CS ECE 4457

Computer Networks:
Architecture and Protocols

Lecture 8
Switched Ethernet
Spanning Tree Protocol

Qizhe Cai

Goals for Today's Lecture

- **"Why" has Ethernet evolved to switched Ethernet?**
- **Experience (the beauty of) Spanning Tree Protocol**
- **Why** do we need network layer?
    - **Why** not just use switched Ethernet across the Internet?

Recap: Link layer

- **Traditional Link Layer: Broadcast Ethernet**
- **CSMA/CD**
  - Random access on a broadcast channel
  - Exponential Backoff
- **Why Frames?**
  - Data link layer **interfaces** with **physical layer** using **frames**
  - To incorporate sentinel bits for identifying frame start/end
  - To incorporate link layer source and destination names
  - To incorporate CRC for checking correctness of received frames
- **Modern Link Layer: Switched Ethernet**
  - Understanding switched Ethernet is the goal of today's lecture

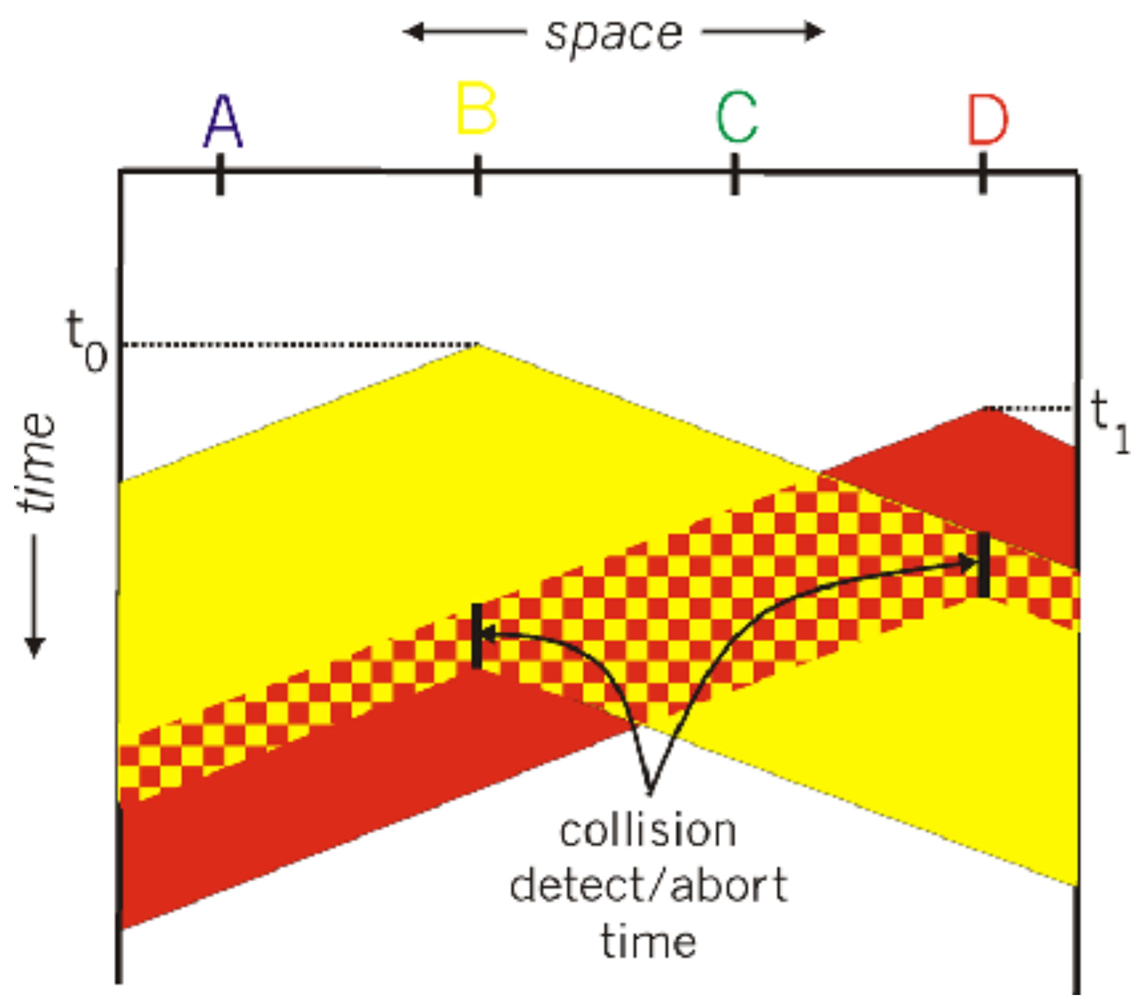
Questions?

WHY Switched Ethernet?

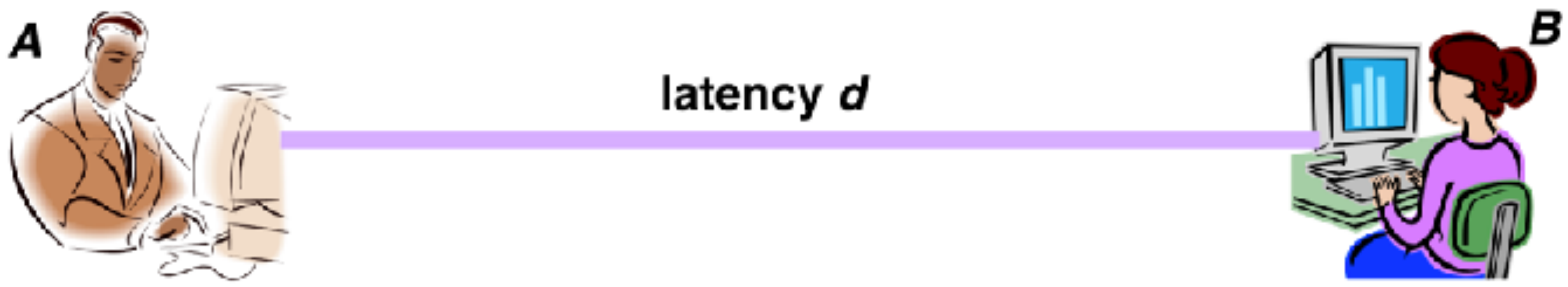
Collision Detection limits Ethernet scalability

- B and D can tell that collision occurred

- However, need restrictions on
    - **Minimum frame size**
    - **Maximum distance**

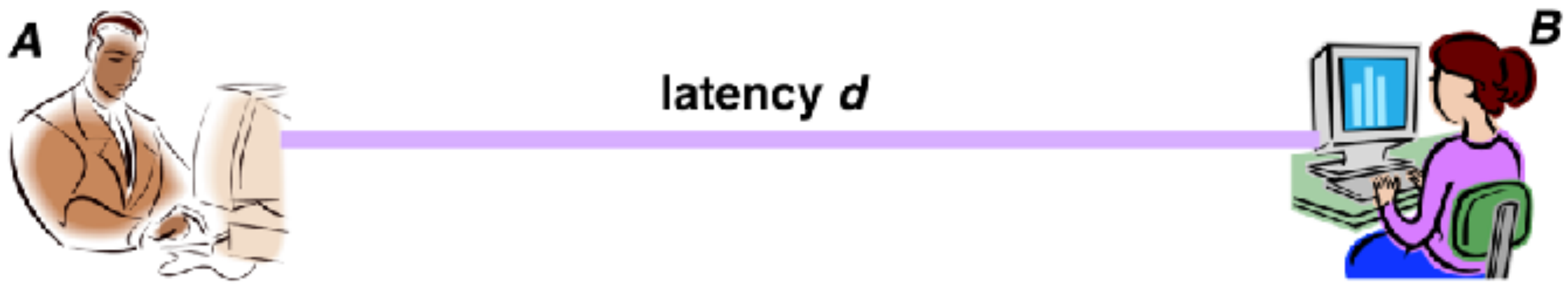


Limits on Traditional Ethernet Scalability



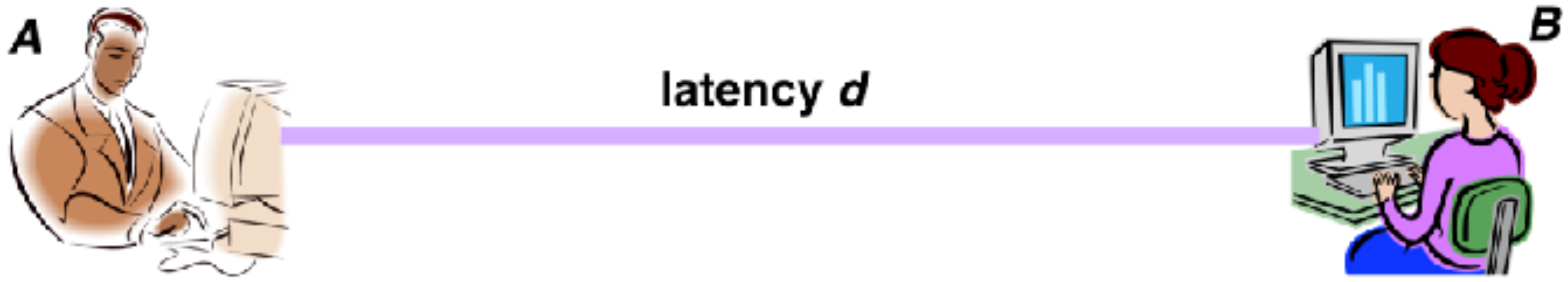
- **Latency depends on physical length of link**
    - Propagation delay
- **Suppose A sends a packet at time 0**
    - B sees an idle line at all times before **d**
    - … so B happily starts transmitting a packet
- **B detects a collision at time d**
    - But A can’t see collision until **2d**
    - **A must have a frame size such that transmission time > 2d**
    - **Need transmission time > 2 * propagation delay**

Limits on Traditional Ethernet Scalability



- **Transmission time > 2 * propagation delay**
- **Requires either very large frames (underutilization) or small scale.**
    - **Example: consider 100 Mbps Ethernet**
    - **Suppose** minimum frame length: 512 bits (64 bytes)
        - Transmission time = 5.12 μsec
        - Thus, propagation delay < 2.56 μsec
        - Length < 2.56 μsec * speed of light
        - Length < 768m
- **Cannot scale beyond ~76.8m for 1Gbps and beyond ~7.68m for 10Gbps**

Limits on Traditional Ethernet Scalability



- Transmission time > 2 * propagation delay
- Cannot scale beyond ~76.8m for 1Gbps and beyond ~7.68m for 10Gbps
- This is WHY modern Ethernet networks are “switched”

