COMP 431
Internet Services & Protocols

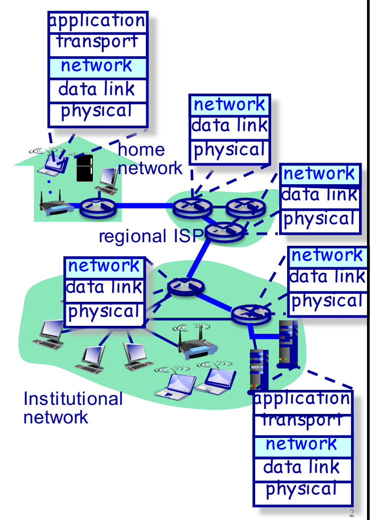
The Network Layer:
Routing & Addressing

Jasleen Kaur

March 31, 2020

The Network Layer: Routing & Addressing
Outline

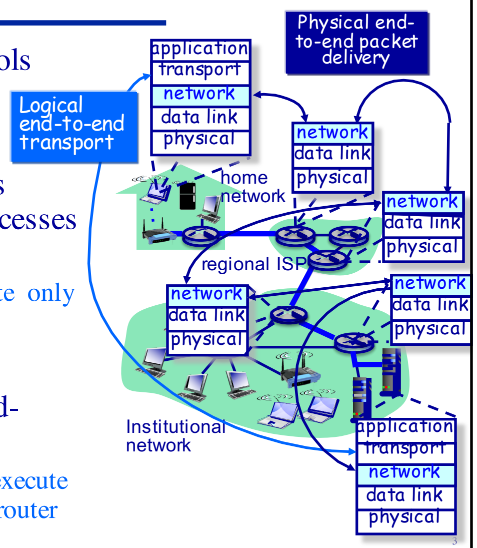
- Network layer services
- Routing algorithms
  - » Least cost path computation algorithms
- Hierarchical routing
  - » Connecting networks of networks
- IP Internet Protocol
  - » Addressing
  - » IPv6
- Routing on the Internet
  - » Intra-domain routing
  - » Inter-domain routing



The Network Layer

Network layer functions

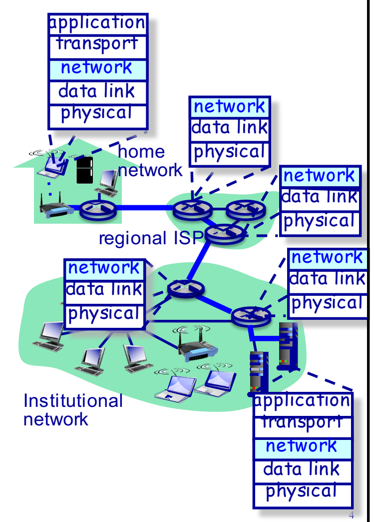
- Application-layer protocols define when and how messages are sent
- Transport-layer protocols deliver data between processes on different end-systems
  - » Transport protocols execute only on end systems
- Network-layer protocols deliver data from one end-system to another
  - » Network layer protocols execute on *every* end-system and router



The Network Layer

Network layer functions

- The network-layer provides four important functions:
  - » *Addressing*: the means by which end systems identify each other
  - » *Path determination*: the route taken by packets from source to destination
  - » *Switching*: the movement of packets from an input interface to an appropriate output interface
  - » *Call setup*: The establishment of a virtual circuit from sender to receiver



Network-Layer Service Models

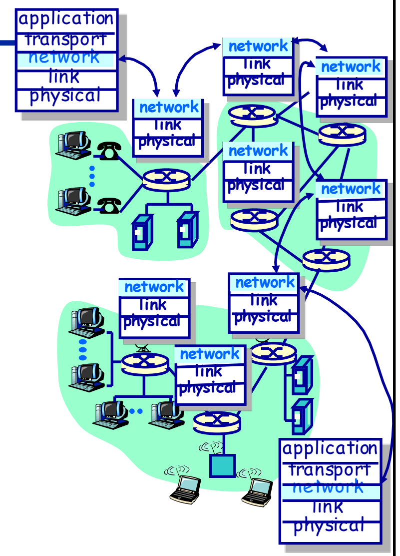
Datagram *v.* Virtual Circuit networks

- IP networks:
  - » Data exchanged among computers
    - ❖ "Elastic" service, no strict timing requirements
  - » "Smart" end systems (computers)
    - ❖ Can adapt, perform control, error recovery
    - ❖ Simple inside the network, complexity at "edges"
  - » Operates over "any" link layer technology
    - ❖ Uniform service difficult
    - ❖ But interoperation "easy"
  - » New services easily added (most services implemented at the edge)
- ATM Networks
  - » Evolved from telephony
  - » Human conversation:
    - ❖ Strict timing, reliability requirements
    - ❖ Need for guaranteed service
  - » "Dumb" end systems (telephones)
    - ❖ Tremendous complexity inside the network
  - » No interoperation with other networks
  - » New services requires "the network" to be upgraded

