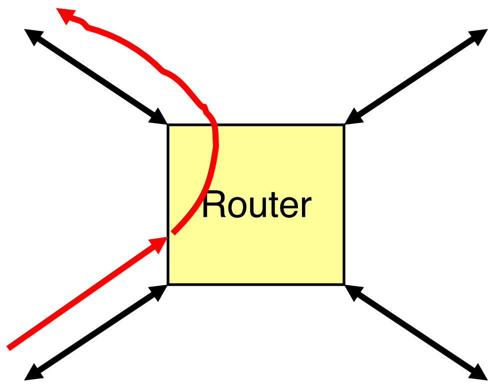
IP Layer: IP Routing: Types of routing algorithms

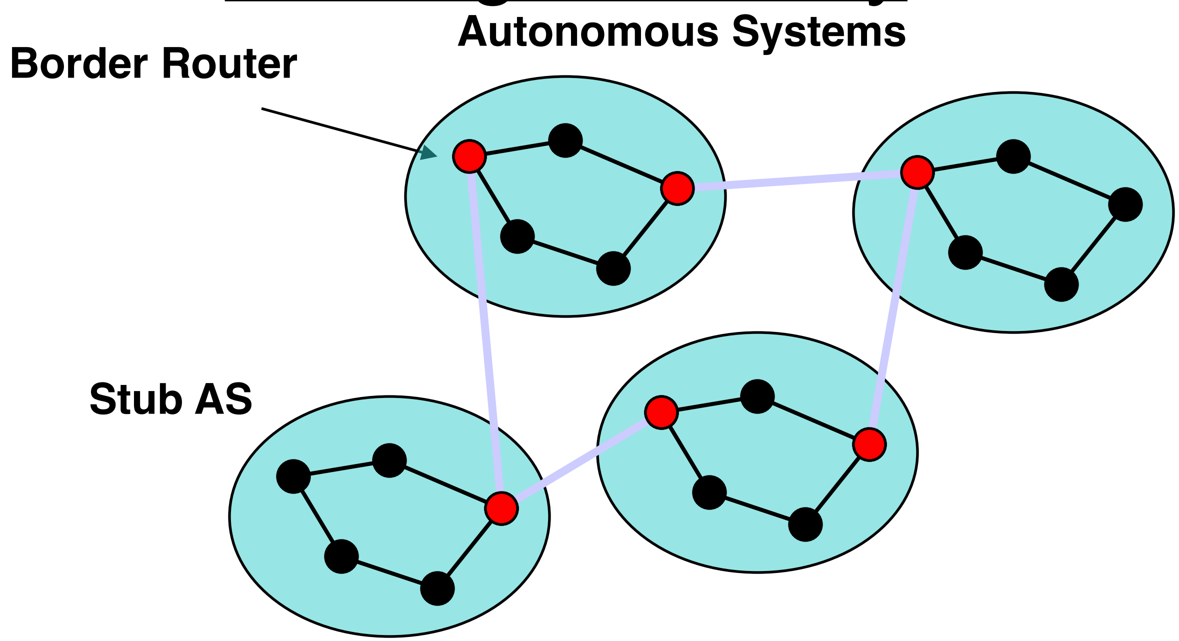
ECE 50863 – Computer Network Systems

Router Operation



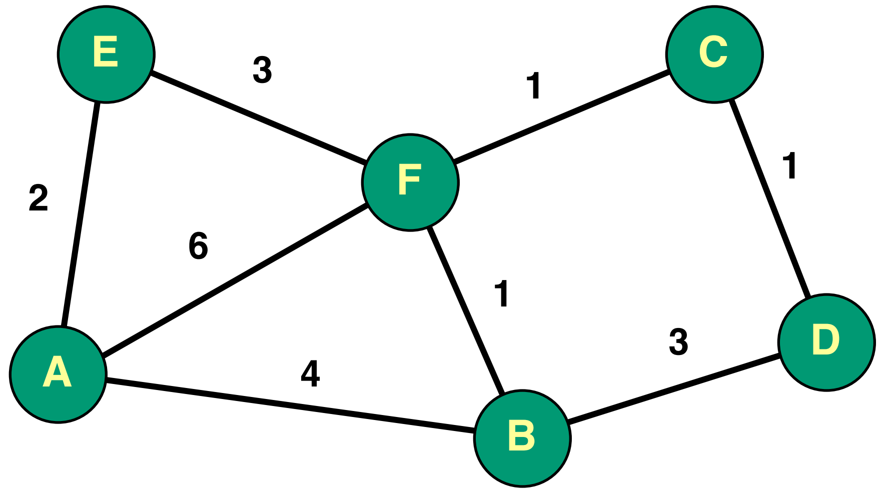
- When Packet Arrives at Router
  - Examine header to determine intended destination
  - *Look up in table to determine next hop in path*
  - Send packet out appropriate port
- Terminology
  - Each router *forwards* packet to next router
  - Overall goal is to *route* packet from source to destination
- Next topic: How to generate the routing table?

Routing Hierarchy



- Autonomous System
  - Network administered by single entity
- Intradomain Routing
  - Routing within single AS
  - Typically ~ 200 nodes
- Interdomain Routing
  - Routing between AS's

Graph Model



- Represent each router as node
- Direct link between routers represented by edge
  - Symmetric links ⇒ undirected graph
- Edge “cost” c(x,y) denotes measure of difficulty of using link

- Task