Evolution

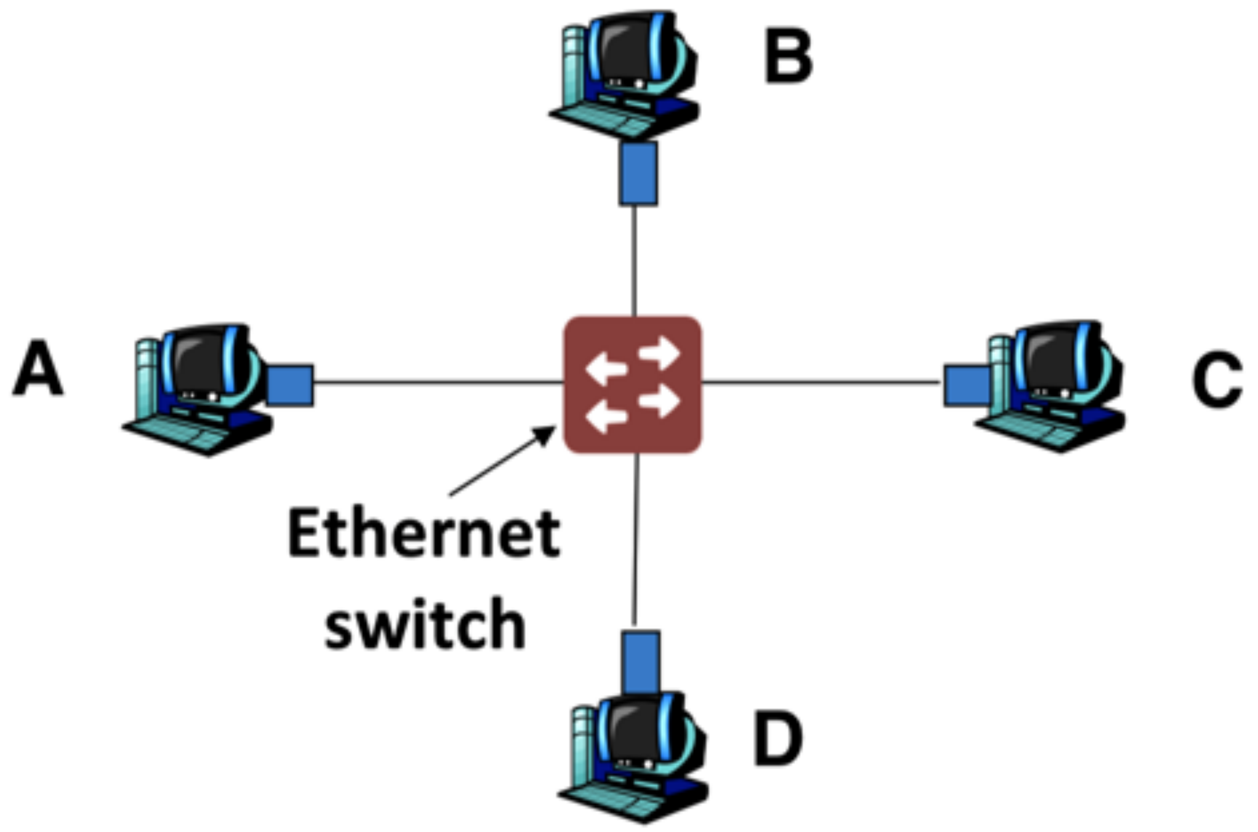
- **Ethernet was invented as a broadcast technology**
    - Hosts share channel
    - Each packet received by all attached hosts
    - CSMA/CD for access control

- **Current Ethernets are “switched”**
    - Point-to-point medium between switches;
    - Point-to-point medium between each host and switch
    - Sharing only when needed (using CSMA/CD)

Questions?

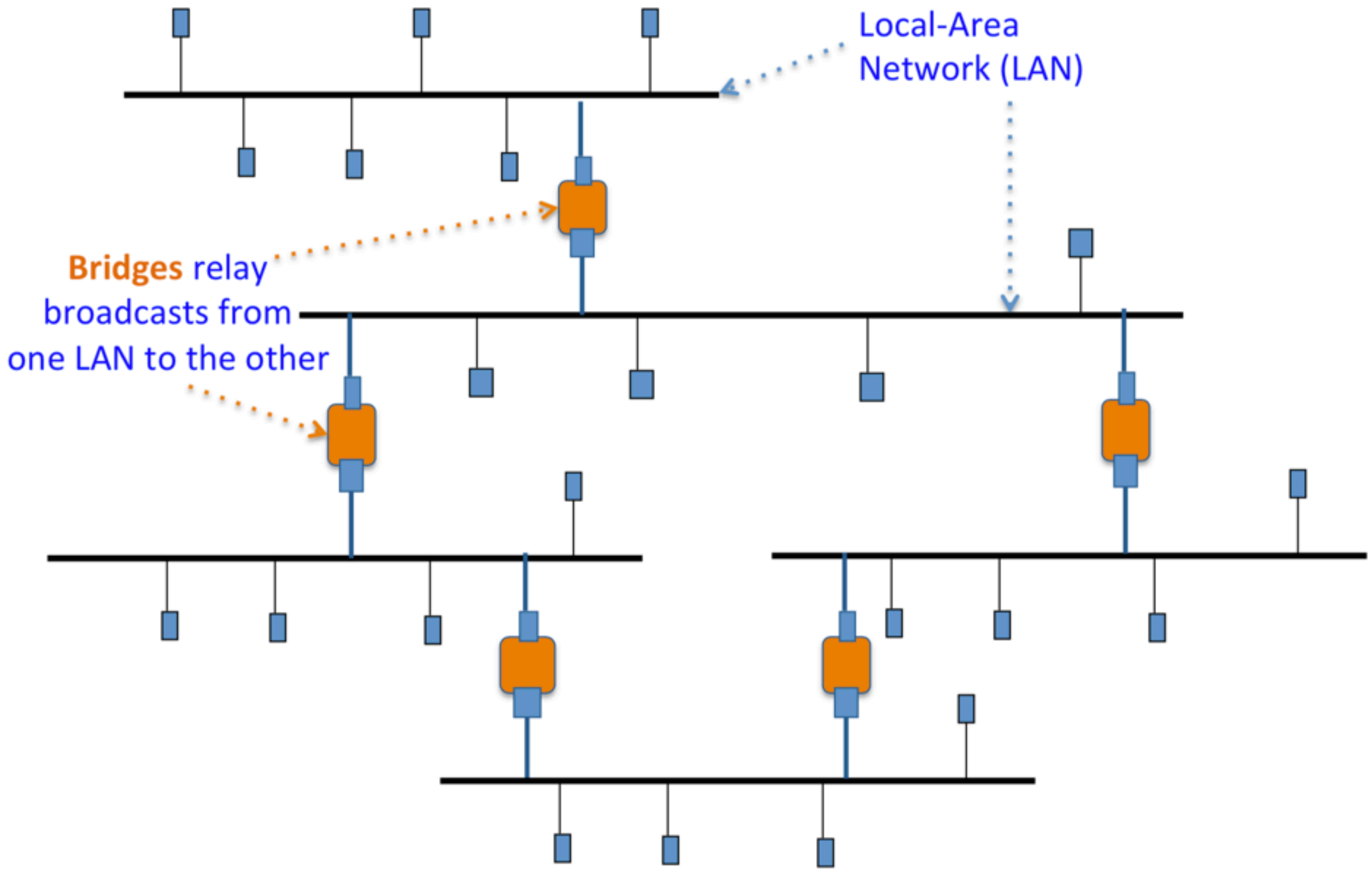
Switched Ethernet

Switched Ethernet

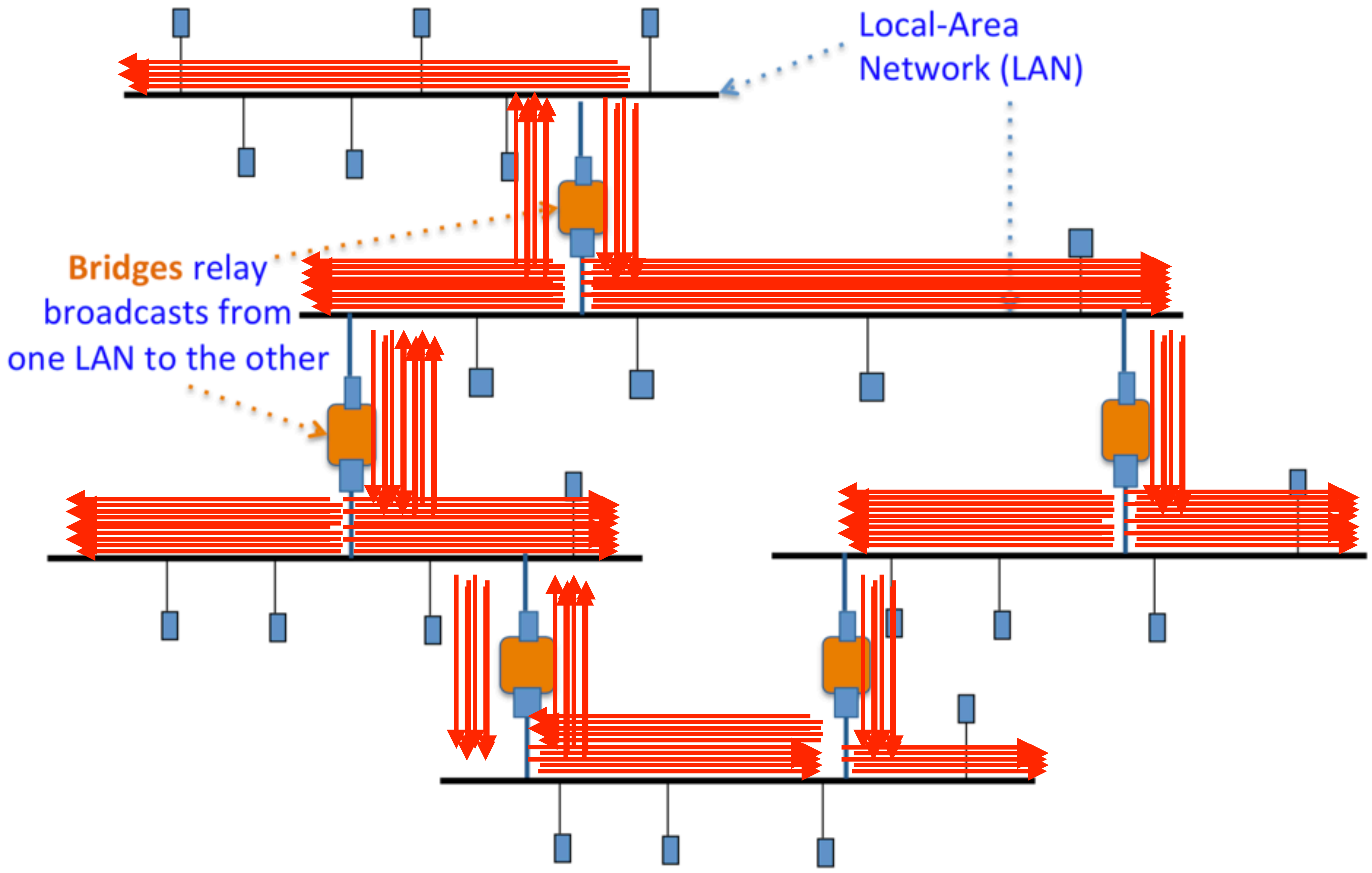


- Enables concurrent communication
    - Host A can talk to C, while B talks to D
    - No collisions -> no need for CSMA, CD
    - No constraints on link lengths or frame size

Routing in Switched Ethernet (Extended LANs)



