Introduction to Computer Networks

CS640

Ethernet

https://pages.cs.wisc.edu/~mgliu/CS640/S26/index.html

Ming Liu
mgliu@cs.wisc.edu

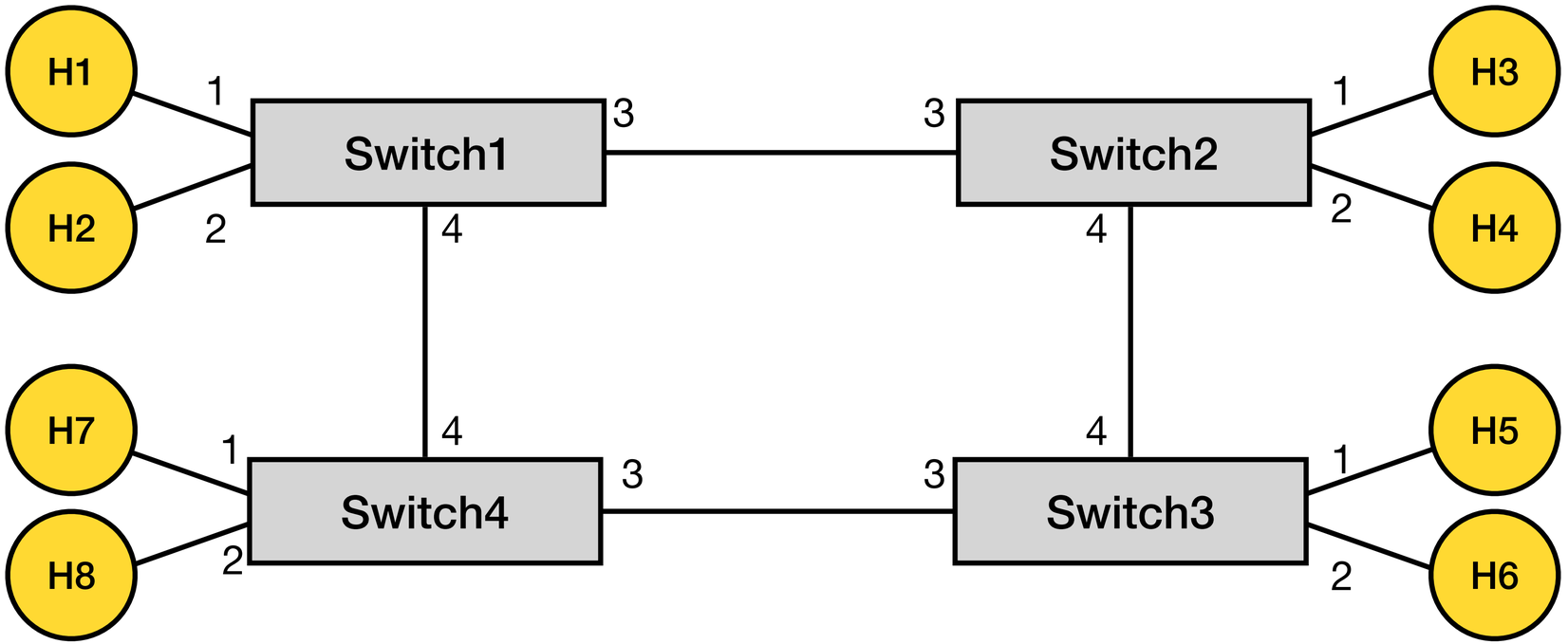
Outline

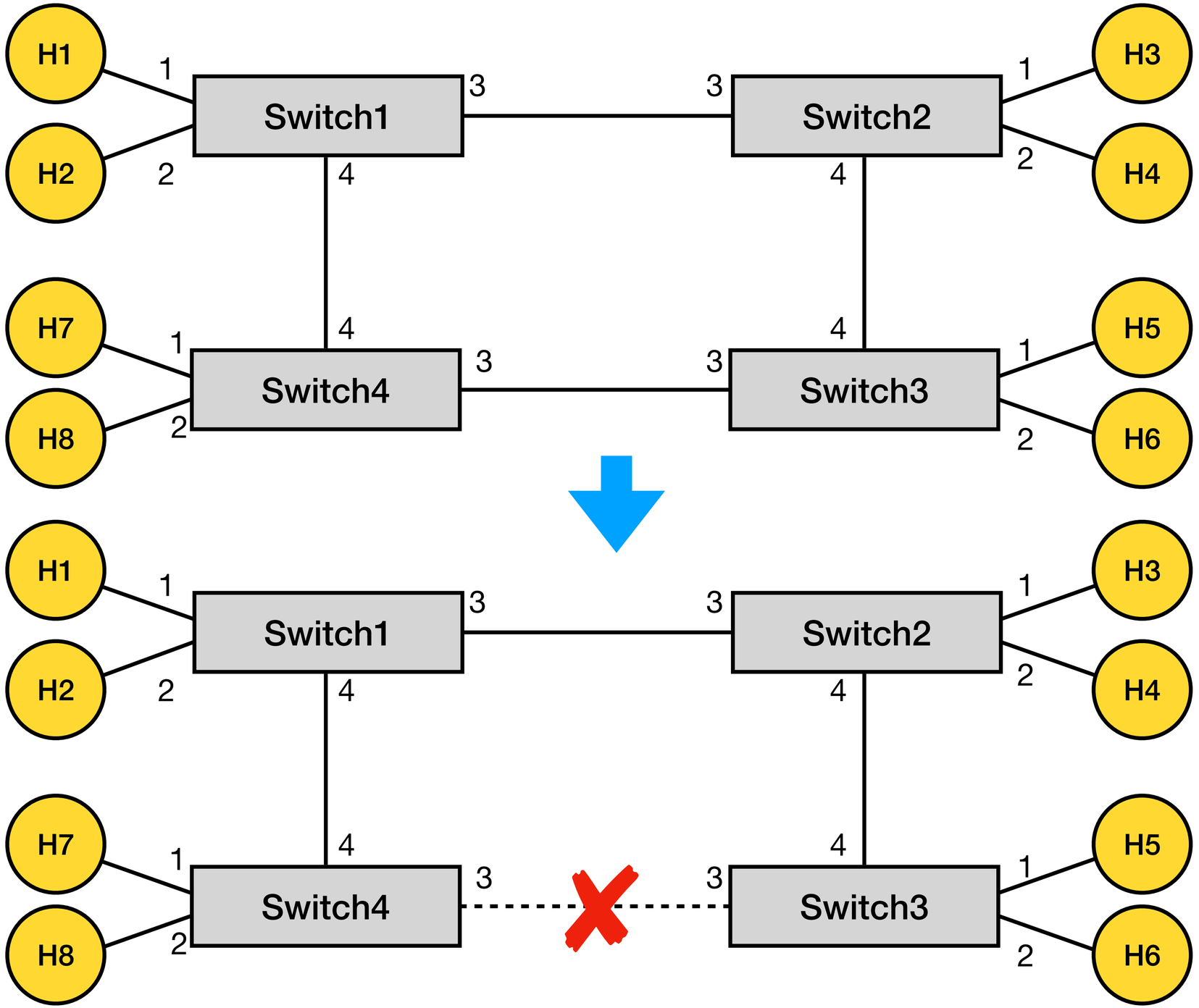
- Last
  - L2 Switching
- Today
  - STP
  - Ethernet
- Announcements
  - N/A

