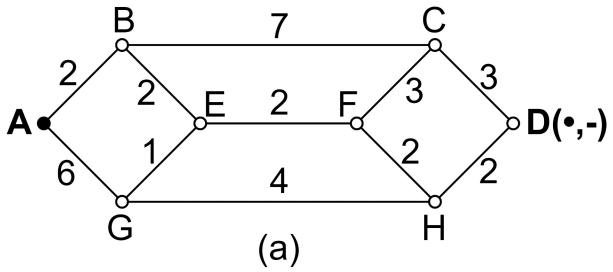
- link is labeled with distance
- node is labeled with distance from source along best known path

Procedure according to Dijkstra

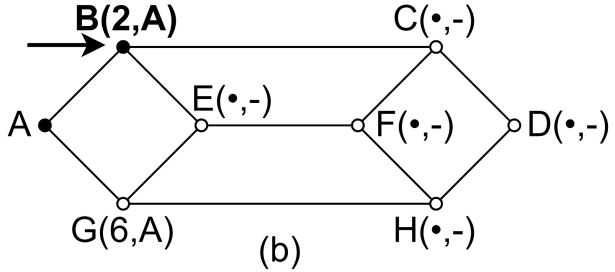
find the shortest path from A to D:

1. Node B has been labeled as permanent (black mark)
2. relabel all directly adjacent nodes with the distance to B
   (path length, nodes adjacent to source):
   - A (does not apply, because it is the origin),
   - i.e. E (4,B), C (9,B)

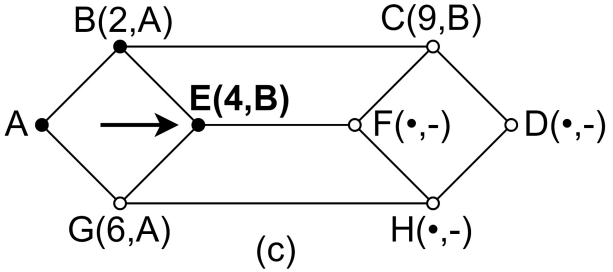
Non-Adaptive Shortest Path Routing (worksheet 4)



(a)



(b)



(c)

Example:

- link is labeled with distance
- node is labeled with distance from source along best known path