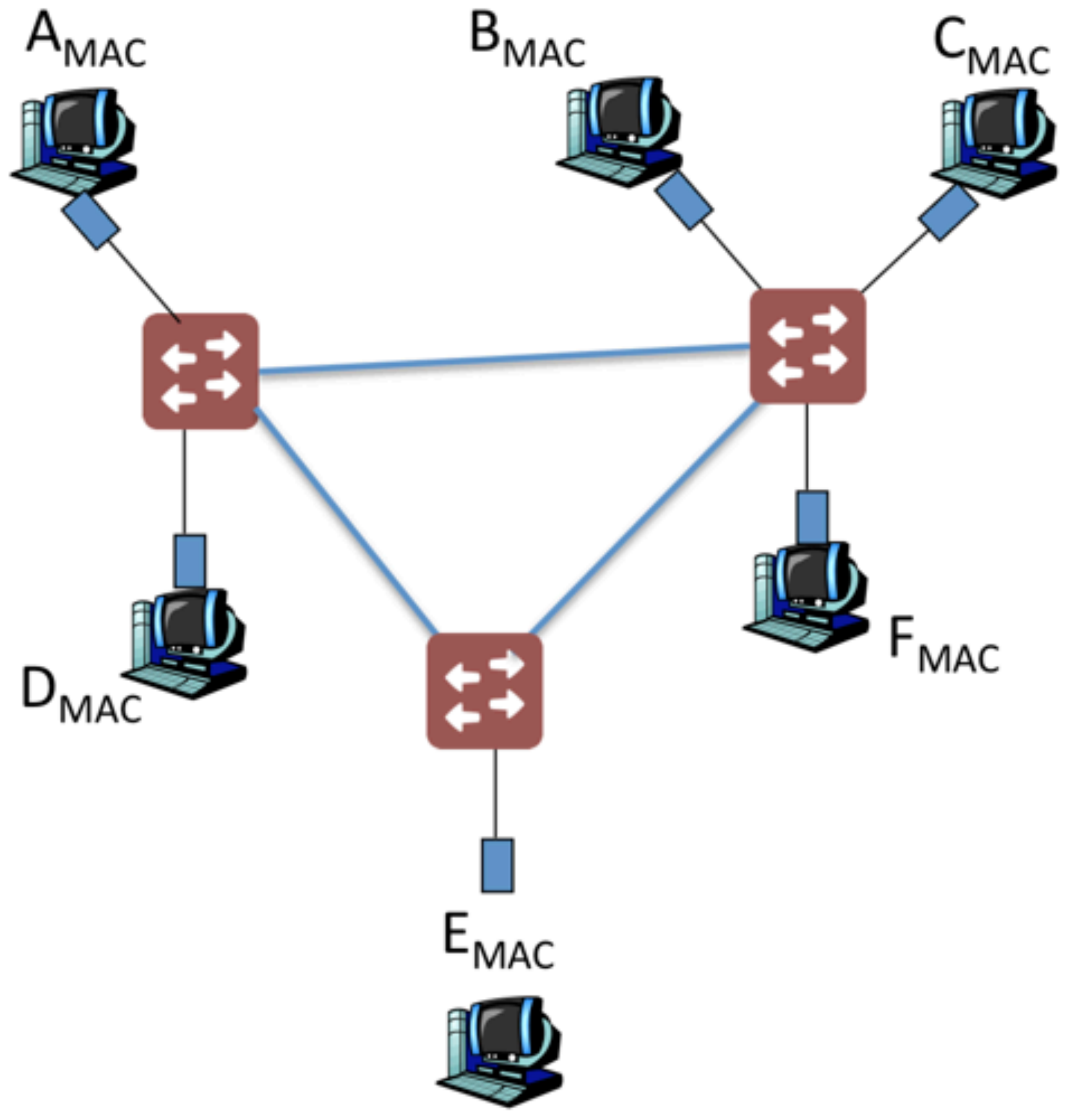
Naïvely Routing in "Extended LANs": Broadcast storm



How to avoid the Broadcast Storm Problem?

Get rid of the loops!

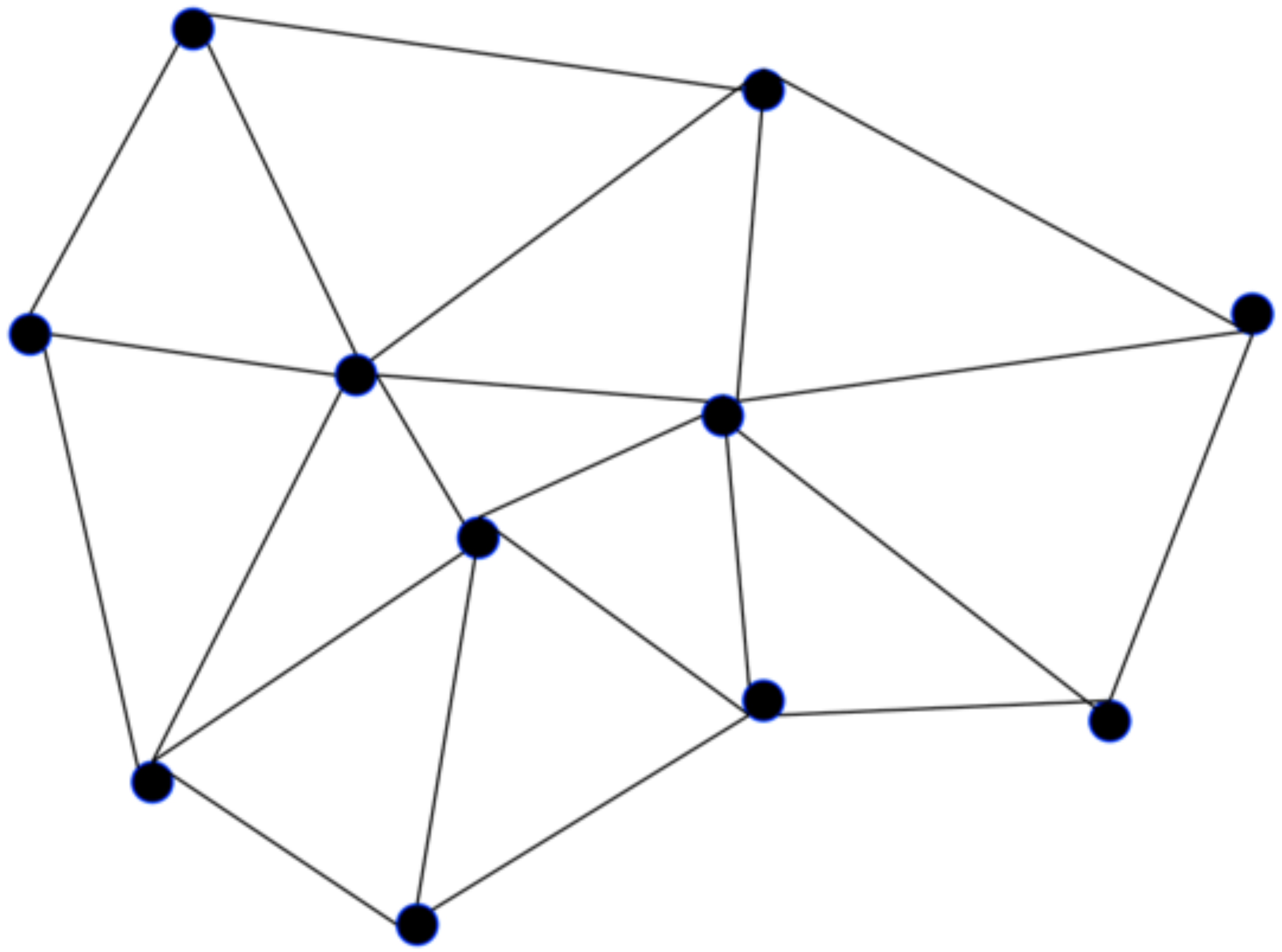
Lets get back to the graph representation!



Easiest Way to Avoid Loops

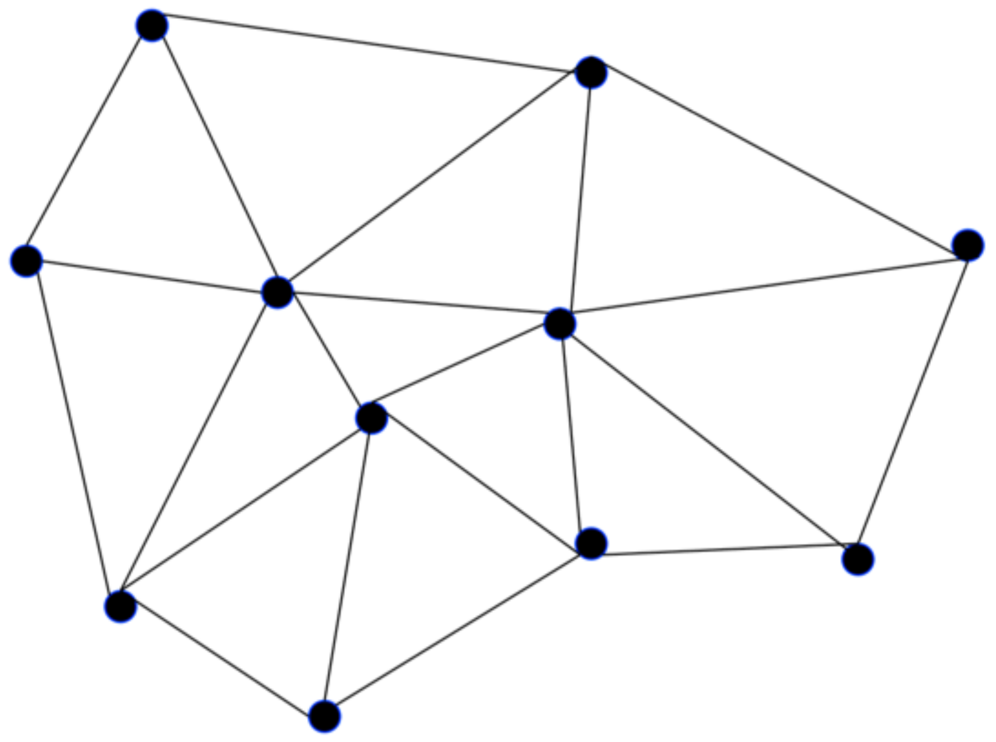
- Use a network topology (graph) where loop is impossible!
- Take arbitrary topology (graph)
- **Build spanning tree**
  - **Subgraph that includes all vertices but contains no cycles**
  - Links not in the spanning tree are not used in forwarding frames
- Only one path to destinations on spanning trees
  - So don't have to worry about loops!

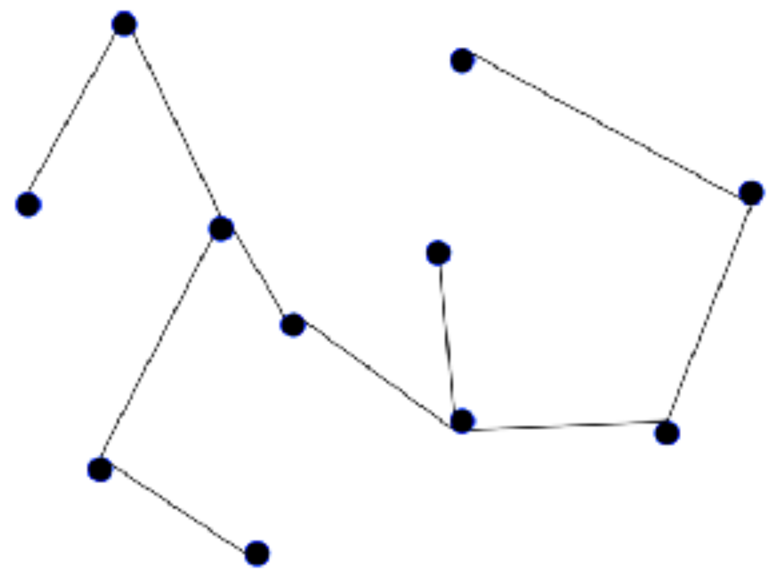
Consider Graph

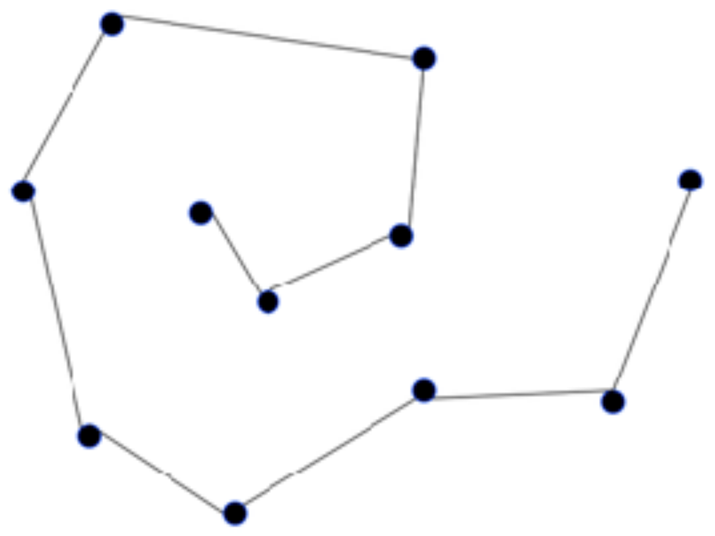


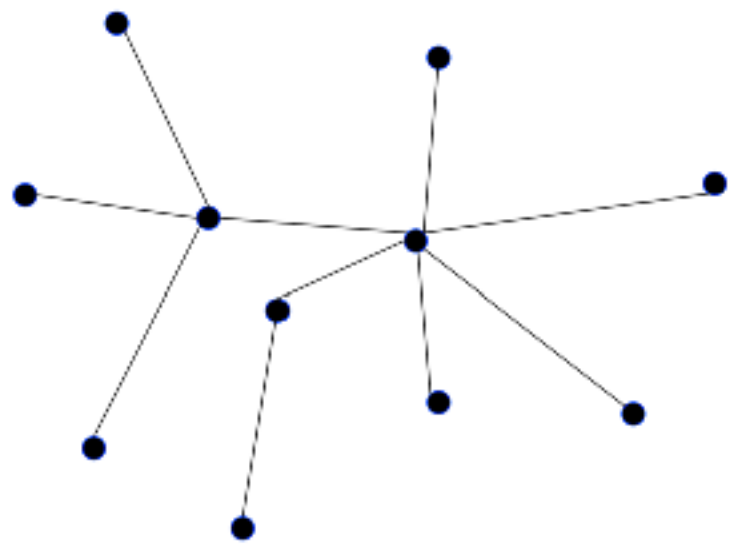
Multiple Spanning Trees

Subgraph that includes all vertices but contains no cycles









Questions?