Recap

- Key Questions
  - How is an Ethernet framework forwarded?
  - How is the forwarding table constructed?
  - How can we avoid forwarding loops?

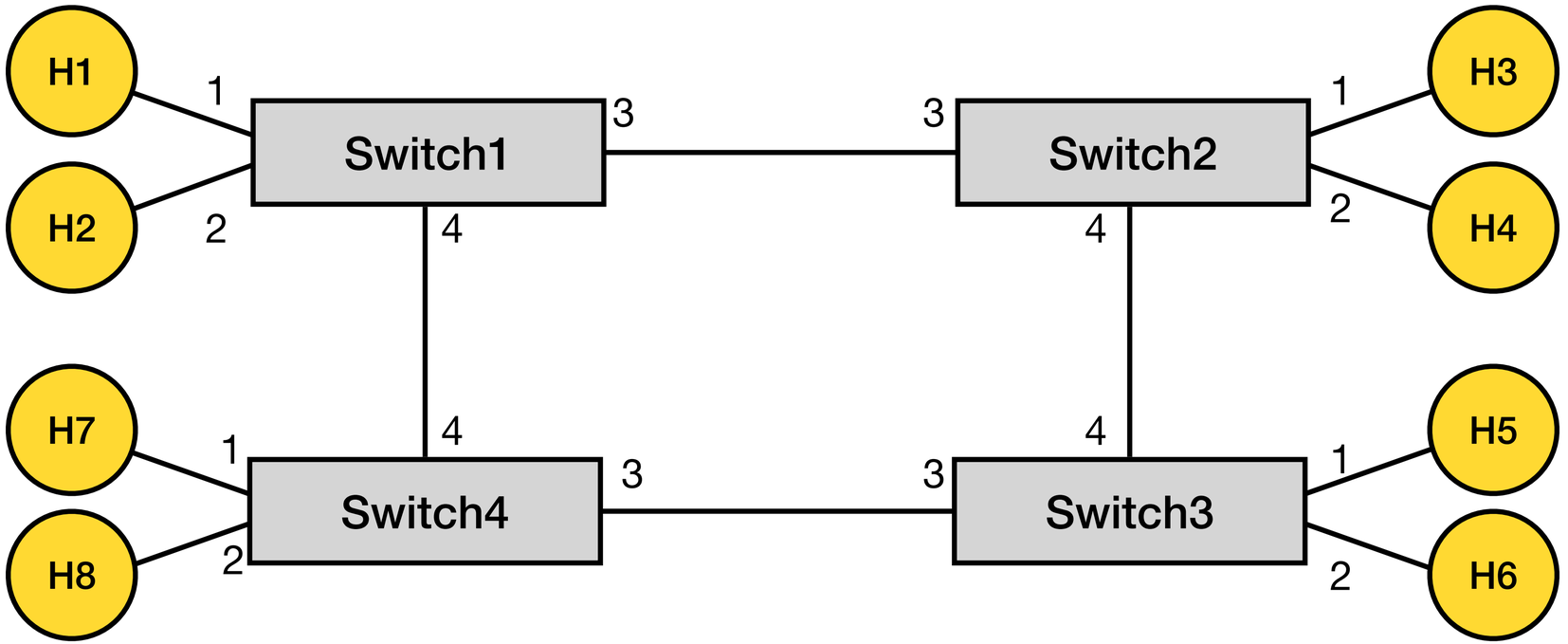
- Terminology
  - MAC address, MAC learning
  - Store-and-forward, cut-through
  - Spanning Tree Protocol (STP)