The Network Layer: Routing & Addressing

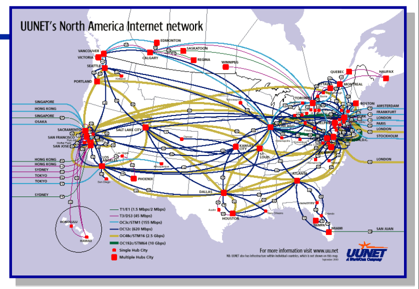
Outline

- Network layer services
- Routing algorithms
  - » Least cost path computation algorithms
- Hierarchical routing
  - » Connecting networks of networks
- IP Internet Protocol
  - » Addressing
  - » IPv6
- Routing on the Internet
  - » Intra-domain routing
  - » Inter-domain routing



Routing Algorithms

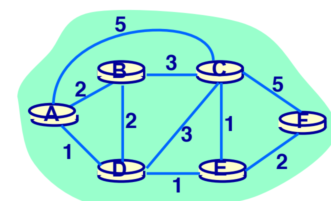
Least-cost path computation



- Goal: To determine a “good” path through the network from source to destination
- Graph abstraction for routing algorithms:
  - » Nodes are routers
  - » Edges are physical links
  - » Edges have a “cost” metric
    - ❖ Cost can be delay, monetary cost, level of congestion, *etc.*
- “Good” path typically means minimum cost path
  - » Also shortest path, …
- (But often ISPs define “good” in terms of business models)

Routing Algorithms

Taxonomy

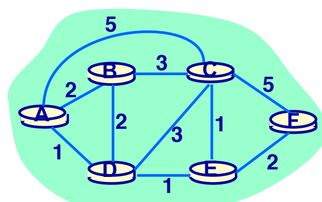


- Global or decentralized?
- Global — all routers have complete graph (topology, costs)
  - » “Link state” algorithms
- Decentralized — router knows link costs to physically connected adjacent nodes
  - » Run iterative algorithm to exchange information with adjacent nodes
  - » “Distance vector” algorithms
  - » (RIP — *Routing Information Protocol*)
- Static or dynamic?
  - » Static — routes change slowly over time
  - » Dynamic — routes change more quickly
    - ❖ Periodic updates, or
    - ❖ Updates in response to link outages or cost changes

Global Routing Algorithms

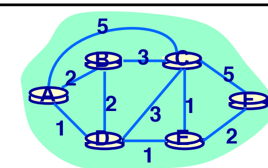
A link-state routing algorithm

- Uses Dijkstra's shortest path graph algorithm
- Complete network topology and link costs known at *all* nodes
  - » Accomplished via *link state flooding*
  - » All nodes learn the "same" topology and cost data
- Each node computes least cost paths from itself to all other nodes
  - » Produces a *routing table* for that node
  - » All nodes compute consistent routing tables
- Algorithm complexity:
  - » N nodes (routers) in the network
  - » $N \times (N+1)/2$ comparisons
  - » (More efficient implementations possible)



Link State Routing

Dijsktra's Algorithm



N is the set of nodes to which we have computed the minimum cost path
$D(x)$ is the current minimum cost path to x
$c(n,m)$ is the cost of the link from n to m

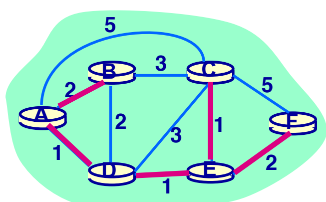
```
Initialization:
 N = {A}
 for all nodes v
   if v adjacent to A
     then D(v) = c(A,v)
     else D(v) = infinity

Loop
  find node w not in N such that D(w) is a minimum
  add node w to N
  update D(v) for all nodes v adjacent to w and not in N:
      D(v) = min( D(v), D(w) + c(w,v) )
   /* new cost to node v is either old cost to v or known
      shortest path cost to w plus cost from w to v */
until all nodes in N
```

Link State Routing

Dijkstra's algorithm: example

Step	start N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infinity	infinity



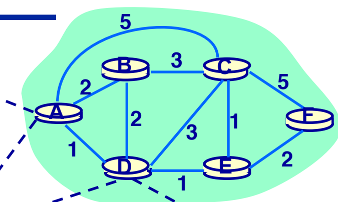
N is the set of nodes to which we have computed the minimum cost path

$D(x)$ is the current minimum cost path to x

$p(x)$ is the predecessor of x on the current minimum cost path to x

Link State Routing

Link state routing table



Link State Routing Table for A

destination	***first*** node in least cost path	
B	B	**B**
C	E → C	**D**
D	D	**D**
E	D → E	**D**
F	E → F	**D**

