ECE 356/COMPSI 356
Computer Network Architecture

Link State Routing

Wednesday October 2nd, 2019

Duke UNIVERSITY

Recap

- Last lecture: distance vector routing
- Readings for this lecture: **PD 3.3.3**

2

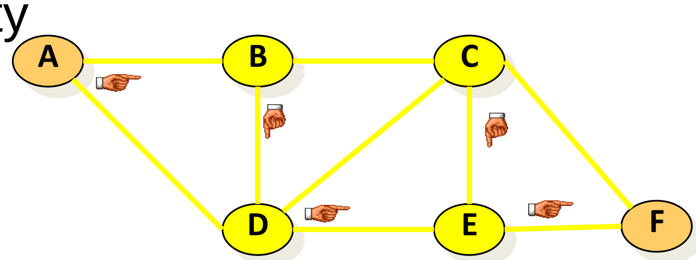
Duke UNIVERSITY

Lecture Outline

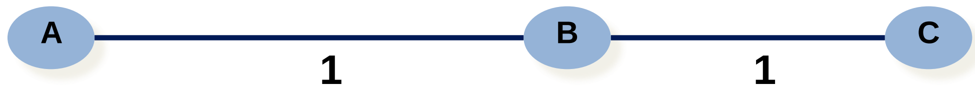
- Link-state routing
- Reliable flooding
- Algorithm: Dijkstra
- Protocol: Open Shortest Path First (OSPF)

Distance Vector vs. Link State Routing

- DV only sees next hop “direction”
- Wrong directions lead to wrong routes
  - ➢Count to infinity

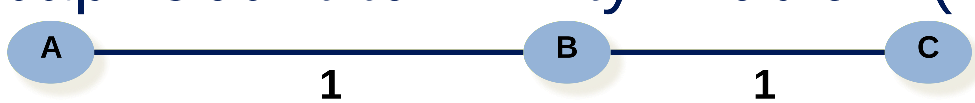


Recap: Count-to-Infinity Problem (1/2)



A's Routing Table					B's Routing Table		
to	via (next hop)	cost			to	via (next hop)	cost
C	B	2			C	C	1
			now link B-C goes down				
C	B	2			C	-	∞
			C 2 →	← C ∞			
C	-	∞			C	A	3
			C ∞ →	← C 3			

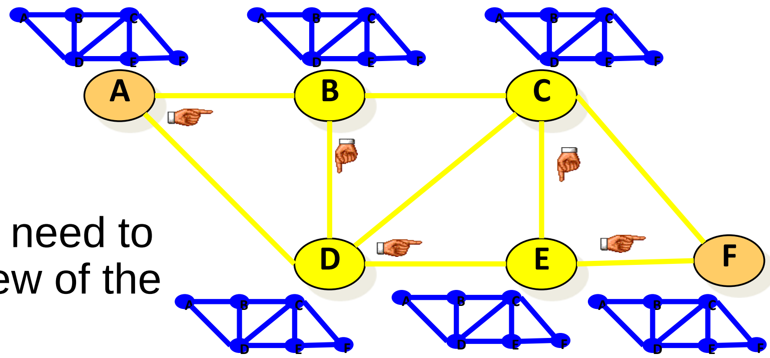
Recap: Count-to-Infinity Problem (2/2)



A's Routing Table					B's Routing Table		
C	B	2			C	-	∞
			C 2 →	← C ∞			
C	-	∞			C	A	3
			C ∞ →	← C 3			
C	B	4			C	-	∞
			C 4 →	← C ∞			