  - Determine least cost path from every node to every other node
    - Path cost d(x,y) = sum of link costs

Types of intra-domain routing algorithms

- Centralized
  - Collect graph structure in one place
  - Use standard graph algorithm
  - Disseminate routing tables
- Partially Distributed
  - Every node collects complete graph structure
  - Each computes shortest paths from it
  - Each generates own routing table
  - "Link-state" algorithm
- Fully Distributed
  - No one has copy of graph
  - Nodes construct their own tables iteratively
  - Each sends information about its table to neighbors
  - "Distance-Vector" algorithm

Distance Vector: Example Routing Tables

Table for A		
Dest	Cost	Next Hop
A	0	A
B	4	B
C	6	E
D	7	B
E	2	E
F	5	E

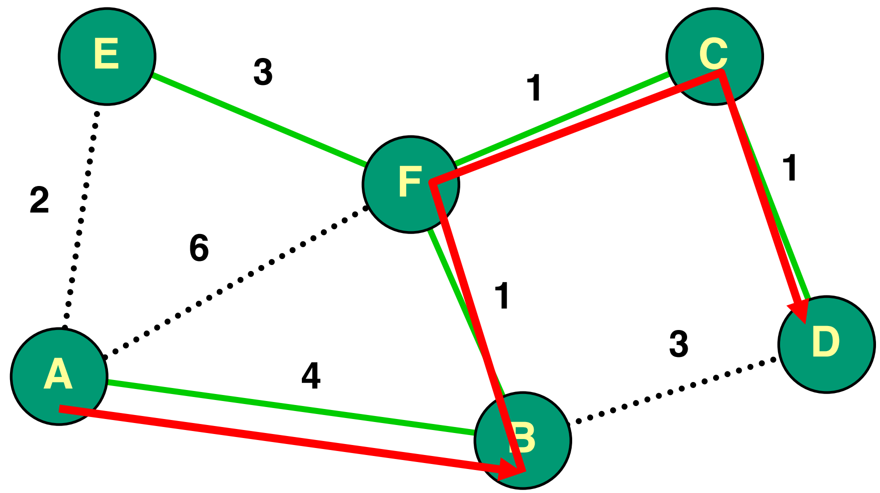


Table for B		
Dest	Cost	Next Hop
A	4	A
B	0	B
C	2	F
D	3	F
E	4	F
F	1	F

- Multiple choices for B
  - B-D, B-F-C-D
- A may not know route B takes
- A does not know the existence of links such as B-F, F-C etc.