H1
H2
H3
H4
H5
H6
H7
H8
Switch1
Switch2
Switch3
Switch4
1
2
3
4


H1
H2
H7
H8
H3
H4
H5
H6
Switch1
Switch2
Switch3
Switch4
1
2
3
4

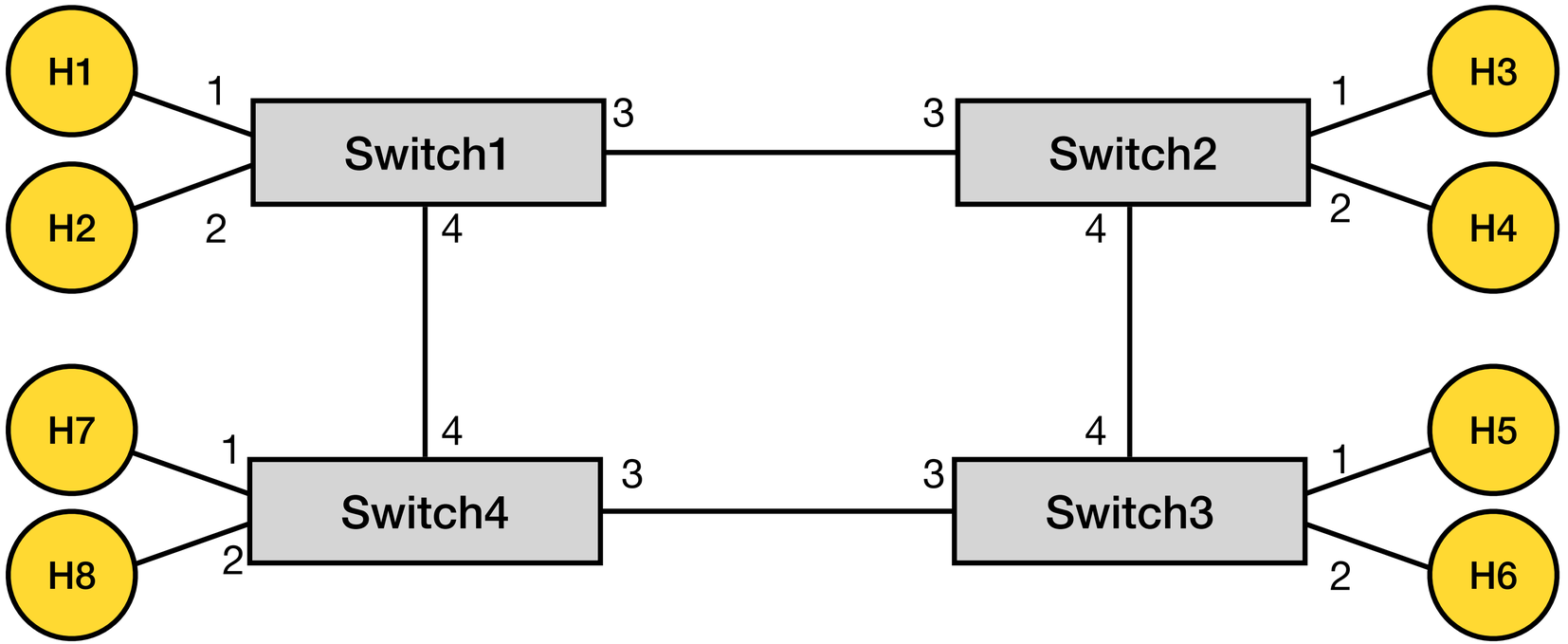
Suppose we have a global central view

- STP becomes simple
  - Designate one node as the root
  - Find the shortest path from all other nodes to the root nodes