Distance Vector vs. Link State Routing

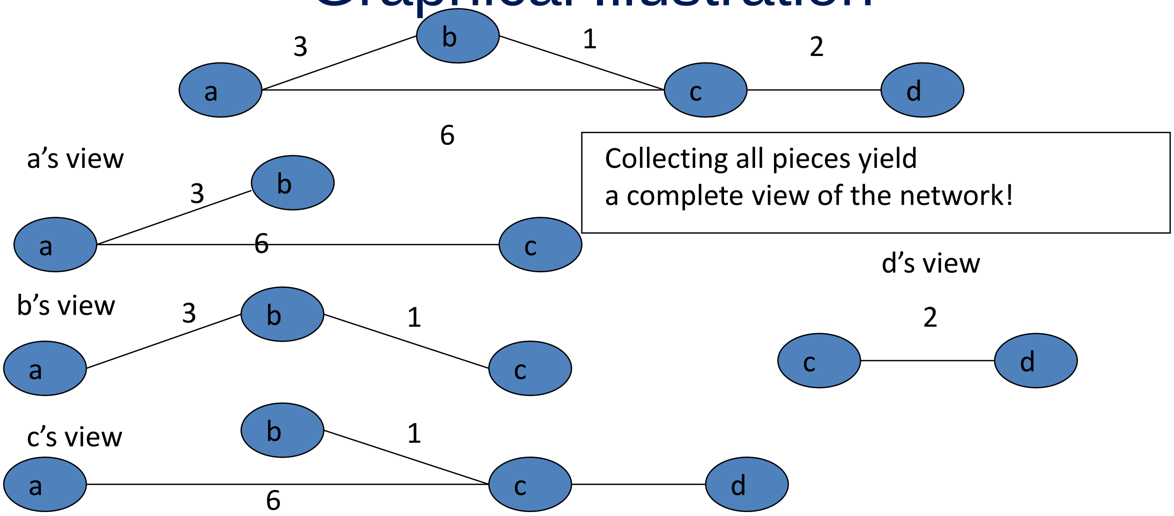
- In link state routing, each node has a complete map of the topology
- If a node fails, each node can calculate the new route
- Challenge: all nodes need to have a consistent view of the network



Link State Routing: Basic Operations

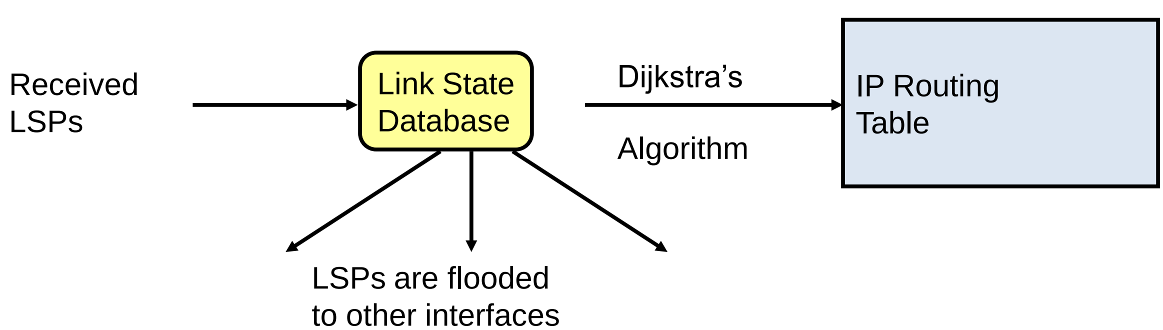
1. Each router establishes *link adjacency*
2. Each router generates a *link state advertisement (LSA), and* floods it to the network
3. Each router maintains a database of all received LSAs
   - *Topological database* or *link state database*
4. Each router runs the Dijkstra's algorithm

Link State Routing: Graphical Illustration



9

Operation of a Link State Routing Protocol



Lecture Outline

- Link-state routing
- **Reliable flooding**
- Algorithm: Dijkstra
- Protocol: Open shortest path first (OSPF)

Reliable Flooding

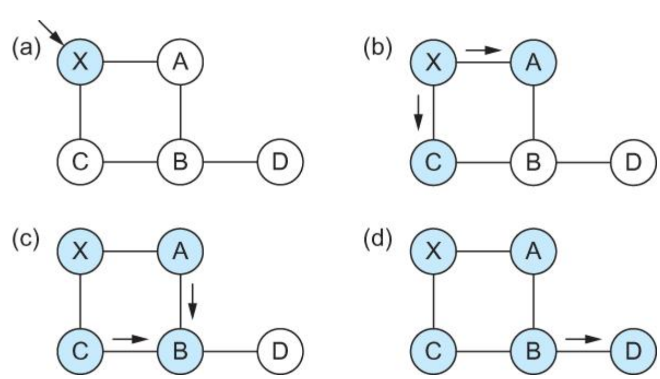
- We've learned a flooding algorithm used by Ethernet switches
- Question: why is it insufficient for link-state routing?
  - ➢ Lost LSAs may result in inconsistent topologies at different routers
  - ➢ Inconsistent topologies may lead to routing loops