Issues in constructing routing tables

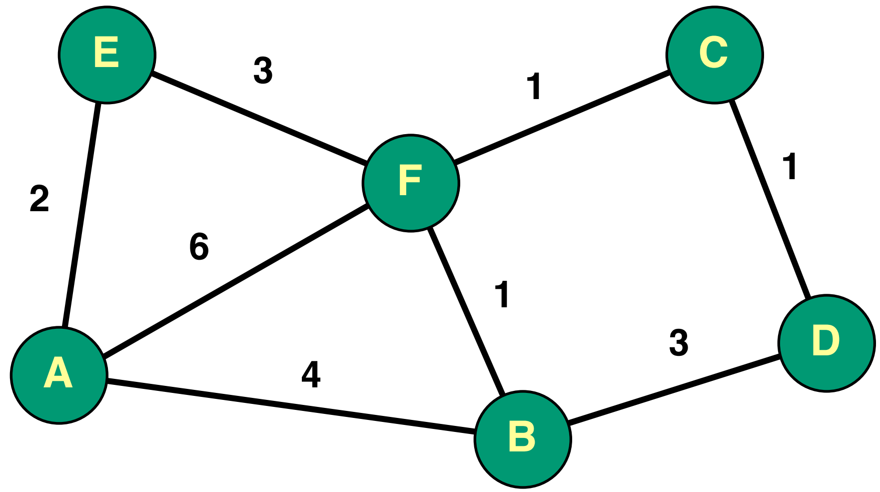
- Ensure packets don’t get stuck in a loop
- Find “good” paths
- Adapt to changes in edge costs
- Adapt to failure of nodes

IP Layer: IP Routing: Distance Vector Routing Algorithm

ECE 50863 – Computer Network Systems

Distance-Vector Method

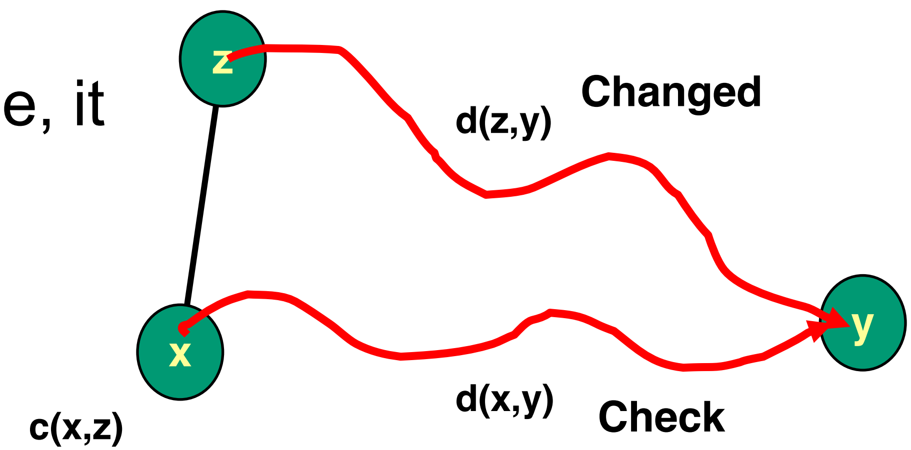
Initial Table for A		
Dest	Cost	Next Hop
A	0	A
B	4	B
C	∞	–
D	∞	–
E	2	E
F	6	F