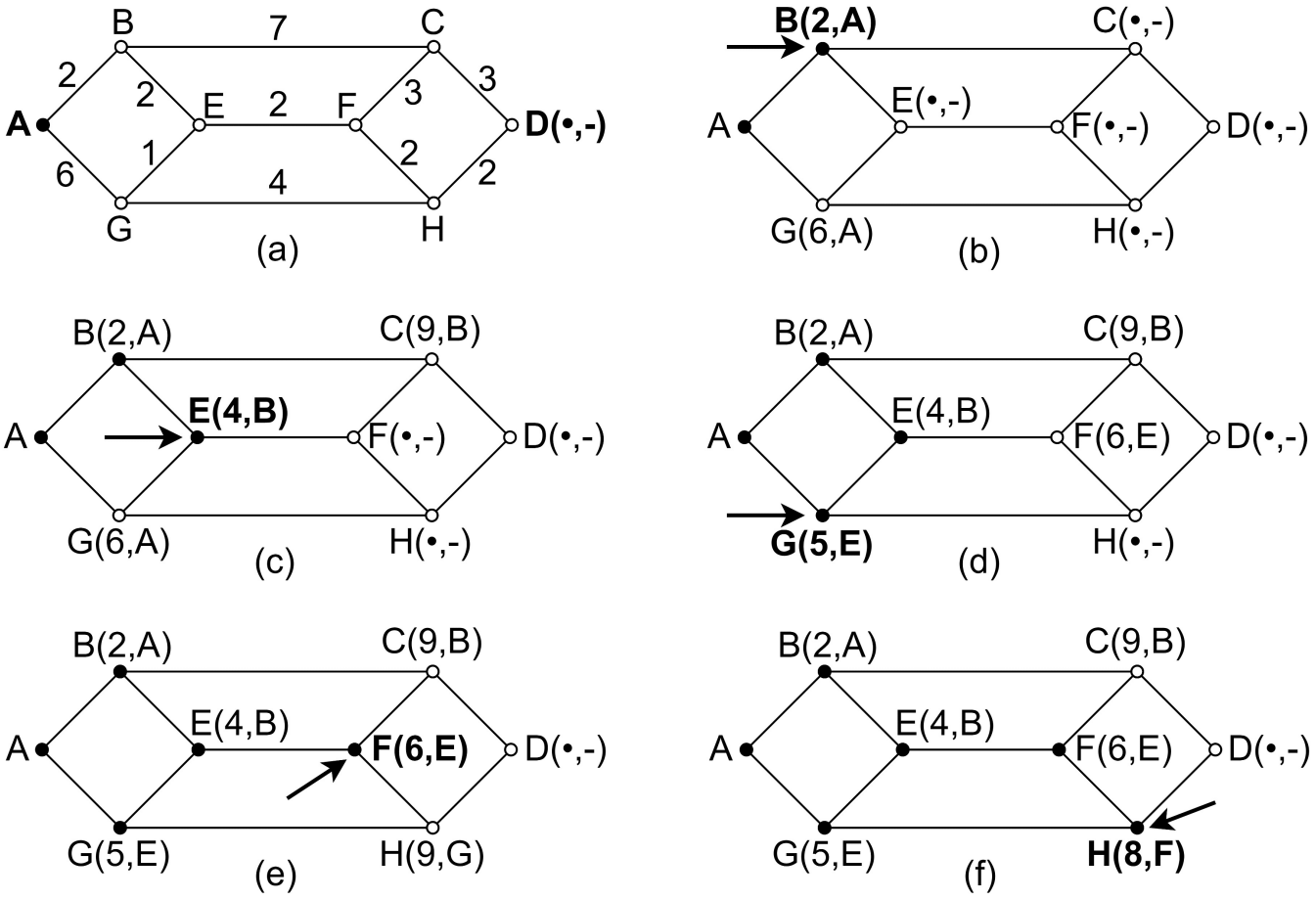
Procedure according to Dijkstra

find the shortest path from A to D:

1. ....
2. ....
3. examine all tentatively labeled nodes;
   - make the node with the smallest label permanent: e.g. E(4,B)
4. this node will be the new working node for the iterative procedure ...

Non-Adaptive Shortest Path Routing (worksheet 5)

And continue with source E …



5.2 Non-Adaptive Routing: Flooding

Principle:

IS transmits the received packet to all adjacent IS

- (except over the path it came in)
- but this generates "an infinite amount" of packets

Methods to **limit** packets

1. HOP COUNTER in the packet header
   - each IS decrements this hop counter
   - when the hop counter = 0, the packet is discarded
   - initialization for maximum path length (if known);
     - worst case: subnet diameter

2. each STATION REMEMBERS THE PACKETS THAT HAVE ALREADY BEEN TRANSFERRED and deletes them upon recurrence
   - source router inserts seq.no. into packets received from hosts
   - each router needs an "already seen seq.no." list per source router
   - packet with sequence number on list is dropped
   - sequence number list must be prevented from growing indefinitely
     - store only upper-counter / highest sequence number(s)

Flooding

Variation: Selective Flooding:

- do not send out on every line
- IS transmits received packet to adjacent stations, LOCATED IN THE DIRECTION OF THE DESTINATION
- with 'regular' topologies this makes sense and is an optimization
- but some topologies do not fit well to this approach

Comment:

- geographically-oriented routing: some recent interest for mobile scenarios (e.g., vehicular networks)

Flooding: Evaluation and use

- overhead: not practical in most applications
- extremely robust: military use
- reaches all IS: e.g., exchange of control data between nodes
- initialization phase: does not need information about topology
- always finds shortest path: can be used as benchmark