Reliable Flooding

- LSPs are transmitted reliably between adjacent routers
  - ACK and retransmission
- For a node x, if it receives an LSA sent by y
  - Stores LSA(y) if it does not have a copy
  - Otherwise, compares SeqNo. If newer, store; otherwise discard
  - If the LSA(y) is new, floods LSA(y) to all neighbors except the incoming neighbor

Duke UNIVERSITY

An Example of Reliable Flooding



Reliable delivery on each of the links

Duke UNIVERSITY

When to Flood an LSP

- Triggered if a link's state has changed
  - ➢ Detecting failure
    - Neighbors exchange hello messages
    - If not receiving hello, assume dead
- Periodic generating a new LSA
  - ➢ Fault tolerance (what if LSA in memory is corrupted?)

Duke UNIVERSITY

Lecture Outline

- Link-state routing
- Reliable flooding
- **Algorithm: Dijkstra**
- Protocol: Open shortest path first (OSPF)

Duke UNIVERSITY

Path Computation (1/2)

Dijkstra's Shortest Path Algorithm for a Graph

Input: Graph **(N,E)** with

N the set of nodes and **E** the set of edges

c_{vw} link cost ($c_{vw} = \infty$ if $(v,w) \notin E$, $c_{vv} = 0$)

s source node.

Path Computation (2/2)

Output: D_n cost of the least-cost path from node s to node n

```
M = {s};
for each  n ∉ M
   Dn  = csn;
while (M ≠ all nodes) do
   Find w ∉ M for which Dw = min{Dj ; j ∉ M};
   Add w to M;
   for each neighbor n of w and n ∉ M
      Dn = min[ Dn, Dw + cwn ];
      Update route;
enddo
```

Practical Implementation: Forward Search Algorithm (1/2)

- More efficient: extracting min from a smaller set rather than the entire graph
- Two lists: Confirmed and Tentative
- Each entry: (destination, cost, nextHop)

1. Confirmed = {(s,0,s)}
2. Next ← Confirmed.last

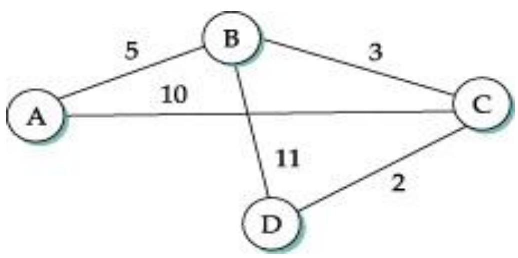
Duke UNIVERSITY

Practical Implementation: Forward Search Algorithm (2/2)

3. For each Nbr of Next
   - ➢ Cost ← myself to Next + Next to Nbr
     - If Neighbor not in Confirmed or Tentative
       - – Add (Nbr, Cost, my.Nexthop(Next)) to Tentative
   - ➢ If Nbr is in Tentative, and Cost is less than Nbr.Cost, update Nbr.Cost to Cost
4. If Tentative not empty, pick the entry with smallest cost in Tentative and move it to Confirmed, and return to Step 2
   - ➢ Pick the smallest cost from a smaller list Tentative, rather than the rest of the graph

Duke UNIVERSITY

Forward Search: An Example



Step	Confirmed	Tentative
1	(D,0,-)	
2		
3		
4		
5		
6		
7		

Step	Confirmed	Tentative
1	(D,0,-)	
2	(D,0,-)	(B,11,B), (C,2,C)
3	(D,0,-), (C,2,C)	(B,11,B)
4	(D,0,-), (C,2,C)	(B,5,C) (A,12,C)
5	(D,0,-), (C,2,C), (B,5,C)	(A,12,C)
6	(D,0,-),(C,2,C),(B,5,C)	(A,10,C)
7	(D,0,-),(C,2,C),(B,5,C), (A,10,C)	

Lecture Outline

- Link-state routing
- Reliable flooding
- Algorithm: Dijkstra
- **Protocol: Open Shortest Path First (OSPF)**