Link State Routing Table for D

destination	***first*** node in least cost path	
A	A	**A**
B	B	**B**
C	E → C	**E**
E	E	**E**
F	E → F	**E**

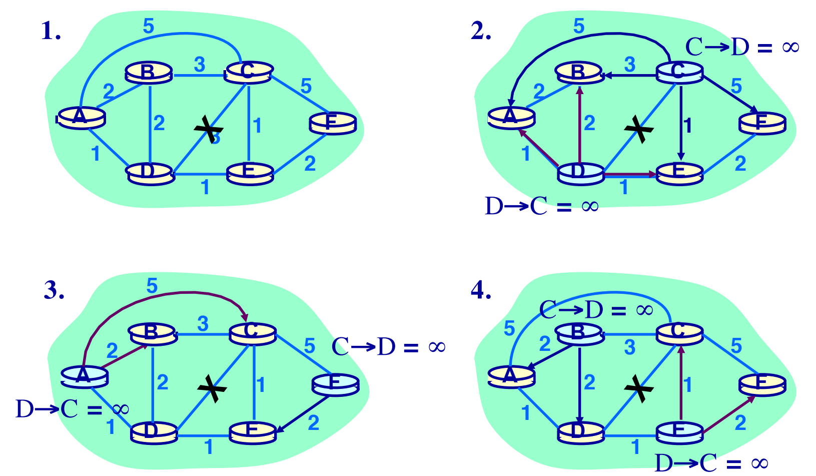
Link State Routing

Link State Flooding Algorithm

- The data stored for an edge in the graph (the link between nodes *X* and *Y*) consists of:
  - » Cost from *X* to *Y* (*X*-*Y*) and from *Y* to *X* (*Y*-*X*)
  - » A unique timestamp for the last update to each cost
- A node that discovers a change in cost for one of its attached links forwards the update to all adjacent nodes
- A node receiving an update forwards it based on a comparison of the update timestamp and the timestamp on its local data for the link:
  - » Update is later (or new): Forward to all adjacent nodes (except sender) and update local data
  - » Update is earlier: Send local data back to sender
  - » Update is equal: Do nothing

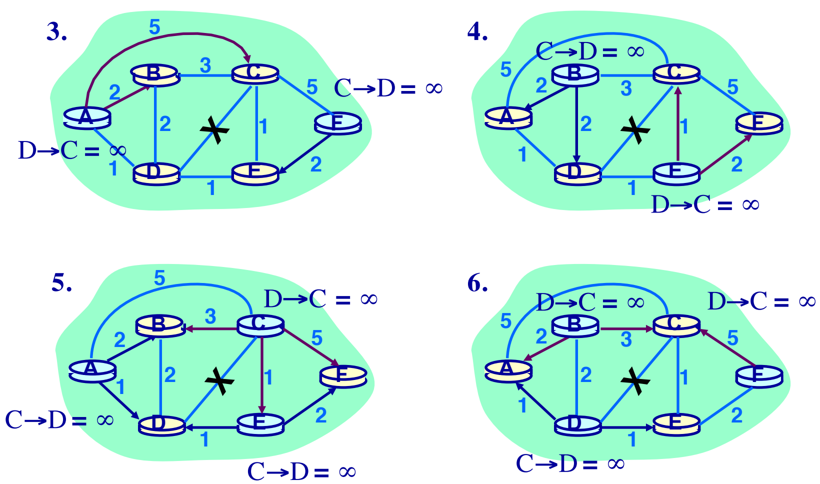
Link State Flooding Algorithm

Example



Link State Flooding Algorithm

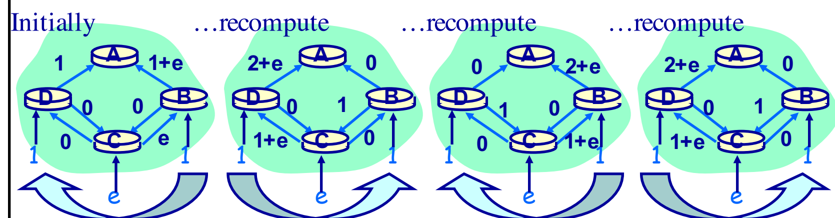
Example (Continued)



Link State Routing

Oscillating routes

- "Route oscillations" are possible in link state algorithms
- Let the link cost equal the amount of carried traffic
  - » Assume the link cost is updated as traffic changes



Node	Path to A
D	A
C	B
B	A

Node	Path to A
D	A
C	D
B	C

Node	Path to A
D	C
C	B
B	A

Node	Path to A
D	A
C	D
B	C