6. Adaptive Routing

Class ADAPTIVE ALGORITHMS

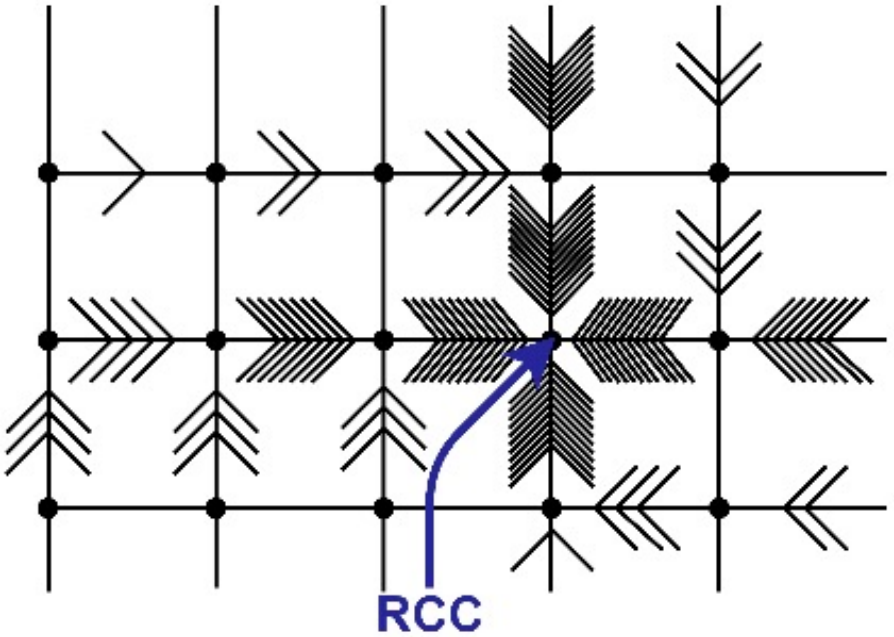
- decisions are based on current network state
  - measurements / estimates of the topology and traffic volume
- further sub-classification into
  - centralized algorithms
  - isolated algorithms

  and
  - distributed algorithms

6.1 Adaptive Centralized Routing

Principle:

- in the network:
  - RCC (Routing Control Center)
- each IS sends periodically information about the current status to the RCC
  - list of all available neighbors
  - actual queue lengths
  - line utilization, etc.
- Routing Control Center RCC
  - collects information
  - calculates the optimal path for each IS pair
  - generates routing tables and distributes these to the ISs



Example: TYMNET

- packet exchanging network
- 1000 nodes/IS
- virtual circuits

Adaptive Centralized Routing

Characteristics:

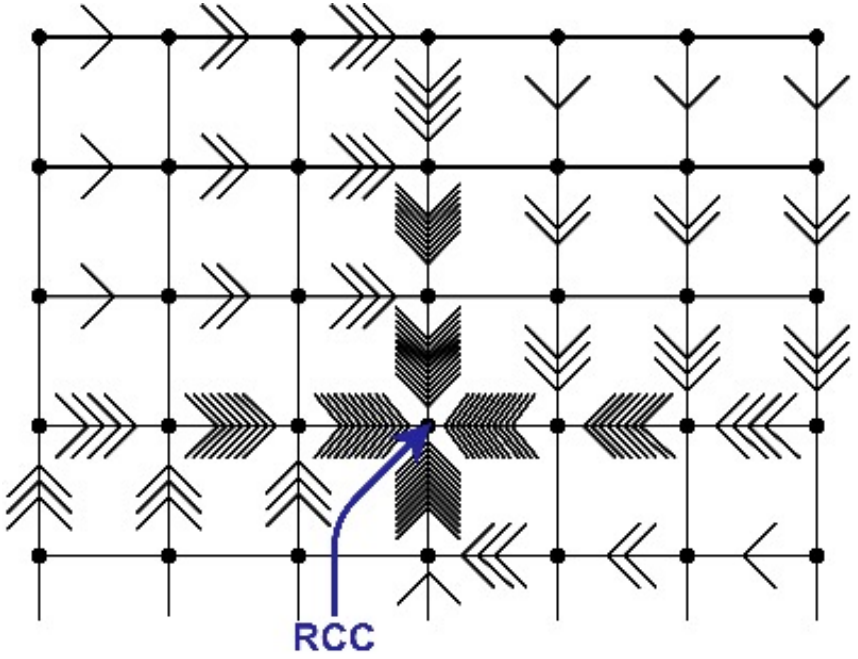
- Routing Control Center RCC has complete information

➔ perfect decisions

- and IS is free of routing calculations

but

- re-calculations quite often necessary (approx. once/min or more often)
- low robustness
- no correct decisions if network is partitioned
- IS receive tables at different times
- traffic concentration in RCC proximity