Suppose we have a global central view

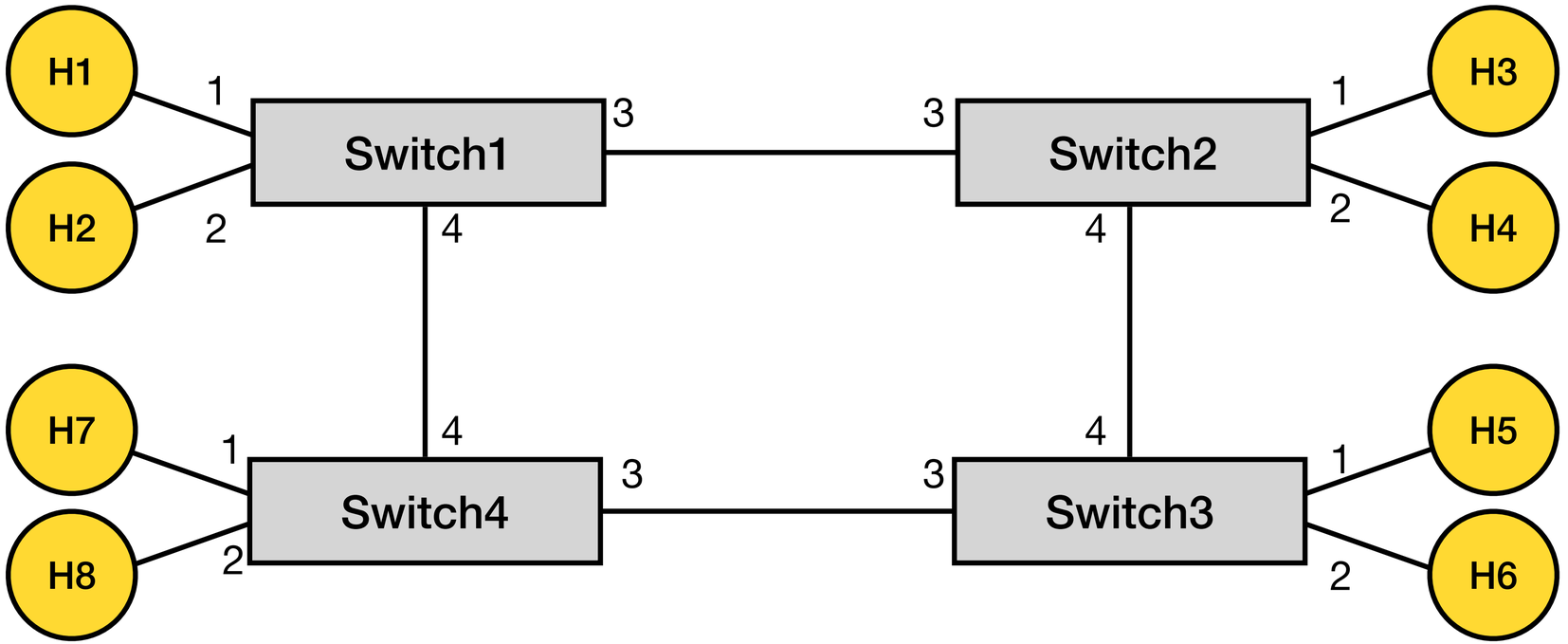
- STP becomes simple
  - Designate one node as the root
  - Find the shortest path from all other nodes to the root nodes
- But without the global central view,



STP #1: States Maintained at the Switch

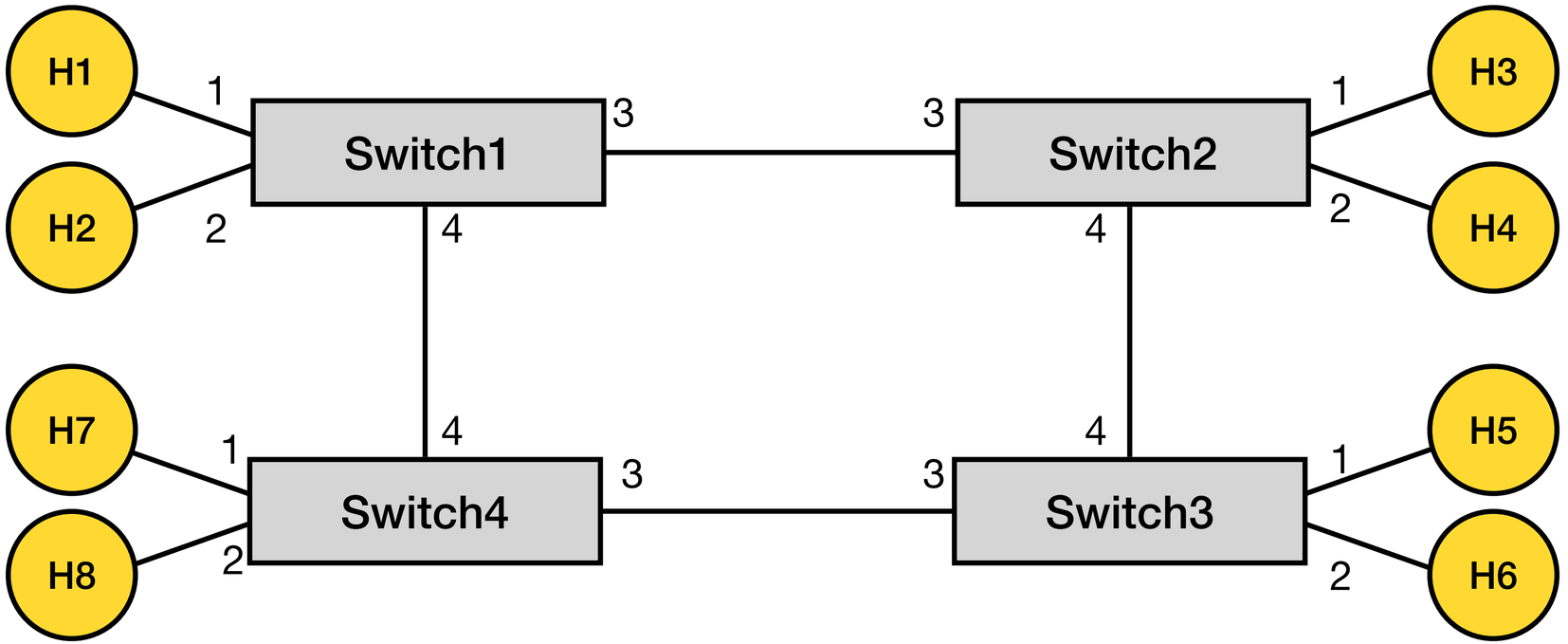
- Four states
  - Local switch ID —> Assigned by the network operator
  - **The switch ID of the root**
  - **The distance and port (i.e., the number of hops) to the root**
  - **Per-port action table**