Spanning Tree Approach

- Take arbitrary topology
- Pick subset of links that form a spanning tree
- Only forward packets on the spanning tree
  - => No loops
  - => No broadcast storm

Spanning Tree Protocol

- Protocol by which bridges construct a spanning tree
- Nice properties
    - Zero configuration (by operators or users)
    - Self healing
- Still used today
- Constraints for backwards compatibility
    - No changes to end-hosts
    - Maintain plug-n-play aspect
- Earlier Ethernet achieved plug-n-play by leveraging a broadcast medium
    - Can we do the same for a switched topology?

Algorithm has Two Aspects…

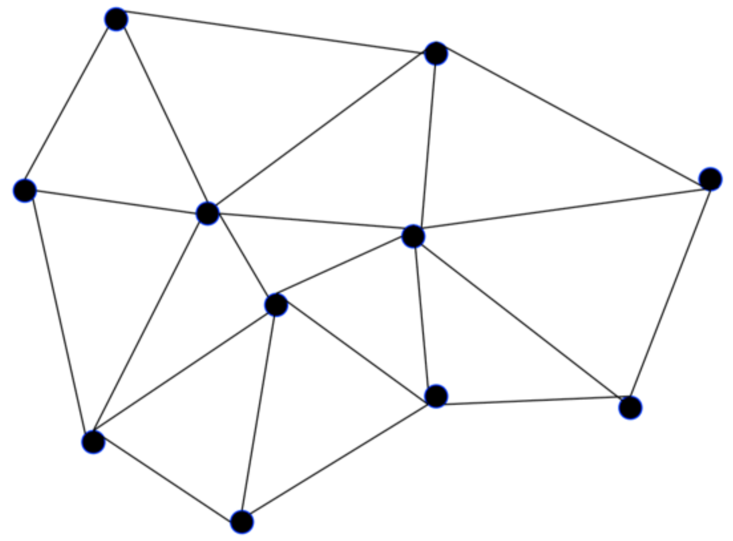
- Pick a root:
    - Destination to which the shortest paths go
    - Pick the one with the smallest identifier (MAC name/address)
- For each switches/routers, Compute the shortest paths to the root
    - No shortest path can have a cycle
    - Only keep the links on the shortest path
    - Break ties in some way
        - so we only keep one shortest path from each node
- Ethernet's spanning tree construction does both with a single algorithm

Breaking Ties

- When there are multiple shortest paths to the root:
  - Choose the path via neighbor switch with the smallest identifier

- **One could use any tie breaking system**
  - This is just an easy one to remember and implement

Constructing a Spanning Tree



- **Messages (Y,d,X)**
  - Proposing Y as the root
  - From node X
  - And advertising a distance d between X and Y
- Switches elect the node with smallest identifier (MAC address) as root
  - **Y** in messages
- Each switch determines if a link is on its shortest path to the root
  - If not, excludes it from the tree
  - **d** to **Y** in the message is used to determine this

Steps in Spanning Tree Protocol

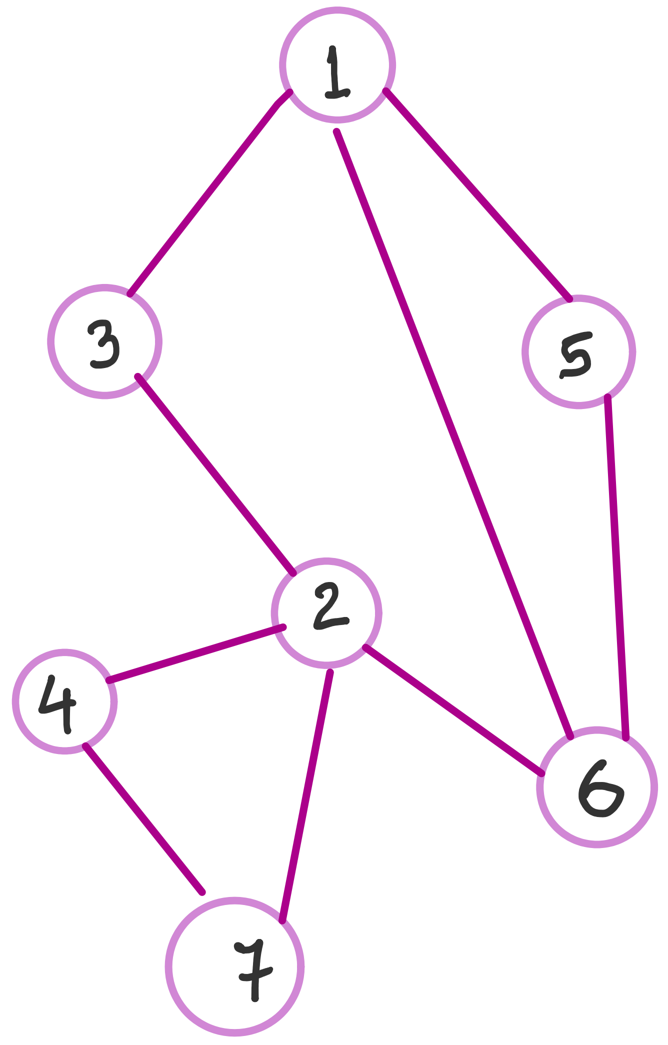
- **Messages (Y,d,X)**
    - Proposing root Y; from node X; advertising a distance d to Y
- Initially each switch proposes itself as the root
    - that is, switch X announces (X,0,X) to its neighbors
- At each switch Z:

  WHENEVER a message (Y,d,X) is received from X:
    - IF Y's id < current root
        - THEN set root = Y; next-hop = X
    - IF Shortest distance to root > d + distance_from_X
        - THEN set shortest-distance-to-root = d + distance_from_X
    - IF **root changed OR shortest distance to the root changed:**
        - Send all neighbors message (Y, shortest-distance-to-root, Z)

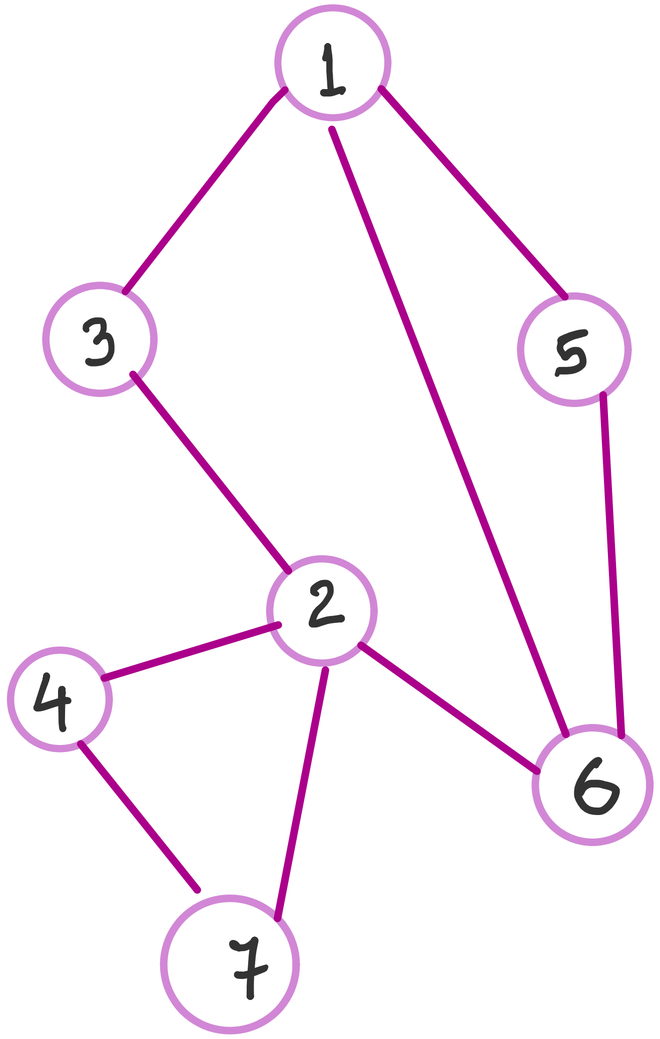
Group Exercise:

Lets run the Spanning Tree Protocol on this example

(assume all links have “distance” 1)