6.3 Adaptive Distributed – Distance-Vector Routing

Distance-Vector Routing

Group of DISTANCE VECTOR ROUTING ALGORITHMS

- also known as
  - distributed Bellman-Ford algorithm, Ford-Fulkerson algorithm

Usage

- was the original ARPANET routing algorithm
- has been used in the Internet as
  RIP (ROUTING INFORMATION PROTOCOL)

Basic principle

- IS maintains table (i.e., vector) stating
  - best known distance to destinations
  - and line to be used
- ISs update tables
  - by exchanging routing information with their neighbors

Distance-Vector Routing - Foundations

Each IS maintains routing table with one entry per router in the subnet
- estimation of distance (hops, delay, packets queued, ...) to dest.
- outgoing line to be used for that destination

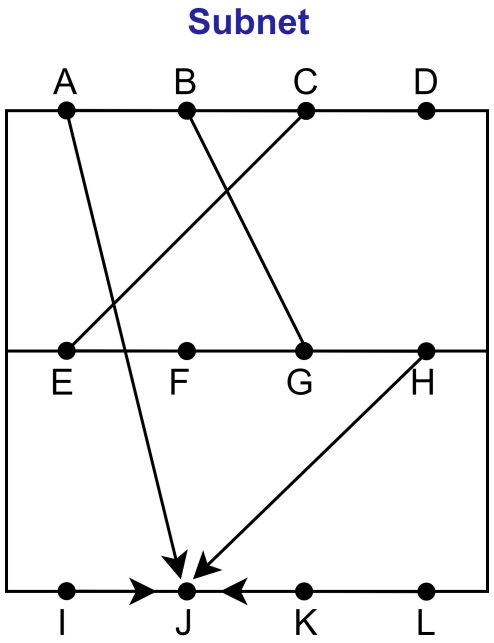
Each IS is assumed to know the "distance(s)" to each neighbor
- number of hops (= 1)
- delay (echo packets)
- queue length (e.g., used in the ARPANET),...

IS sends lists with estimated distances to all destinations periodically to its neighbors Y
- e.g., Internet RIP every 30 sec, maximum distance 15 hops

X receives list with distances from neighbor Y, e.g., D(Z) from Y to Z
- distance X to Y: d
- distance Y to Z: D(Z)
- i.e. distance X to Z (via Y): D(Z) + d

IS calculates a new routing table from the received lists, containing
- destination IS, preferred outgoing path, "distance"

Distance-Vector Routing – Example



delays at and of nodes A/I/H/K/. (column).
to nodes A;B;C;D… (row)

To	routing table of A	routing table of I	routing table of H	routing table of K	new estimated delay from J	line
A	0	24	20	21	**8**	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	**12**	H
I	21	0	14	22	**10**	I
J	9	11	7	10	0	-
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA Delay=8	JI Delay=10	JH Delay=12	JK Delay=6	new routing table for J	

Previous routing table will not be taken into consideration
➔ Reaction to deteriorations!?

Distance Vector Routing – Adaptation

Information distribution over new

- short paths (with few hops): fast
- long paths with many hops: SLOW

Example: route **improvement**

- previously: A unknown
- later: A connected with distance 1 to B, this will be announced
- Note: Synchronous update used here for simplification
- distribution proportional to topological spread

A	B	C	D	E	
	∞	∞	∞	∞	Initially
	1	∞	∞	∞	After 1 exchange
	1	2	∞	∞	After 2 exchanges
	1	2	3	∞	After 3 exchanges
	1	2	3	4	After 4 exchanges

Distance Vector Routing – “Count to Infinity”

Example: route **deterioration**,
(here: connection destroyed)

- A previously known, but now detached
- the values are derived from (incorrect) connections of distant IS

Comment

- limit “infinite” to a finite value, depending on the metrics
  - example:
    "infinite = maximum path length + 1"

A_1	B_1	C_1	D_1	E_1	
	1	2	3	4	Initially

B: no connection directly to A,
but C reports distance CA=2
i. e. BA = BC+ CA = 1 + 2 = 3
actually wrong!