} STP protocol



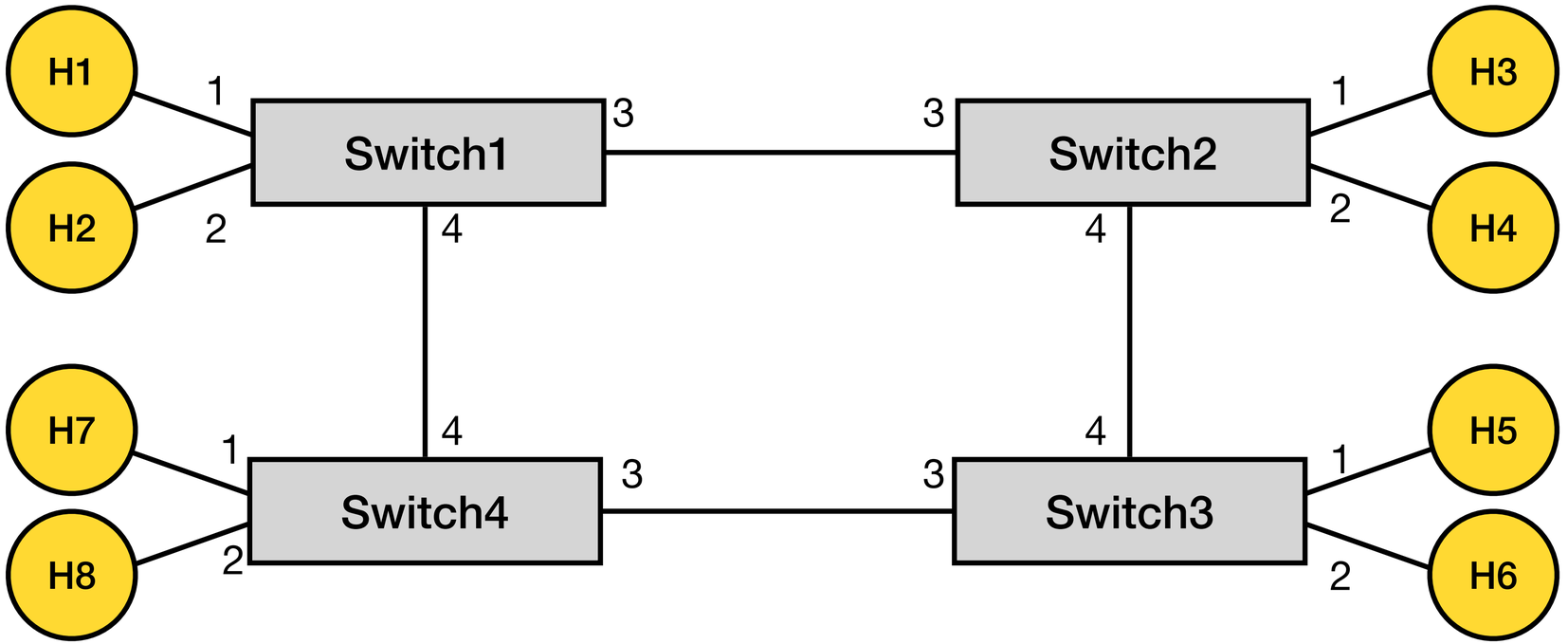
STP #1: States Maintained at the Switch

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1						
Switch 2	2						
Switch 3	3						
Switch 4	4						