OSPF

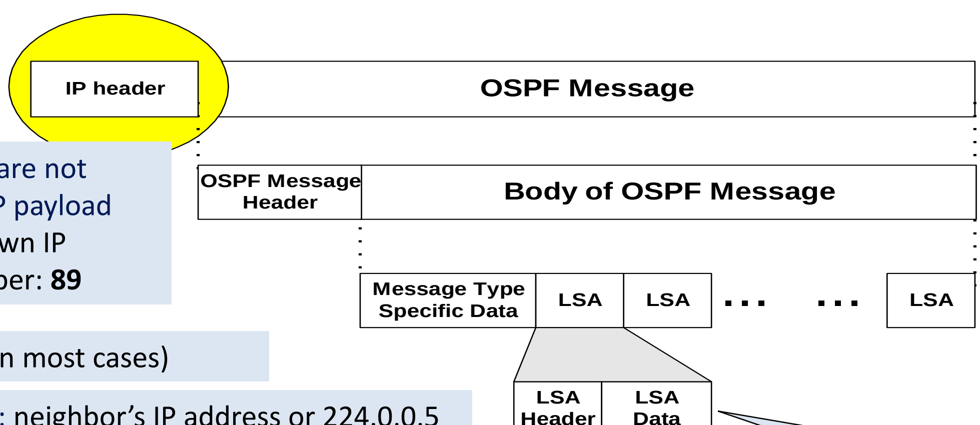
- OSPF = Open Shortest Path First
  - “Open” stands for non-proprietary
- A link state routing protocol
- The complexity of OSPF is significant
  - RIP (RFC 2453 ~ 40 pages)
  - OSPF (RFC 2328 ~ 250 pages)

Features of OSPF

- Provides <u>authentication</u> of routing messages
  - Otherwise routing messages can be spoofed, with disastrous consequences
  - Blackhole attack, wormhole attack
- Allows hierarchical routing
  - Divide a domain into *areas*
- Enables load balancing by allowing traffic to be split evenly across routes with equal cost

Duke UNIVERSITY

OSPF Packet Format



OSPF packets are not carried as UDP payload
OSPF has its own IP protocol number: **89**

TTL: set to 1 (in most cases)

Destination IP: neighbor's IP address or 224.0.0.5 (ALLSPFRouters) or 224.0.0.6 (AllDRouters)

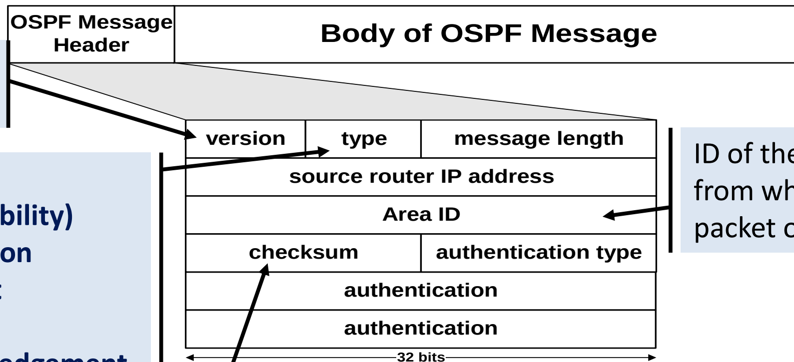
Link state advertisement

Duke UNIVERSITY

OSPF Common Header (1/2)

2: current version is OSPF V2

Message types:
1: Hello (tests reachability)
2: Database description
3: Link status request
4: Link state update
5: Link state acknowledgement

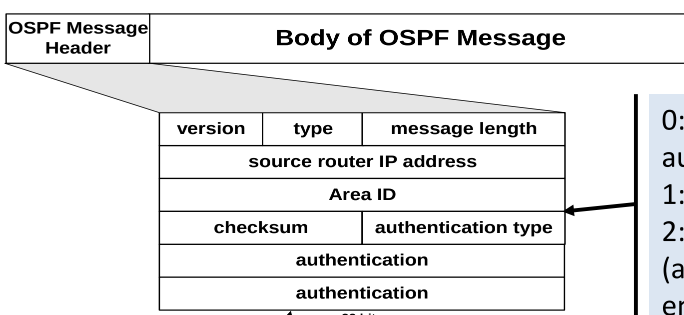


ID of the area from which the packet originated

Standard IP checksum taken over entire packet

Duke UNIVERSITY

OSPF Common Header (2/2)