3	2	3	4	After 1 change
3	4	3	4	After 2 changes
5	4	5	4	After 3 changes
5	6	5	6	After 4 changes
7	6	7	6	After 5 changes
7	8	7	8	After 6 changes
…				
∞	∞	∞	∞	

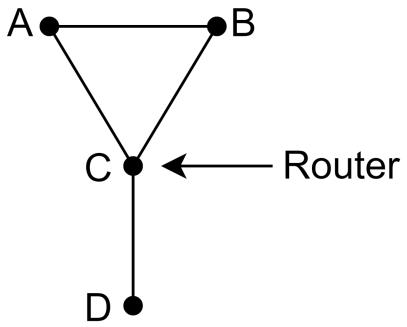
Distance Vector R. Variant “Split Horizon Algorithm”

Objective - based on the Distance Vector principle
- to improve the "count to infinity" property

Principle
- in general, to announce the "distance" to each neighbour
- special case:
  - if neighbour Y exists on the reported route,
  - then X reports the response "false" to Y

➔ distance X (via Y) according to arbitrary i: ∞

Example: deterioration,
e.g. connection destroyed
- B to C: A = ∞ (real),
- C to B: A = ∞ (because A is on path), ...



Note:
still poor, depending on topology, example:
- connection CD is removed
- A receives "false information" via B
- B receives "false information" via A

➔ slow distribution (just as before)

6.4 Adaptive Distributed – Link State Routing

Basic principle: each IS

- measures the "distance" to the directly adjacent ISs,
- distributes information,
- calculates ideal routes using information received from all IS

Steps

1. determine the address of adjacent IS
2. measure the "distance" (delay, ...) to neighbor IS
3. organize the local link state information in a packet
4. distribute the information to all IS
5. calculate the route based on the information of all IS

Usage

- introduced into ARPANET in 1979, nowadays most prevalent
- IS-IS (Intermediate System-Intermediate System)
  - developed by DECNET
  - also used as ISO CLNP in NSFNET
  - Novell Netware developed its own variation from this (NLSP)
- OSPF (Open Shortest Path First)
  - since 1990 Internet RFC 1247 and several newer RFCs

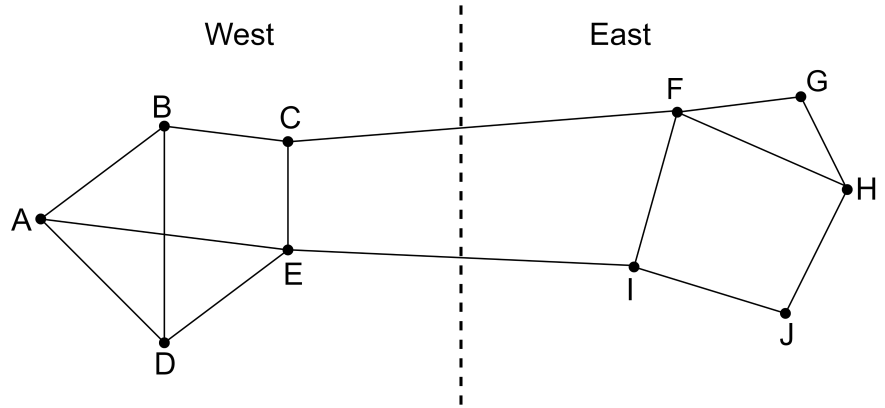
Link State Routing

1. Phase:
   gather information about the adjacent intermediate systems
   - initialization procedure:
     - new IS:
       - sends a HELLO message over each L2 channel
     - adjacent IS:
       - responds with its own address, unique within the network

Link State Routing

2. Phase:
define the "distance"

- distance is generally defined as delay
- detection via transmission of ECHO messages, which are reflected at receiver
- multiple transmission:
  - improved average value
  - with or without payload:
    - with payload is usually better,
  - but "with load" may lead to an "oscillation" of the load:

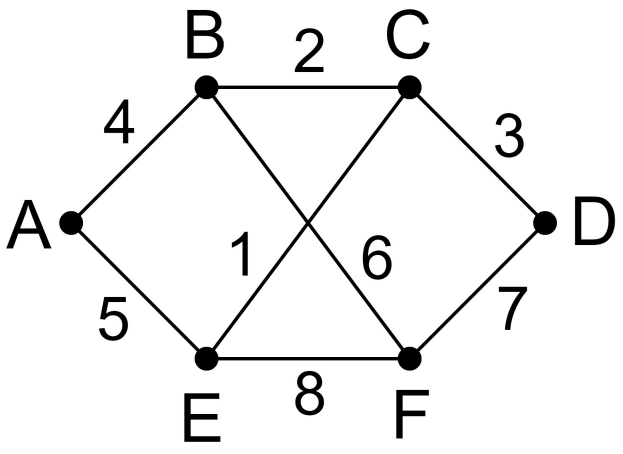


  - after each new routing table the other link CF or EI is charged

Link State Routing

3. Phase:
organize the information as link state packet

- including own address, sequence number, age, "distance"
- timing problems: validity and time of sending
  - periodically
  - in case of major changes



Link State Packets:

A	
Seq.	
Age	
B	4
E	5

B	
Seq.	
Age	
A	4
C	2
F	6

C	
Seq.	
Age	
B	2
D	3
E	1

D	
Seq.	
Age	
C	3
F	7

E	
Seq.	
Age	
A	5
C	1
F	8

F	
Seq.	
Age	
B	6
D	7
E	8

Link State Routing

4. Phase:
distribute the local information to all IS

- flooding this information as “link state packet” (LSP)
  - very robust
  - sequence number in packets
- problem: inconsistency
  - varying states simultaneously available in the network
  - indicate and limit the age of packet,
    i.e. IS removes packets that are too old

Link State Routing

5. Phase:
compute new routes
- each node has full topology (by combining all LSPs)
- each node can run Dijkstra's shortest path algorithm

→ each IS for itself
- possibly large amount of data available

Handling changes in network topology!?
- new nodes?
- disappearing nodes? ➔ home work ;-)

7 Addressing

3 types of identifiers: names, addresses and routes [Shoch 78]

"The NAME of a resource indicates WHAT we seek,
an ADDRESS indicates WHERE it is, and
a ROUTE says HOW TO GET THERE."

Objectives:

- global addressing concept for ES
- simplified address allocation
- addresses independent from
  - type and topology of the subnetworks
  - number and type of the subnetworks to which the ES have been connected
  - location of a source ES

Different approaches for addressing concepts feasible

- Not only IP addresses

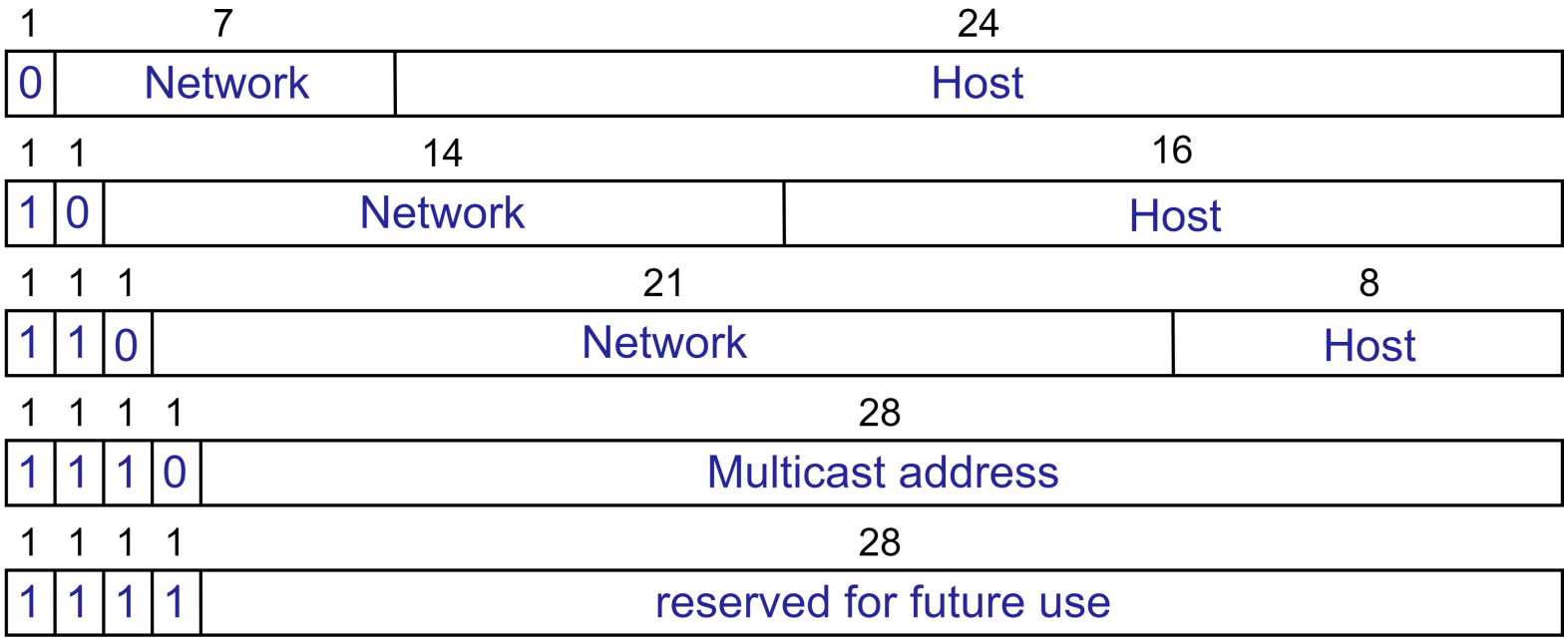
Internet Addresses (IPv4)