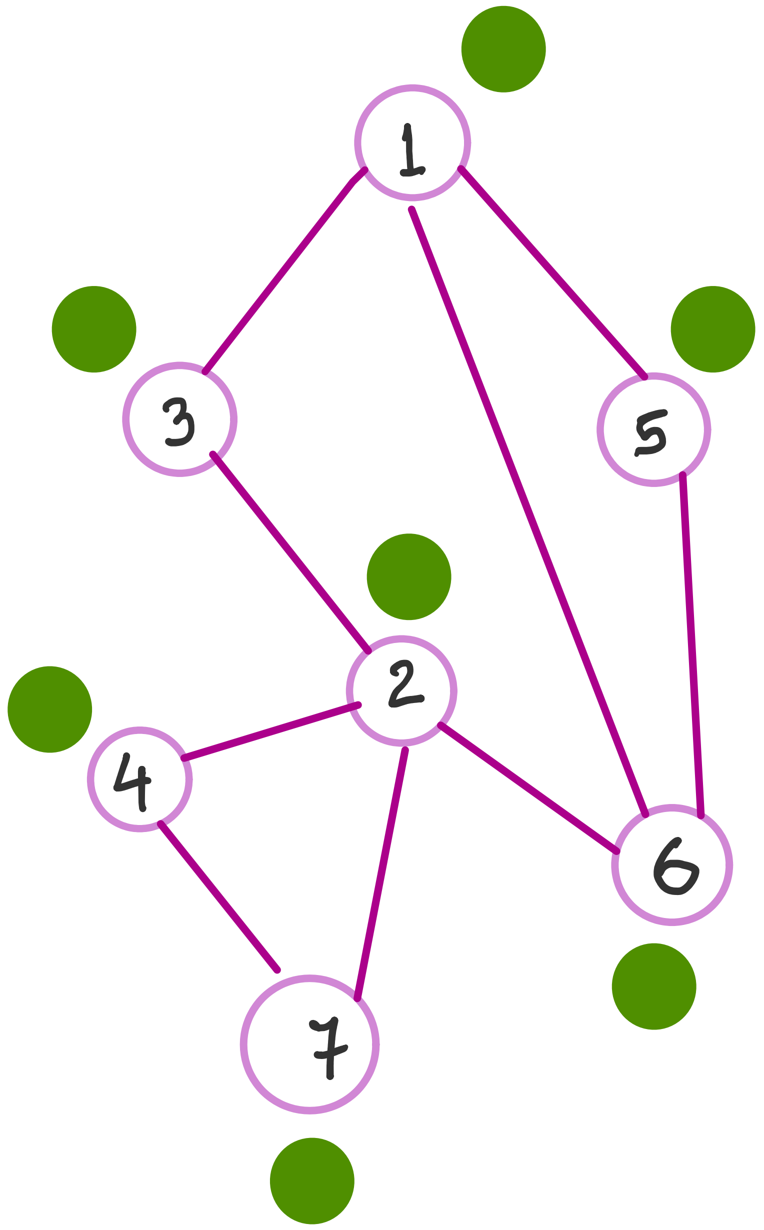


Round 1



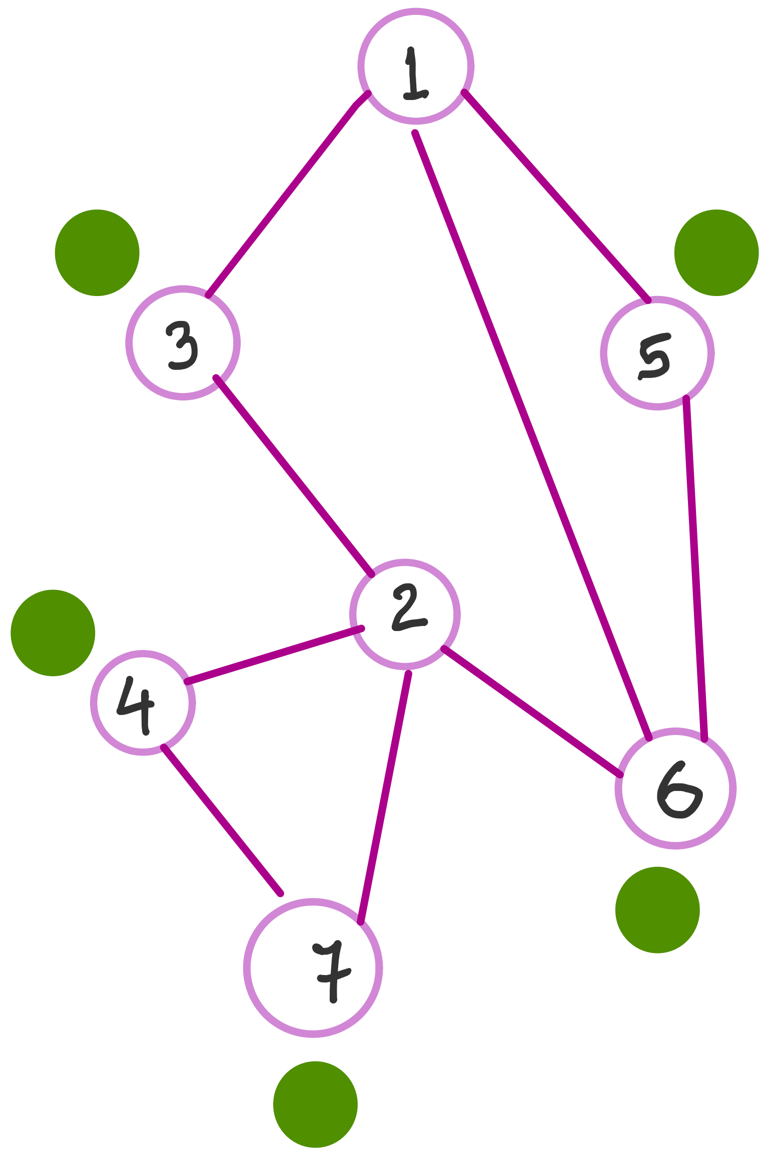
	Receive	Send	Next-hop
1		(1, 0, 1)	1
2		(2, 0, 2)	2
3		(3, 0, 3)	3
4		(4, 0, 4)	4
5		(5, 0, 5)	5
6		(6, 0, 6)	6
7		(7, 0, 7)	7

Round 2



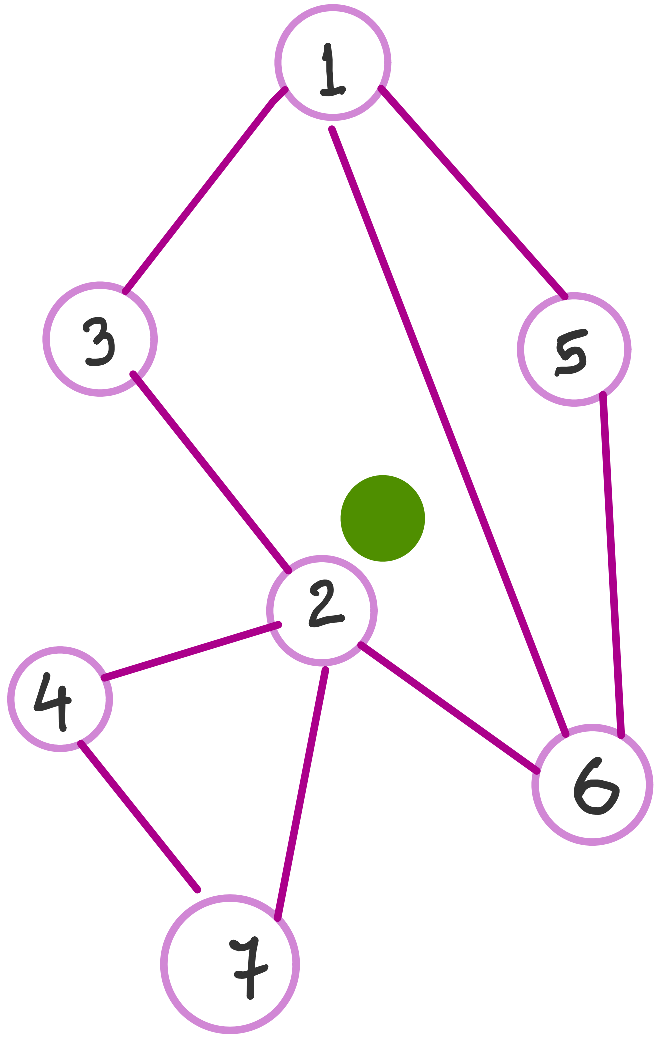
	Receive	Send	Next hop
1 (1, 0, 1)	(3, 0, 3), (5, 0, 5), (6, 0, 6)		1
2 (2, 0, 2)	(3, 0, 3), (4, 0, 4), (6, 0, 6), (7, 0, 7)		2
3 (3, 0, 3)	(1, 0, 1), (2, 0, 2)	**(1, 1, 3)**	**1**
4 (4, 0, 4)	(2, 0, 2), (7, 0, 7)	**(2, 1, 4)**	**2**
5 (5, 0, 5)	(1, 0, 1), (6, 0, 6)	**(1, 1, 5)**	**1**
6 (6, 0, 6)	(1, 0, 1), (2, 0, 2), (5, 0, 5)	**(1, 1, 6)**	**1**
7 (7, 0, 7)	(2, 0, 2), (4, 0, 4)	**(2, 1, 7)**	**2**

Round 3



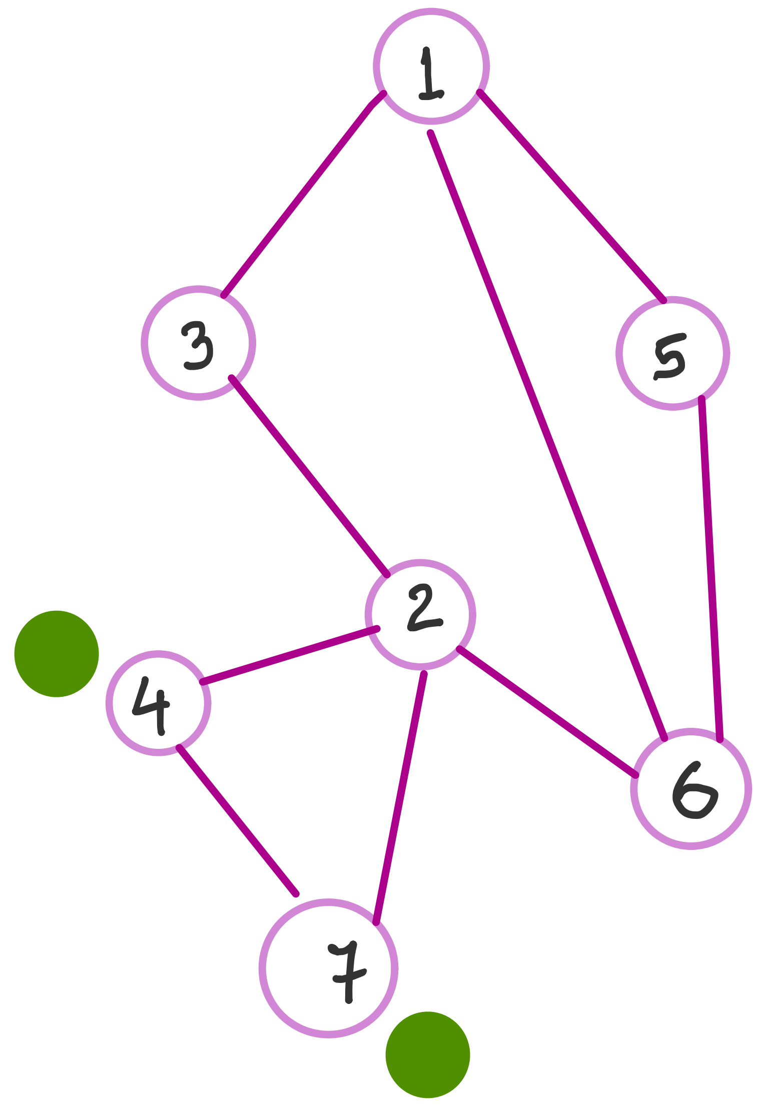
	Receive	Send	Next hop
1 (1, 0, 1)	(1, 1, 3), (1, 1, 5), (1, 1, 6)		1
2 (2, 0, 2)	(1, 1, 3), (2, 1, 4), (1, 1, 6), (2, 1, 7)	**(1, 2, 2)**	**3 (or 6)**
3 (1, 1, 3)			1
4 (2, 1, 4)	(2, 1, 7)		2
5 (1, 1, 5)	(1, 1, 6)		1
6 (1, 1, 6)	(1, 1, 5)		1
7 (2, 1, 7)	(2,1, 4)		2

Round 4



	Receive	Send	Next hop
1 (1, 0, 1)			1
2 (1, 2, 2)			3
3 (1, 1, 3)	(1, 2, 2)		1
4 (2, 1, 4)	(1, 2, 2)	**(1, 3, 4)**	**2**
5 (1, 1, 5)			1
6 (1, 1, 6)	(1, 2, 2)		1
7 (2, 1, 7)	(1, 2, 2)	**(1, 3, 7)**	**2**