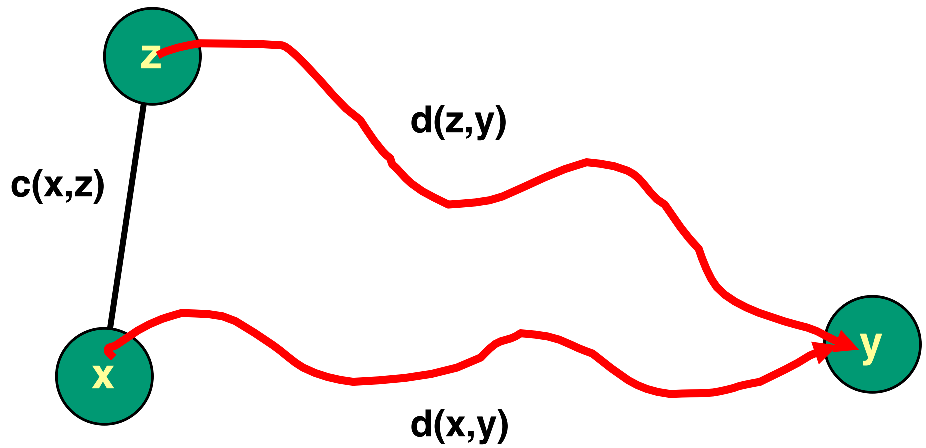
- Idea
  - At any time, have cost/next hop of best known path to destination
  - Use cost ∞ when no path known
- Initially
  - Only have entries for directly connected nodes

Routing Algorithm

- Periodically, every node z sends each neighbor x a copy of its routing table
- When x receives the table, it
  runs Update(x,y,z) for every
  destination y
- Process occurs in "asynchronous" fashion
- Routing tables "eventually" converge



Distance-Vector Update



- Update(x,y,z)

  d ← c(x,z) + d(z,y) # Cost of path from x to y with first hop z

  if d < d(x,y)

  # Found better path

  return d,z # Updated cost / next hop

  else

  return d(x,y), nexthop(x,y) # Existing cost / next hop

What if Node Fails?

- D & F notice that C isn't responding
- Set entries to ∞
- Iterate

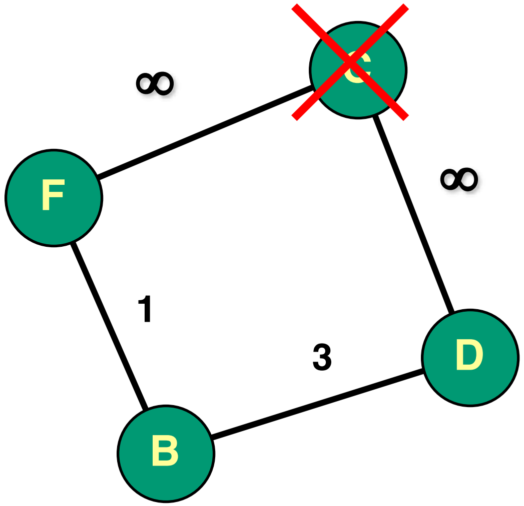


Table for D		
Dst	Cst	Hop
C	∞	–

Table for F		
Dst	Cst	Hop
C	∞	–

Failing Node Iterations

- What happens if B sends updates to D and F?