Global addressing concept for ES (and IS) in the Internet

- 32 bit address (amount limited!) → $2^{32} \approx 4*10^9$ different addr.
- each address is unique worldwide
- structure: Net-ID (Subnet-ID), ES-ID

Originally structured in classes

overall 4 byte (32 bit)



Internet Addresses (IPv4)

Notation

- decimal value for each byte (0...255)
- subdivided by dots
- value range: 0.0.0.0 ... 255.255.255.255

Formats: 5 classes (**historical** view)

A:	1.0.0.0	up to	127.255.255.255
B:	128.0.0.0	up to	191.255.255.255
C:	192.0.0.0	up to	223.255.255.255
D:	224.0.0.0	up to	239.255.255.255 (Multicast)
E:	240.0.0.0	up to	247.255.255.255

Broadcast addresses: (convention: 11...1 for Host-ID)

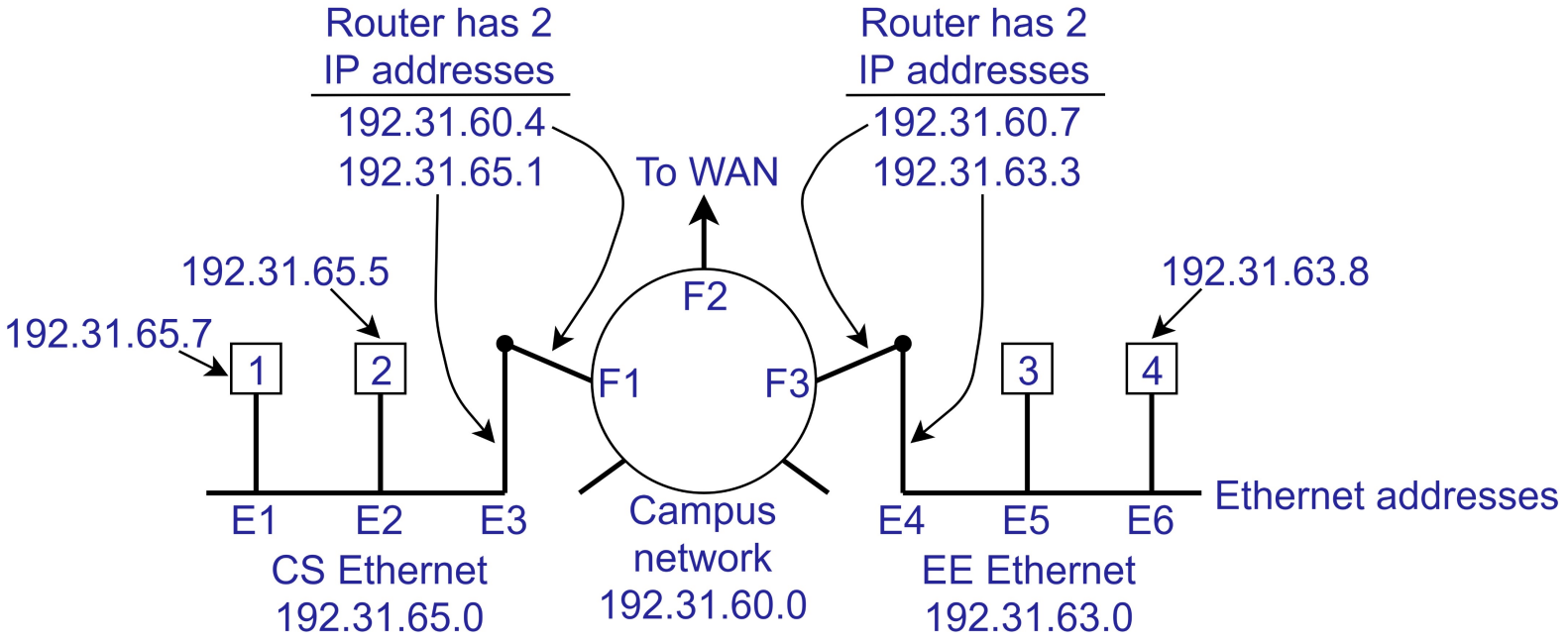
Classful network addressing architecture has mainly been replaced by **Classless Inter-Domain Routing** (CIDR)