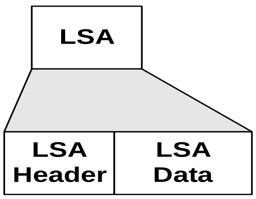


0: No authentication
1: Simple password
2: MD5 checksum (added to to the end of the packet)

Authentication-related information
Nondecreasing sequence number to prevent replay attacks

Duke UNIVERSITY

OSPF LSA Format



- LSAs
  - Type 1: cost of links between routers
  - Type 2: networks to which the router connects
  - Others: hierarchical routing

LSA Header

Link Age	Link Type
Link State ID	
advertising router	
link state sequence number	
checksum	length

Link 1

Link ID		
Link Data		
Link Type	#TOS metrics	Metric

Link 2

Link ID		
Link Data		
Link Type	#TOS metrics	Metric

Type 1 LSA: Cost of Links Between Routers

Link ID		
Link Data		
Link Type	#TOS metrics	Metric

- Link ID: router ID at the other end of the link
- Link Data: identify parallel links
- Metric: cost of the link
- Type: types of the link e.g., point-to-point

Open Question

- How to set link metrics?
- Design choice 1: all to 1
- Design choice 2: based on load
  - Problems?
- In practice: static

Duke UNIVERSITY

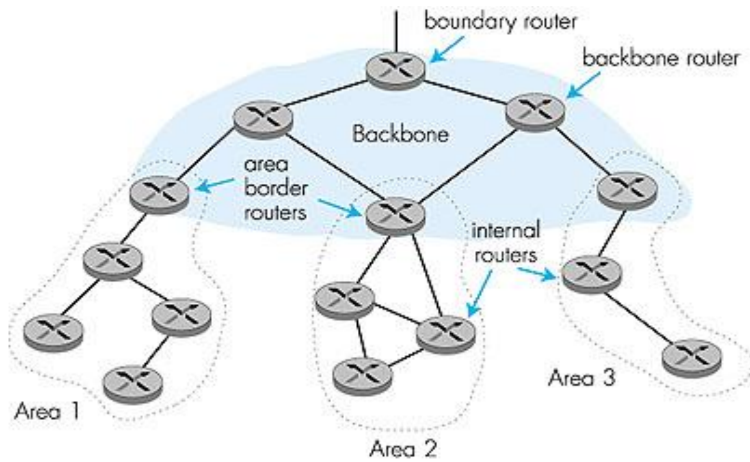
OSPF Metric Example

Bandwidth	OSPF Cost
100 Gbps	1
40 Gbps	1
10 Gbps	1
1 Gbps	1
100 Mbps	1
10 Mbps	10
1.544 Mbps	64
768 Kbps	133
384 Kbps	266
128 Kbps	781

32

Duke UNIVERSITY

Hierarchical OSPF



Duke UNIVERSITY

Hierarchical OSPF

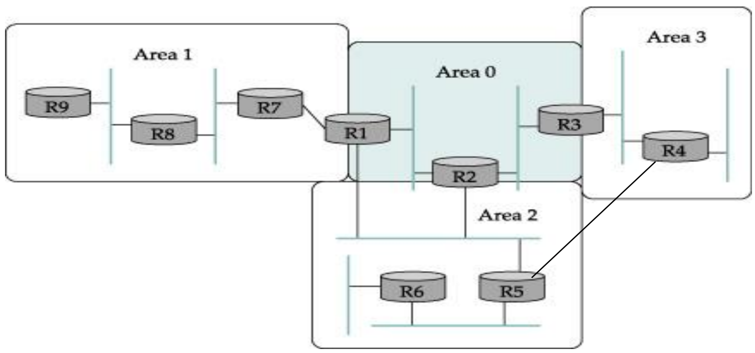
- **Two-level hierarchy**: local area, backbone
  - Link-state advertisements only in area
  - Each nodes has detailed area topology; only know direction (shortest path) to nets in other areas
- **Area border routers**: "summarize" distances to nets in own area, advertise to other area border routers
- **Backbone routers:** run OSPF routing limited to backbone

34

Duke UNIVERSITY

Scalability and Optimal Routing

- A frequent tradeoff in network design
- Hierarchy introduces information hiding



Lecture Outline

- Link-state routing
- Reliable flooding
- Algorithm: Dijkstra
- Protocol: Open Shortest Path First (OSPF)

Next Lecture

- Inter-domain routing (BGP)

Duke UNIVERSITY