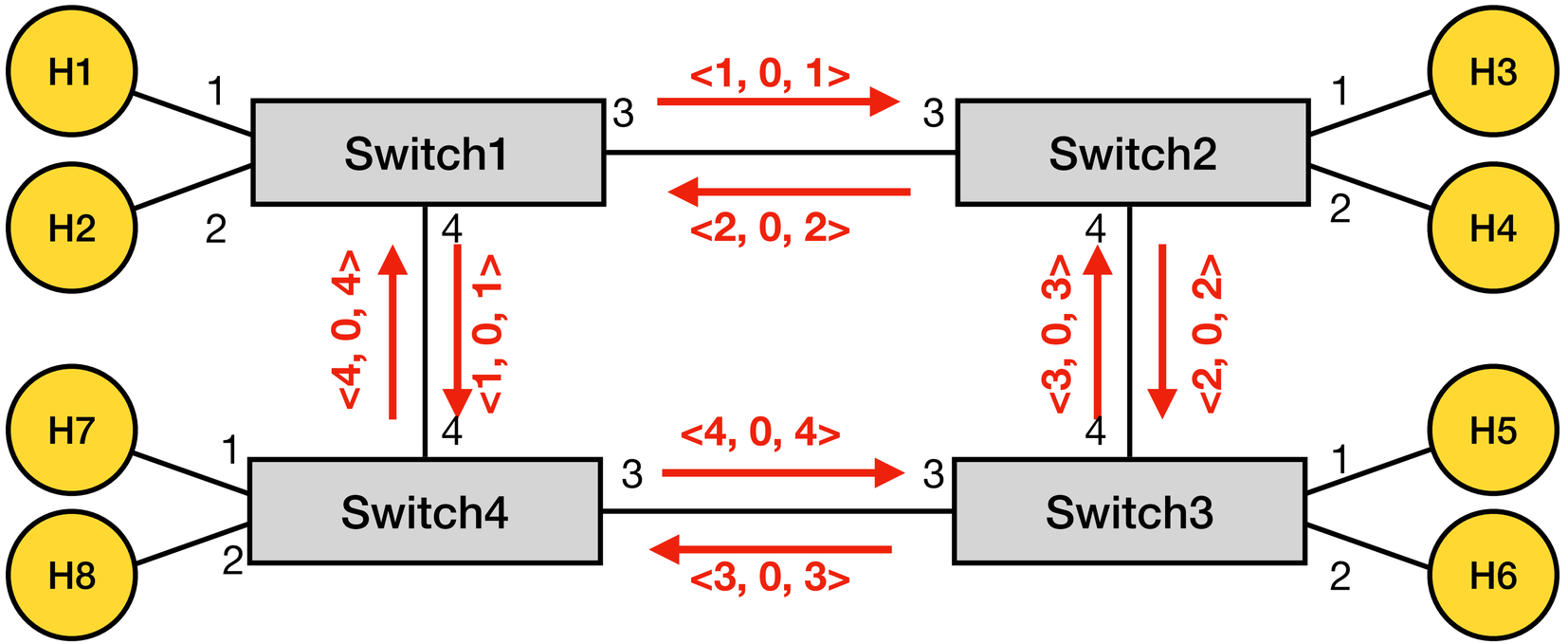
STP #2: Configuration Message

- Three tuples <Y, d, X>
  - Y: the root switch ID in my view
  - d: the distance to the root
  - X: my local switch ID
- The configuration message is issued periodically



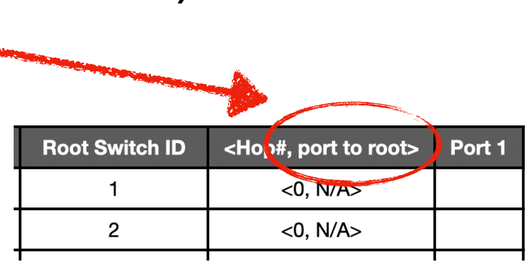
STP #2: Configuration Message

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>				
Switch 2	2	2	<0, N/A>				
Switch 3	3	3	<0, N/A>				
Switch 4	4	4	<0, N/A>				



STP #3: Protocol Actions

- Action #1: Root determination
  - If the root switch ID of the configuration message (<Y, d, X>) is **smaller than** the root switch ID of my local states, the switch should accept the new root switch and perform the following four operations:
    - Change my root switch ID to **Y**
    - Update the hop# (d_cur) to **d_cur = d+1**
    - Mark the switch port that receives the configuration message as **"Broadcast_YES (BC_YES)"**
    - Mark the prior saved switch port (if it existed) as **"Broadcast_NO (BC_NO)"**
  - Otherwise, go to Action #2