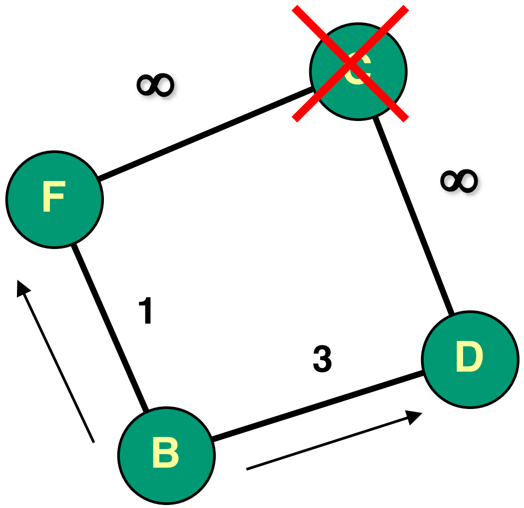


Table for B		
Dst	Cst	Hop
C	2	F

Table for D		
Dst	Cst	Hop
C	∞	–

Table for F		
Dst	Cst	Hop
C	∞	–

Better Route

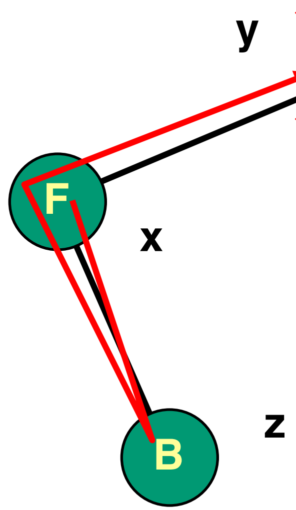
Table for D		
Dst	Cst	Hop
C	5	B

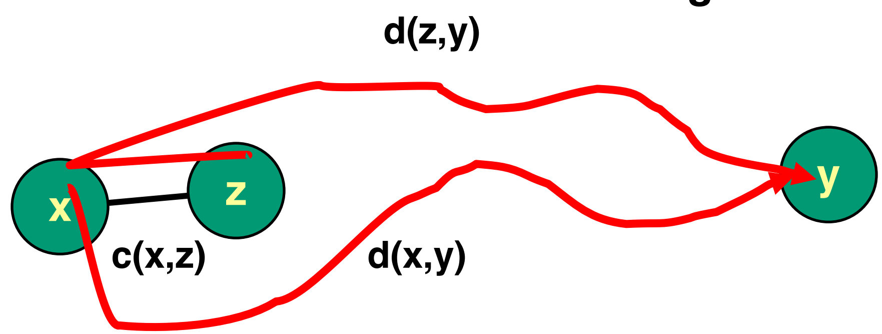
Better Route

Table for F		
Dst	Cst	Hop
C	3	B

Routing tables converge to incorrect values.

Revised Update Rule #1





Variants: "Split Horizon Rule",
"Split Horizon with Poison Reverse"

- Update(x,y,z)

  d ← c(x,z) + d(z,y) # Cost of path from x to y with first hop z

  if d < d(x,y) & x ≠ nexthop(z,y)

    # Found better path

    return d,z

  else

    return d(x,y), nexthop(x,y)

Iterations with Revision #2

- What happens if B sends updates to D and F if split horizon rule were added?

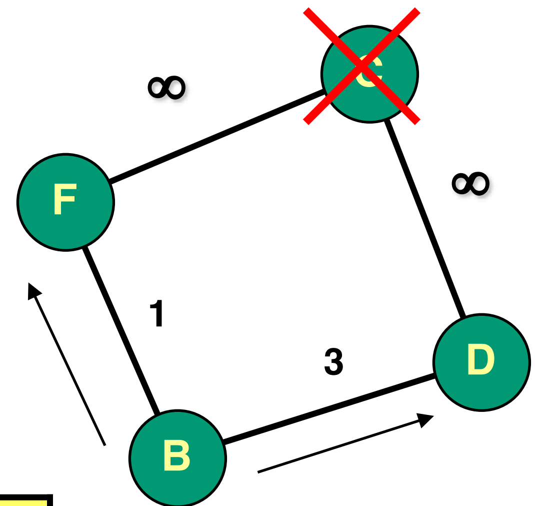


Table for B		
Dst	Cst	Hop
C	2	F

Better Route

Table for D		
Dst	Cst	Hop
C	∞	–

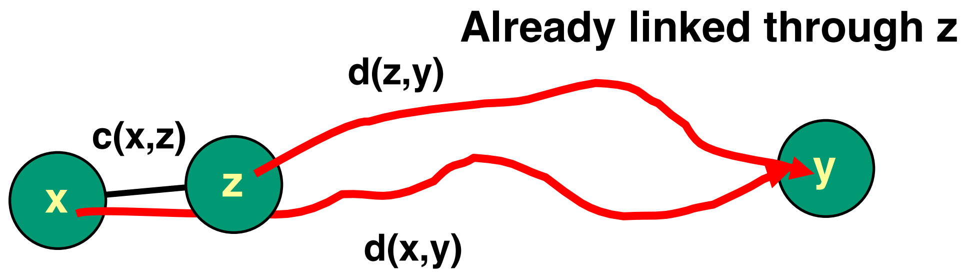
Table for D		
Dst	Cst	Hop
C	5	B

Table for F		
Dst	Cst	Hop
C	∞	–

No Change

F's table converges correctly, but B and D's tables do not converge correctly

Revised Update Rule #2



- Update(x,y,z)

  d ← c(x,z) + d(z,y) # Cost of path from x to y with first hop z

  if nexthop(x,y) = z | # Forced update

  (d < d(x,y) & x ≠ nexthop(z,y))