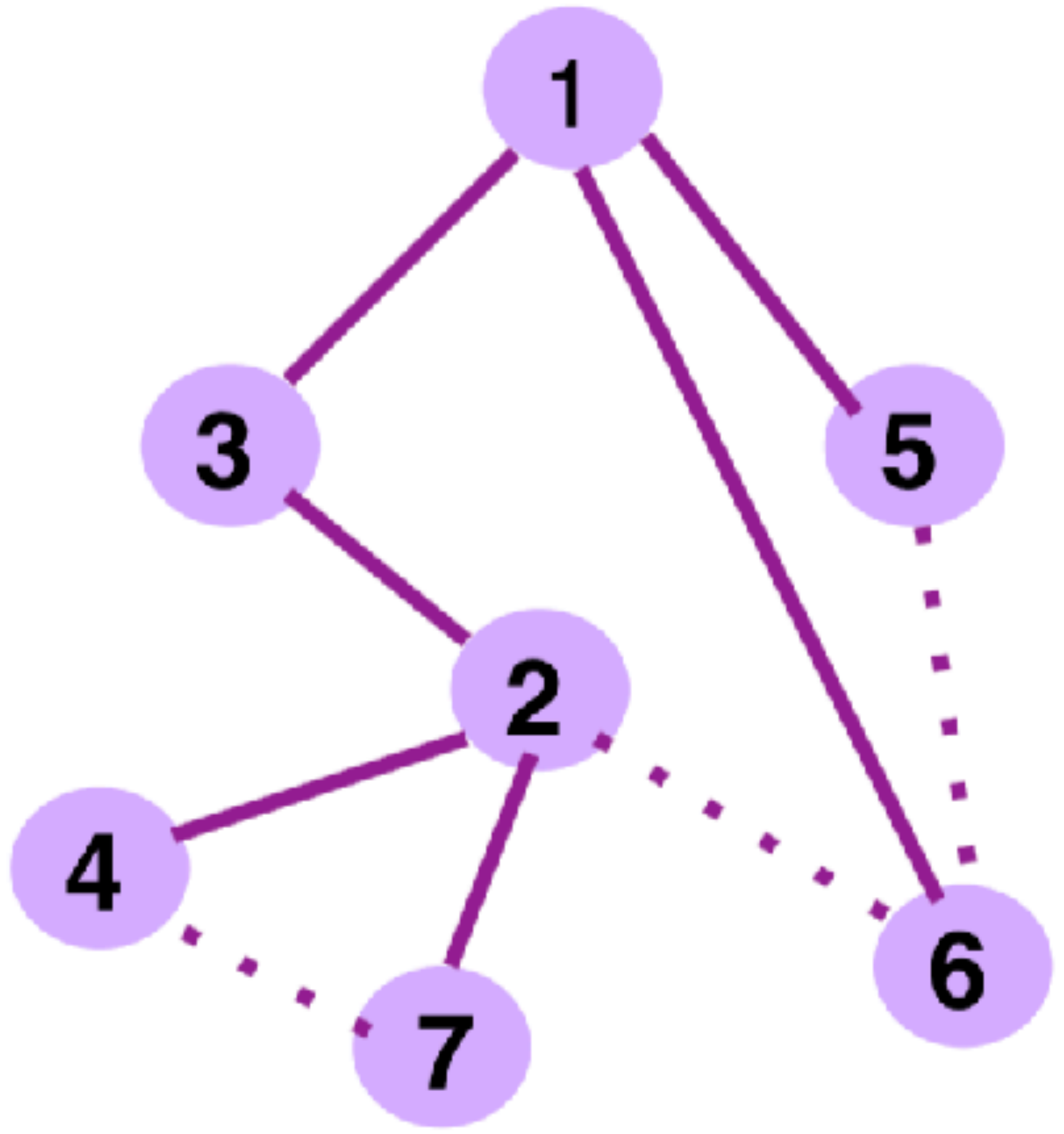
Round 5



	Receive	Send	Next hop
1 (1, 0, 1)			1
2 (1, 2, 2)	(1, 3, 4), (1, 3, 7)		3
3 (1, 1, 3)			1
4 (1, 3, 4)	(1, 3, 7)		2
5 (1, 1, 5)			1
6 (1, 1, 6)			1
7 (1, 3, 7)	(1, 3, 4)		2

After Round 5: We have our Spanning Tree

- 3-1
- 5-1
- 6-1
- 2-3
- 4-2
- 7-2



Questions?

Spanning Tree Protocol ++ (Out of scope)

- Protocol must react to failures
    - Failure of the root node
    - Failure of switches and links

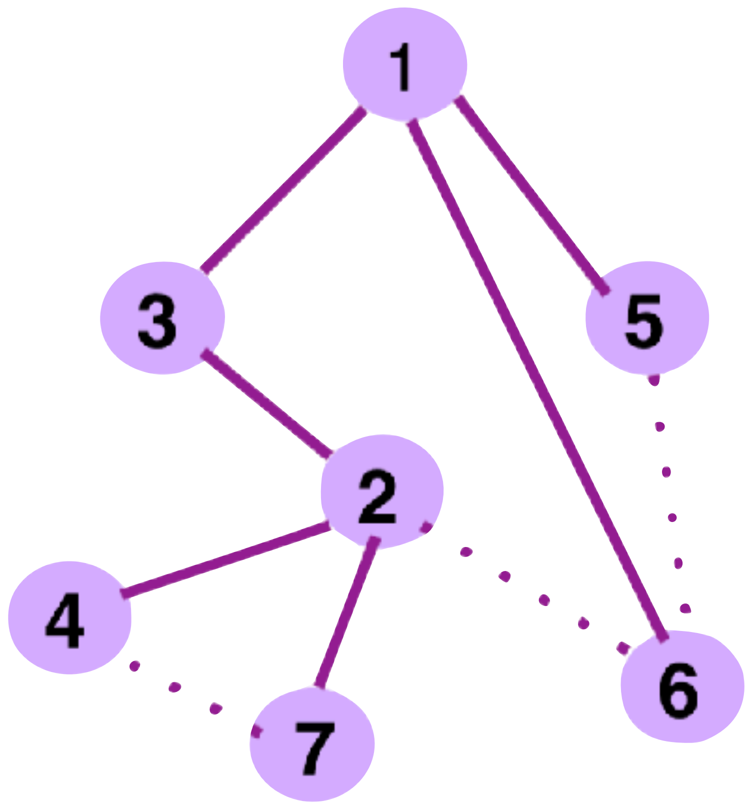
- **Root node sends periodic announcement messages**
    - Few possible implementations, but this is simple to understand
    - Other switches continue forwarding messages as just described

- Detecting failures through timeout (soft state)
    - If the current message has not been **refreshed**, time out and send a (Y, 0, Y) message to all neighbors (in the graph)!

- **Neighbor sends message (Y, d, X) to the node experiences failure**
    - **After receiving multiple (Y, 0, Y)**

Suppose link 2-4 fails

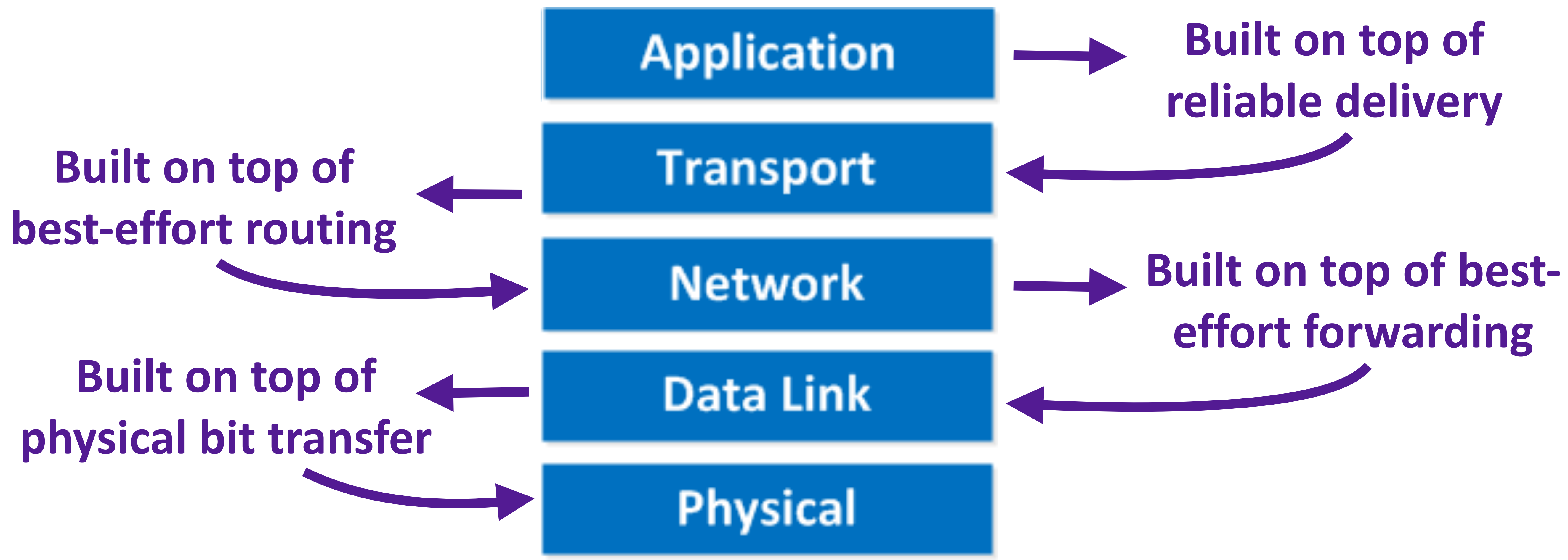
- 4 will send (4, 0, 4) to all its neighbors
    - 4 will stop receiving announcement messages from the root
    - Why?
- At some point, 7 will respond with (1, 3, 7)
- 4 will now update to (1, 4, 4) and send update message
- New spanning tree!



Questions?

The end of Link Layer ….

And the beginning of network layer :-D


Application
Transport
Network
Data Link
Physical
Built on top of reliable delivery
Built on top of best-effort routing
Built on top of best-effort forwarding
Built on top of physical bit transfer

Why do we need a network layer?

- There's only one path from source to destination
- How do you find that path? Ideas?
- Easy to design routing algorithms for trees
  - **Nodes can "flood" packet to all other nodes**