Internet Addresses (IPv4)

Address allocation

- class allocation and network range:
  - by a central authority
  - Network Information Center NIC
- end system
  - local
  - possibly forming a subnetwork

Example network:

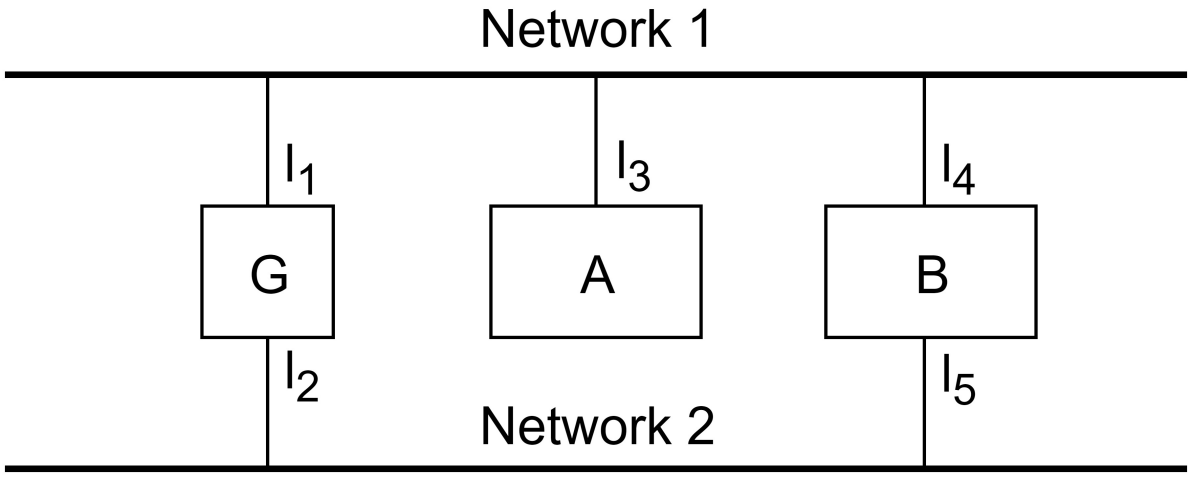


Internet Addresses (IPv4): A Critical Review

Addresses IDENTIFY "NETWORK CONNECTIONS",
not the ES

- "multi-homed" ES have more than one address
- a change of the connection forces the modification of the address
- the address has an impact on the chosen route (constitutes a problem in the mobile area)

Example: A cannot reach B via address I5 if G fails



Further problem:

- Number of addresses is limited

Internet Addresses: IPv6

IP Version 6 (IPv6)

- 16 byte length (instead of 4 byte length, i. e. approx. 3×10^{38})

Distribution

- provider-based: approx. 16 mio. companies distribute addresses
- geographic-based: distribution as it is today
- link, site-used: address relevant only locally (security, Firewall concept)

e. g. new: Anycast

- sending data to an individual of a group
- e. g. the one who is geograhically the closest