  # Forced update or found better path

  return d,z

  else

  return d(x,y), nexthop(x,y)

Iterations with Revision #2

Table for B		
Dst	Cst	Hop
C	2	F

Table for D		
Dst	Cst	Hop
C	∞	–

Table for F		
Dst	Cst	Hop
C	∞	–

No Change B -> F

Better Route B->D

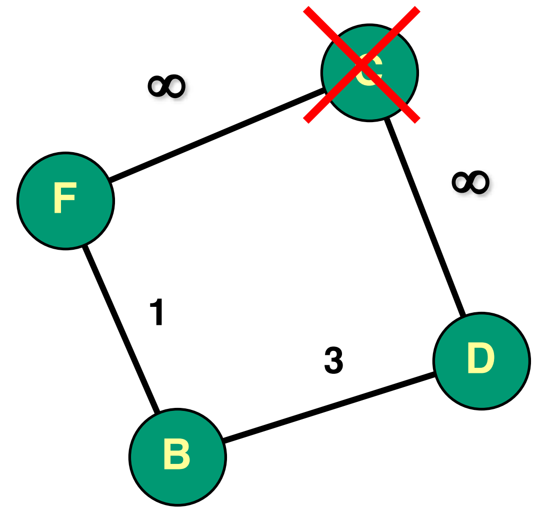
Table for D		
Dst	Cst	Hop
C	5	B

Forced Update (F->B)

Table for B		
Dst	Cst	Hop
C	∞	–

Forced Update (B -> D)

Table for D		
Dst	Cst	Hop
C	∞	–



If split horizon and forced update rules both used:

Routing tables of all routers converge properly.

IP Layer: IP Routing: Limitations of Distance Vector Routing

ECE 50863 – Computer Network Systems

Distance vector in more general topologies

- Split horizon and forced update rules do not suffice
- “Count to infinity” problem leads to slow convergence
  - Sequence of updates where routers keep increasing their costs until it reaches a large number
  - Note: different sequences of updates can lead to different results (”good” and “bad” sequences).
  - Distributed algorithm must converge under all sequences.
- Practice: Set infinity to a large number:
  - Routers count up until this value and this halts the “count to infinity” problem.
  - Quick fix, not a real solution.

Potential solutions

- Triggered updates: Send a neighbor an update each time a routing table entry changes.
  - Additional to periodic update: Send a neighbor an update every 30 seconds
  - Increases odds of convergence, not a guarantee.
- Path vector:
  - Send the full path to the destination, not just next hop.
  - Router A accepts update from Router B only if it is not in the path from B to the destination.
  - Generalization of distance vector.
  - More expensive (larger routing tables, larger updates)

What’s used in practice?

- Routing Information Protocol
  - Earliest IP routing protocol (1982 BSD)
  - Every link has cost 1. “Infinity” = 16
    - Limits to networks where everything reachable within 15 hops
  - Uses periodic and triggered updates
- Path vector not used as much
  - But ideas have influenced inter-domain routing algorithms.
- Increasing move away from distance vector to link state algorithms.

Ways to Compute Shortest Paths

- Centralized
  - Collect graph structure in one place
  - Use standard graph algorithm
  - Disseminate routing tables
- Partially Distributed
  - Every node collects complete graph structure
  - Each computes shortest paths from it
  - Each generates own routing table
  - “Link-state” algorithm
- Fully Distributed
  - No one has copy of graph
  - Nodes construct their own tables iteratively
  - Each sends information about its table to neighbors
  - “Distance-Vector” algorithm

Advantages of Link State Algorithms

- No count to infinity problems
- Convergence problems on failures significantly lowered
  - Every router knows the full topology
- More complex routing solutions possible – not just shortest path
  - Centralized router has full view of network topology