STP #3: Protocol Actions

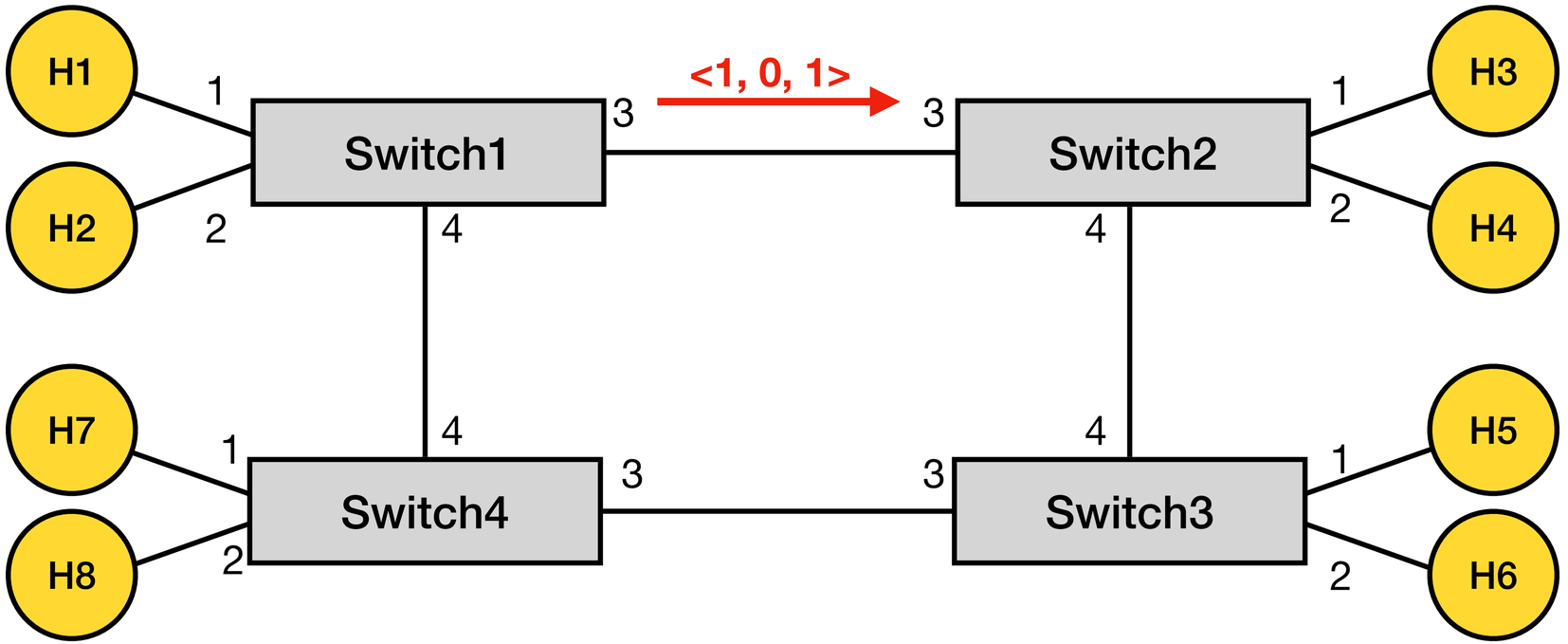
- Action #2: Path determination
  - If the root switch ID of the configuration message (<Y, d, X>) is **the same as** the root switch ID of my local states, the switch should examine the hop# to figure out the shortest path and perform the following operations:
    - If **d+1 < d_cur**, the switch should (a) update the hop# (d_cur) to **d_cur = d+1**; (b) mark the switch port that receives the configuration message as **"Broadcast_YES (BC_YES)"**; (c) mark the prior saved switch port (if it existed) as **"Broadcast_NO (BC_NO)"**;
    - If **d+1 >= d_cur**, the switch should (a) discard the message; (b) mark the switch port that receives the configuration message as **"Broadcast_NO (BC_NO)"**. The root switch skips (b) and marks the port as **"Broadcast_YES (BC_YES)**;
  - Otherwise, go to Action #3

STP #3: Protocol Actions

- Action #3: Discard and block
  - The switch should (a) discard the message: (b) mark the switch port that receives the configuration message as **“Broadcast_NO (BC_NO)”**. If this is the root switch, (b) is skipped and the switch marks the port as **“Broadcast_YES (BC_YES)**;

A Running Example (Switch 1 —> Switch 2)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>				
Switch 2	2	2	<0, N/A>				
Switch 3	3	3	<0, N/A>				
Switch 4	4	4	<0, N/A>				



A Running Example (Switch 1 —> Switch 2)

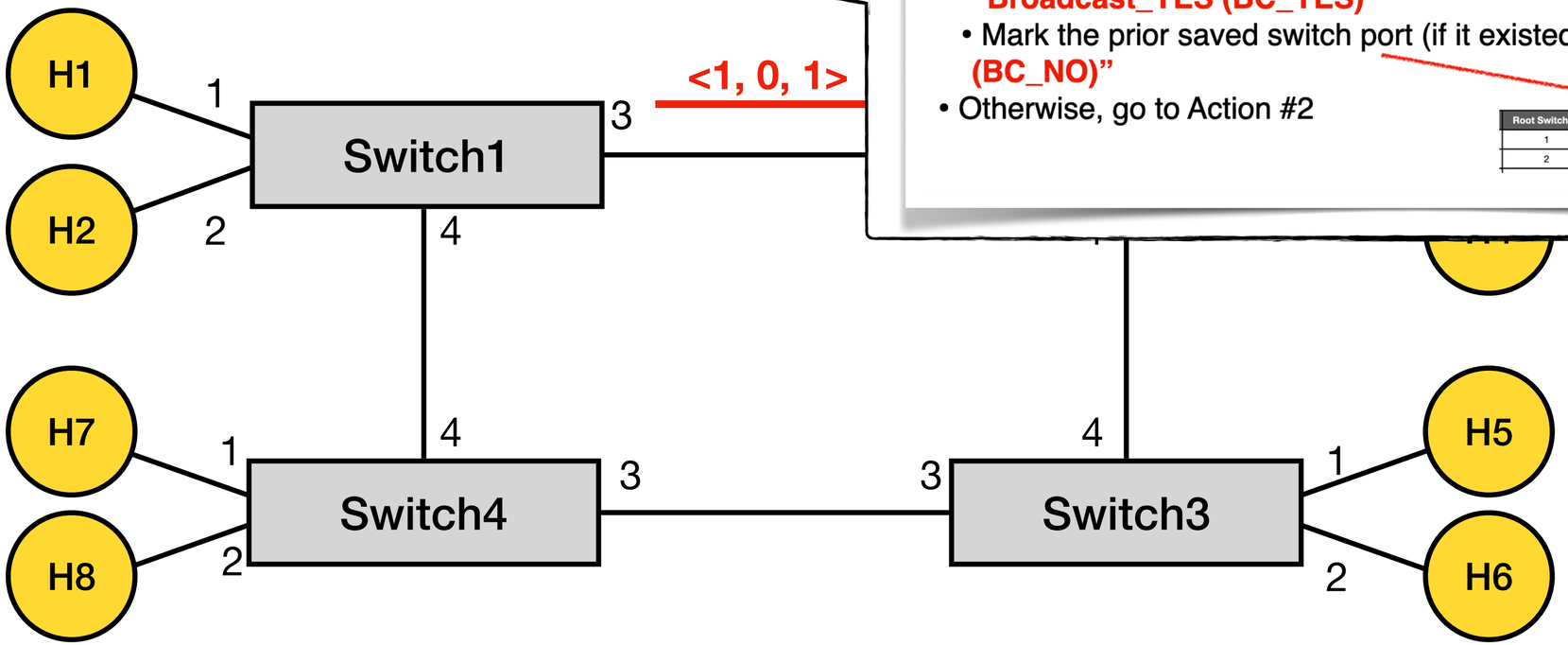
	Local Switch ID	Root Switch ID	<Hop#,
Switch 1	1	1	<(
Switch 2	2	2	<(
Switch 3	3	3	<(
Switch 4	4	4	<(