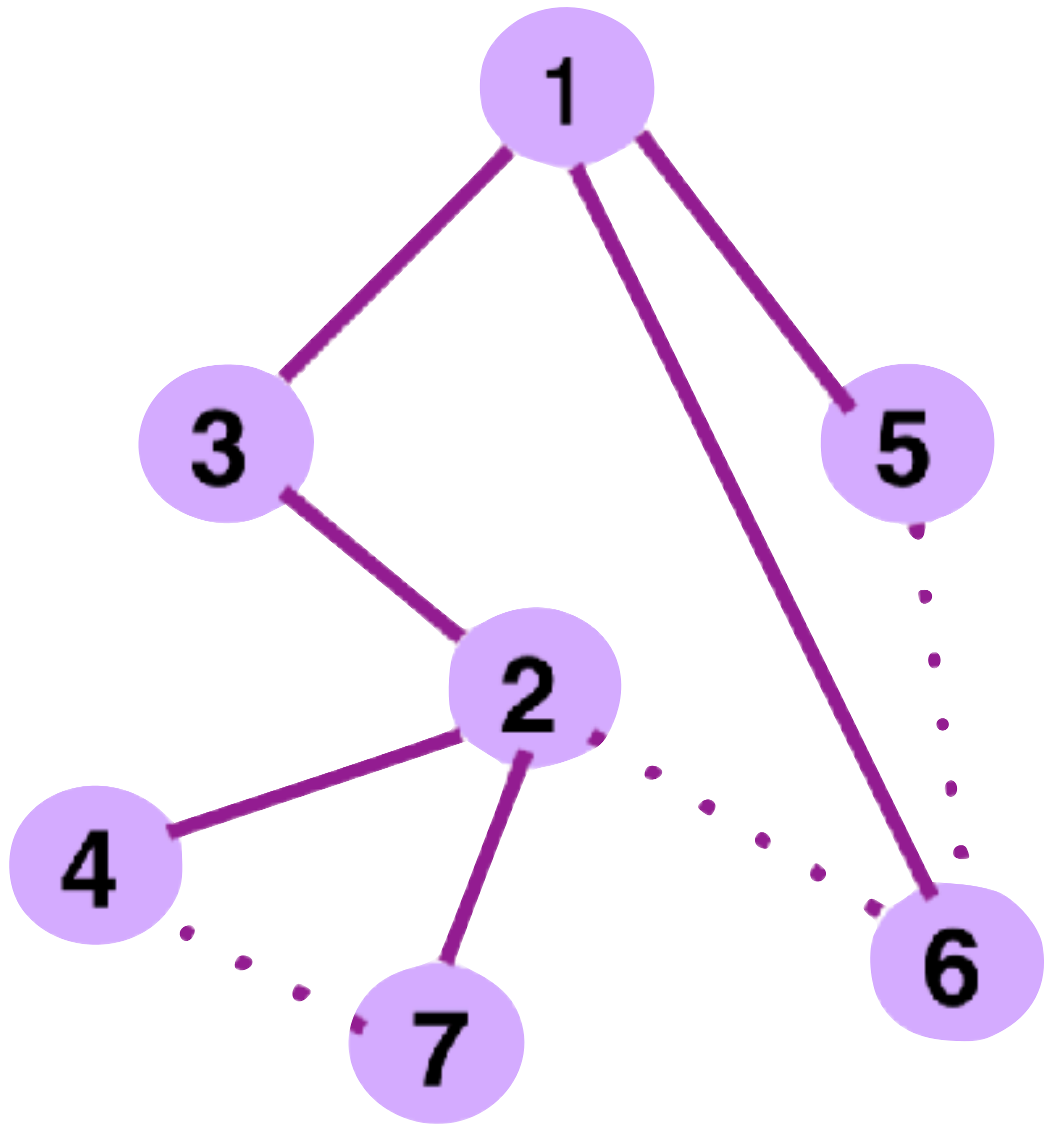
Flooding on a Spanning Tree

- Sends packet to ***every*** node in the network

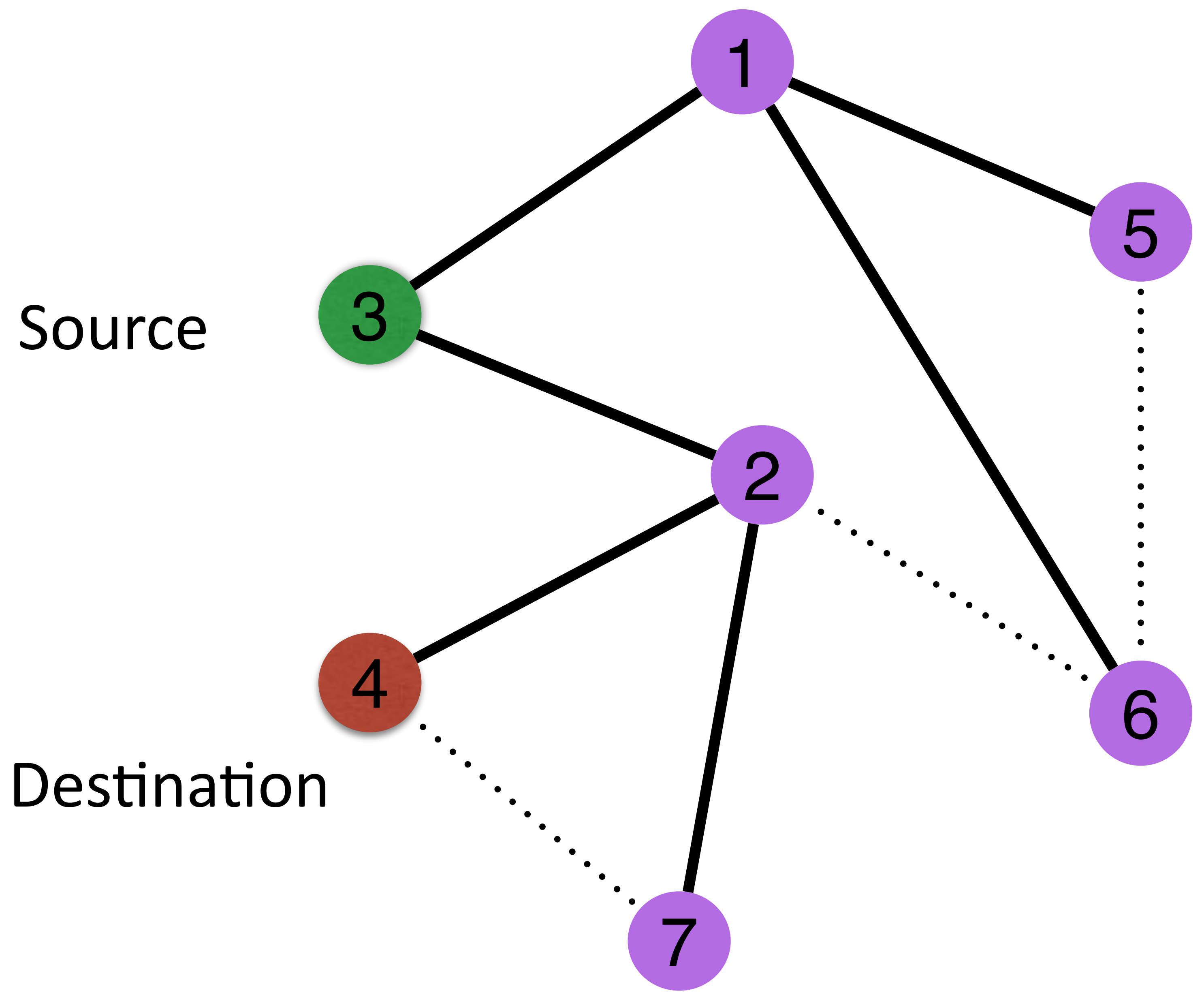
- **Step 1**: Ignore the links not belonging to the Spanning Tree

- **Step 2**: Originating node sends “flood” packet out every link (on spanning tree)

- **Step 3**: Send incoming packet out to all links **other than the one that sent the packet**