IP Layer: IP Routing
Link State Routing

ECE 50863 – Computer Network Systems

Ways to Compute Shortest Paths

- Centralized
  - Collect graph structure in one place
  - Use standard graph algorithm
  - Disseminate routing tables
- Partially Distributed
  - Every node collects complete graph structure
  - Each computes shortest paths from it
  - Each generates own routing table
  - "Link-state" algorithm
- Fully Distributed
  - No one has copy of graph
  - Nodes construct their own tables iteratively
  - Each sends information about its table to neighbors
  - "Distance-Vector" algorithm

OSPF Routing Protocol

- Open
  - Open standard created by IETF
- Shortest-Path First
  - Another name for Dijkstra’s algorithm
- Most Prevalent Intradomain Routing Protocol
- Focus of this lecture:
  - How to obtain graph structure at each node.
- Once graph structure obtained, Dijkstra’s algorithm used to compute shortest paths

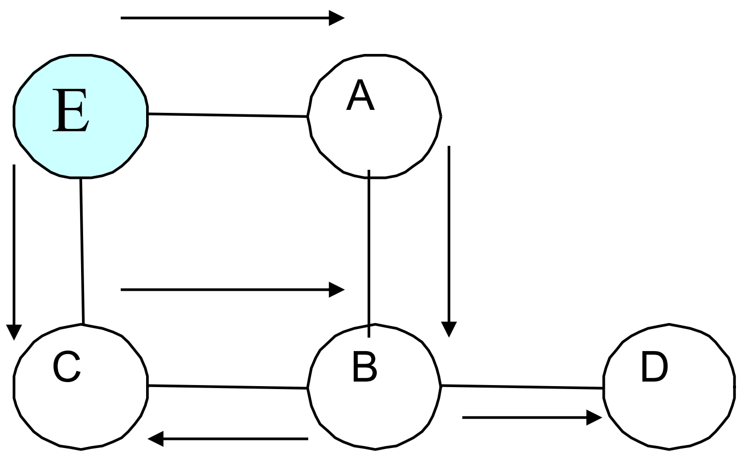
OSPF Reliable Flooding

- Each router transmits Link State Advertisements (LSA)
- Each LSA contains the following information
  - Originating Router
  - List of directly connected neighbors of that node with the cost of the link to each one
  - Sequence Number
    - Incremented each time sending new link information
  - Link State Age
    - Packet expires when a threshold is reached,
- Each LSA “flooded” throughout network.
- Each router can put together entire topology when it receives LSAs originating from all routers

OSPF Flooding Operation

- Node X Receives LSA from Node Y
  - With Sequence Number q
  - Looks for entry with same origin/link ID
- Cases
  - No entry present
    - Add entry, propagate to all neighbors other than Y
  - Entry present with sequence number $p < q$
    - Update entry, propagate to all neighbors other than Y
  - Entry present with sequence number $p > q$
    - Send entry back to Y
    - To tell Y that it has out-of-date information
  - Entry present with sequence number $p = q$
    - Ignore it

Example LSA propagation



C does not propagate LSA from B since it has already heard an LSA with that sequence number from originating router E.

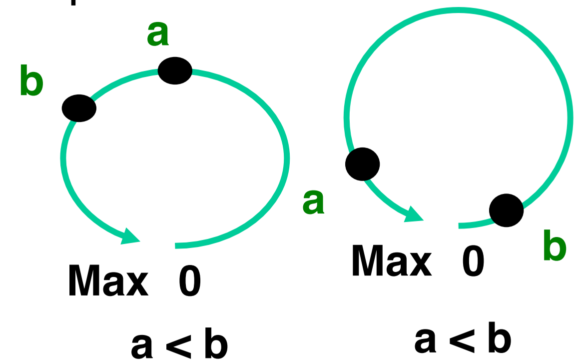
B does not propagate LSA from C since it has already heard an LSA with that sequence number from originating router E.

Flooding Issues

- When Should it be Performed
  - Periodically
  - When status of link changes
    - Detected by connected node
- What Happens when Router Goes Down & Back Up
  - Sequence number reset to 0
    - Other routers may have entries with higher sequence numbers
  - Router will send out LSAs with number 0
  - Will get back LSAs with last valid sequence number p
  - Router sets sequence number to p+1 & resends

Flooding Issues (Cont.)

- What if Sequence Number Wraps Around
  - OSPF V1:
    - Restrict LSAs to same semi-circle by regulating generation
    - But difficult to enforce with data corruption

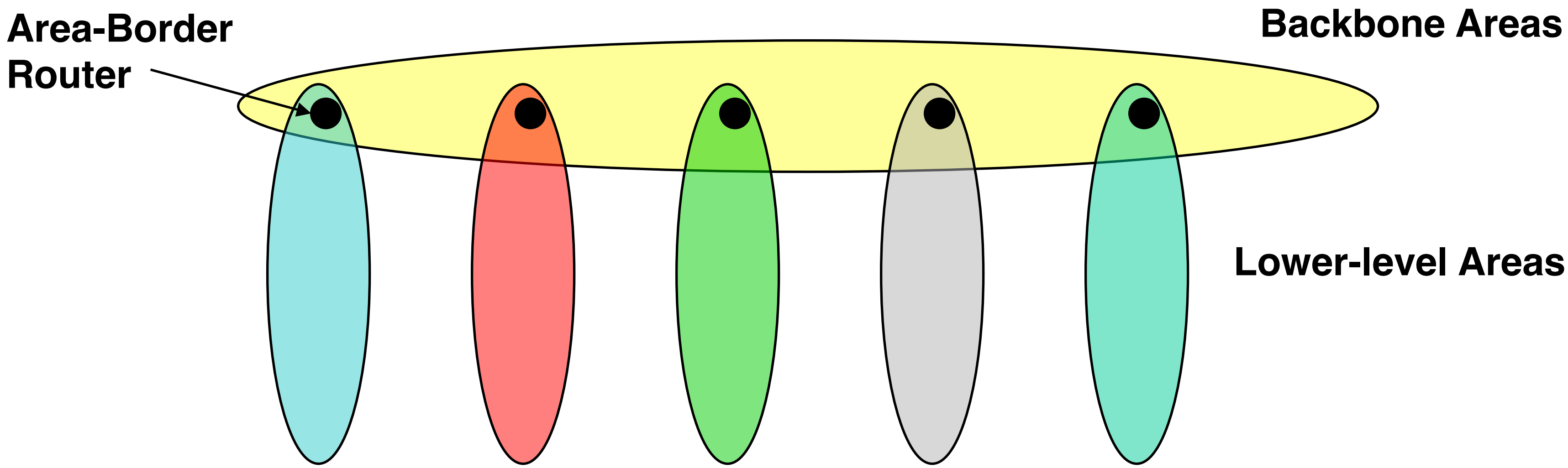


  - OSPF V2:
    - Linear rather than circular space
    - Once sequence number reaches maximum, reset count to min
      - Flush out old sequence number by advertising LSA with MAXAGE
    - With 32-bit counter, doesn't happen very often

OSPF: Advanced features

- Load balancing
  - Allow multiple paths between source and destination
- More complicated metrics beyond delay
  - Congestion, Link utilization, Bandwidth, etc.

OSPF Routing Hierarchy



- Partition Network into “Areas”
  - Router maintains link states of nodes within its area
  - Nodes in lower-level area use area-border router as default router
  - Backbone nodes can “summarize” routes within area

Link State Pros & Cons

- Advantages
  - Rapidly adapts to changes in network; Quicker convergence
    - No count to infinity problems
  - Can use more sophisticated link costs and routing algorithms (not just shortest path)
  - Can incorporate multiple paths
- Disadvantages
  - Grown to have lots of features
    - Sources of complexity & bugs
  - Configuring weights to control link utilizations not easy