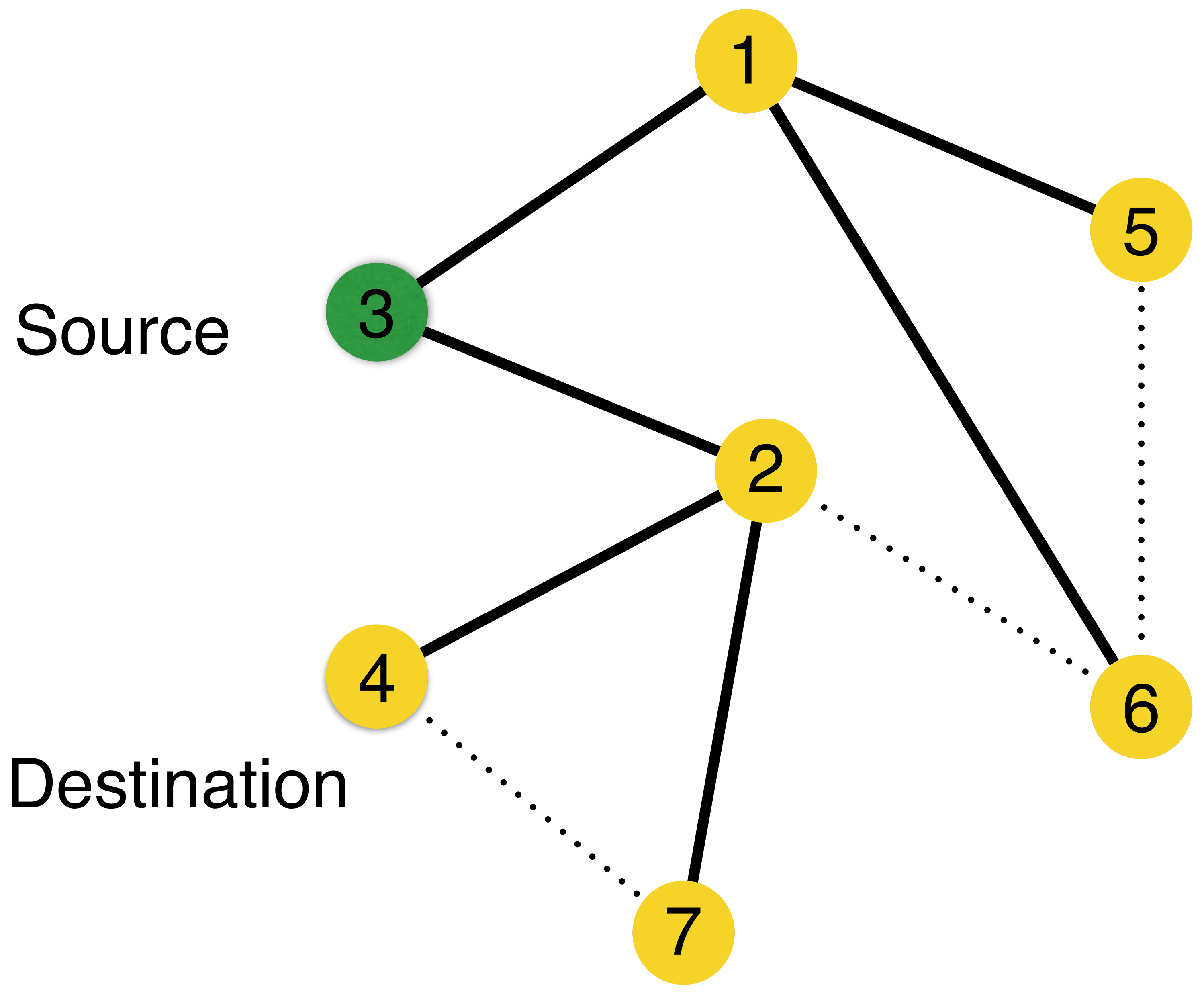


Flooding Example



Flooding Example

Eventually all nodes are covered

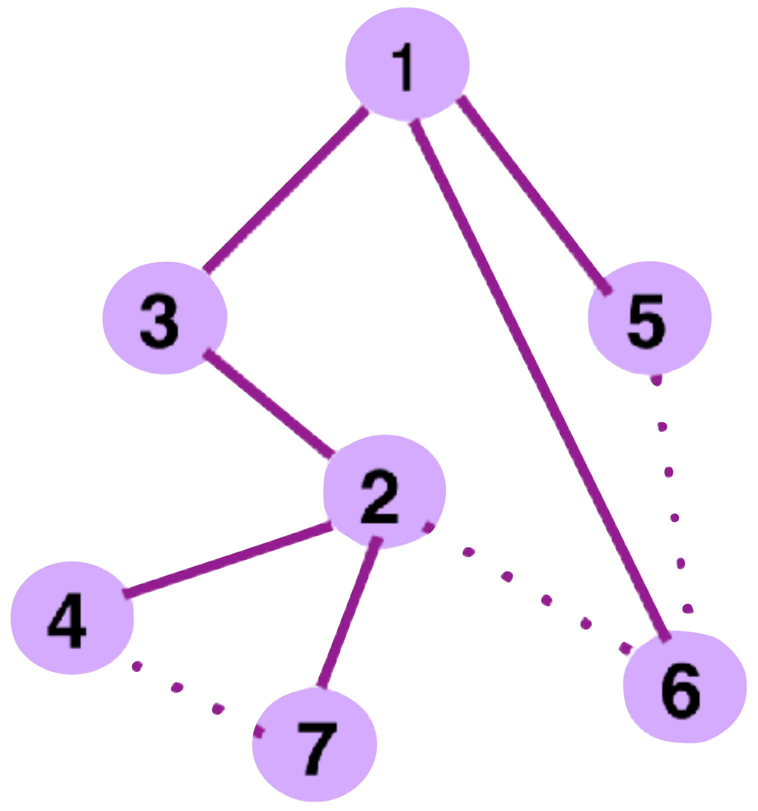


One copy of packet delivered to destination

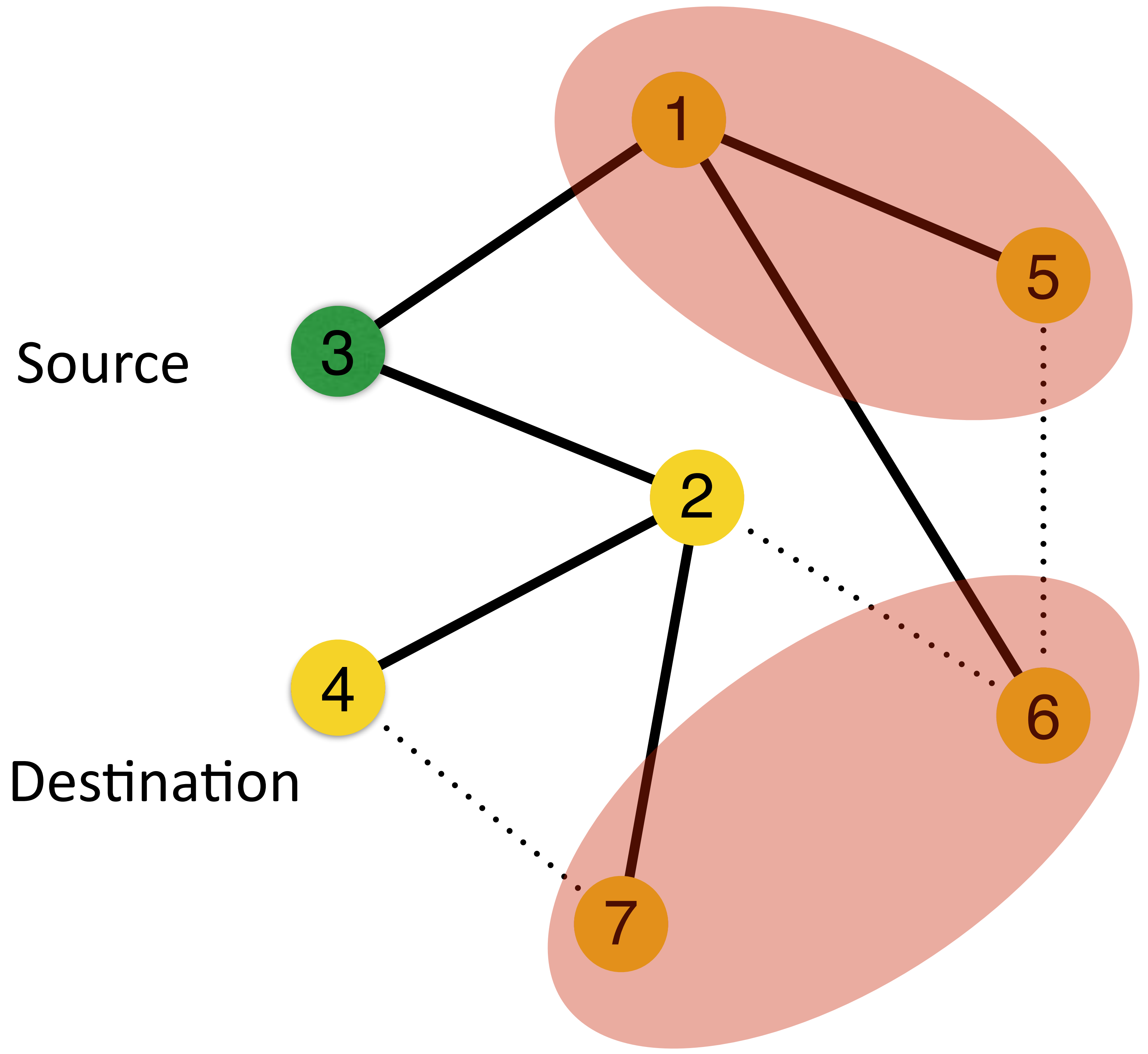
Routing via Flooding on Spanning Tree …

- Easy to design routing algorithms for trees
    - **Nodes can “flood” packet to all other nodes**

- Amazing properties:
    - No routing tables needed!
    - No packets will ever loop.
    - At least (and exactly) one packet must reach the destination
        - Assuming no failures

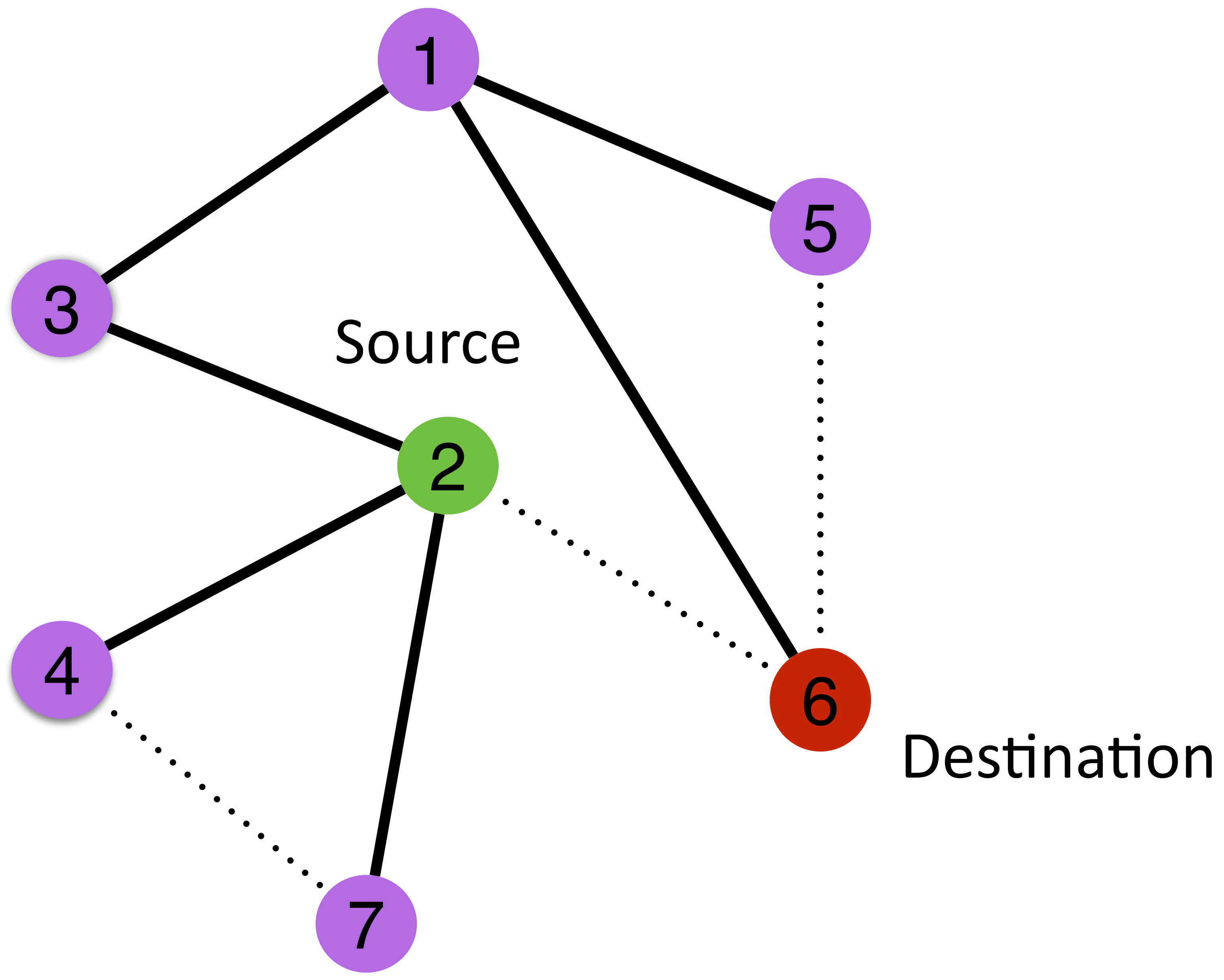


Three fundamental issues!



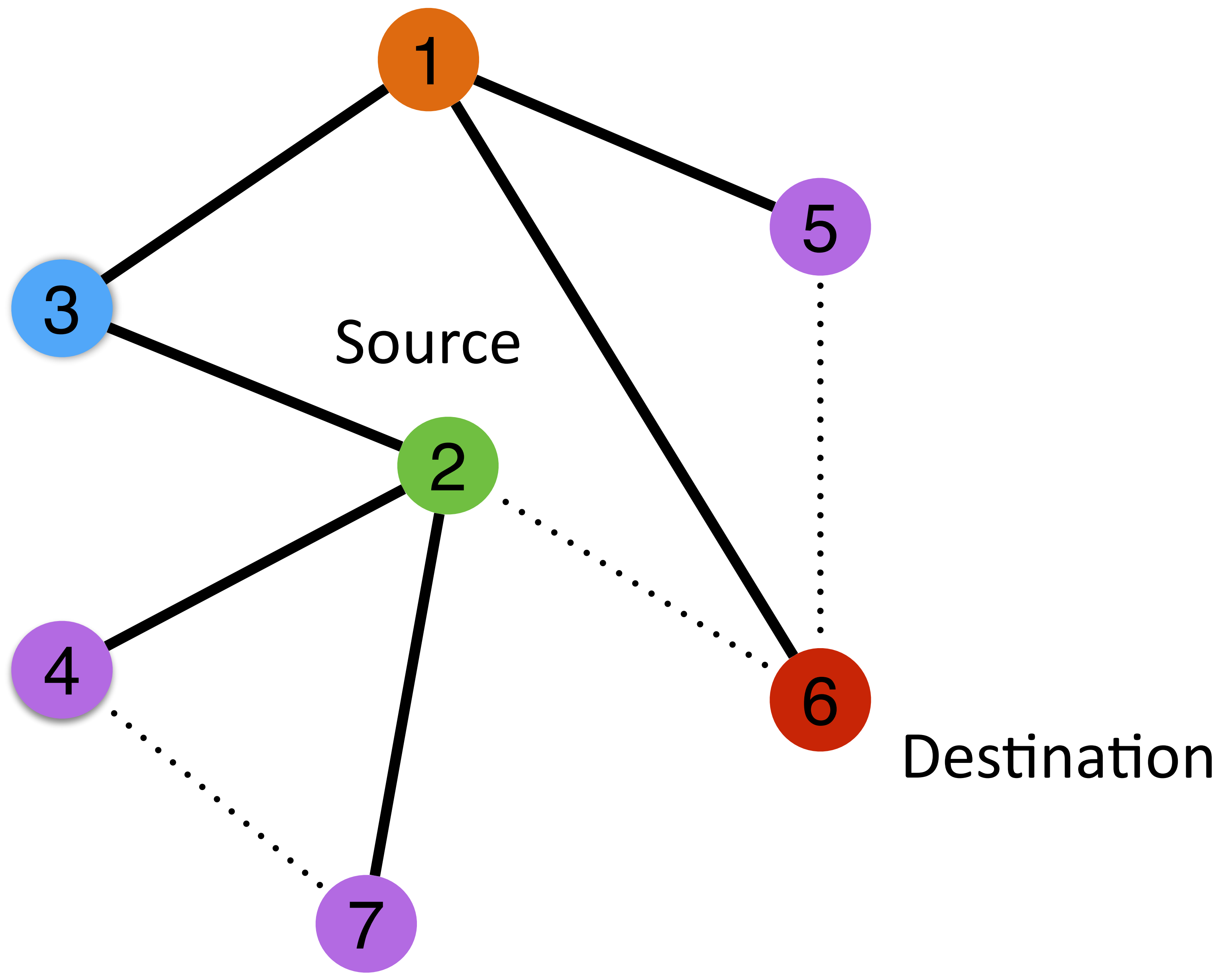
Issue 1: Each host has to do unnecessary packet processing!
(to decide whether the packet is destined to the host)

Three fundamental issues!



Issue 2: Higher latency!
(The packets unnecessarily traverse much longer paths)

Three fundamental issues!



Issue 3: Lower bandwidth availability!
(2-6 and 3-1 packets unnecessarily have to share bandwidth)

Questions?

Why do we need a network layer?

- Network layer performs “routing” of packets to alleviate these issues
- Uses routing tables
- Lets understand routing tables first
    - **We will see routing tables are nothing but …**
    - **Guess?**
    - **….**