Action #1: Root Determination

Action #1: Root determination

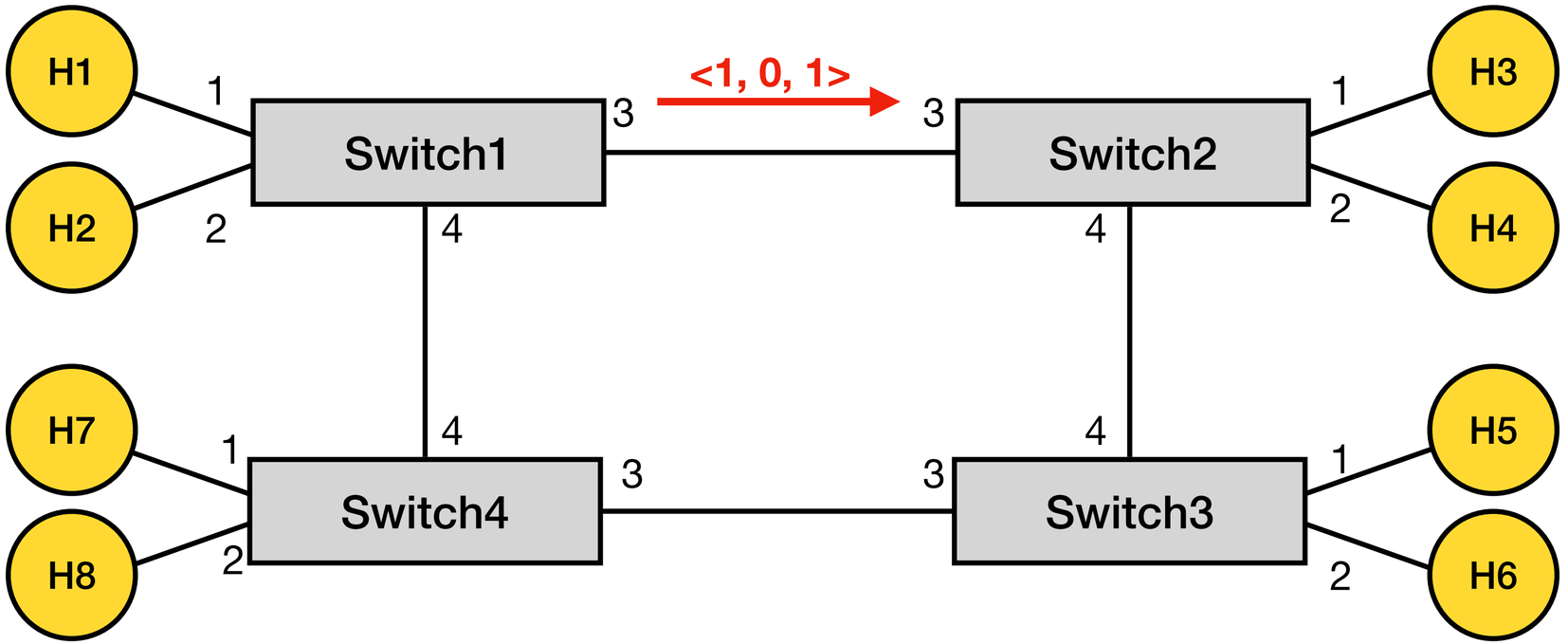
- If the root switch ID of the configuration message (<Y, d, X>) is **smaller than** the root switch ID of my local states, the switch should accept the new root switch and perform the following four operations:
  - Change my root switch ID to **Y**
  - Update the hop# (d_cur) to **d_cur = d+1**
  - Mark the switch port that receives the configuration message as **"Broadcast_YES (BC_YES)"**
  - Mark the prior saved switch port (if it existed) as **"Broadcast_NO (BC_NO)"**
- Otherwise, go to Action #2

Root Switch ID	<Hop#, port to root>	Port 1
1	<0, N/A>	
2	<0, N/A>	



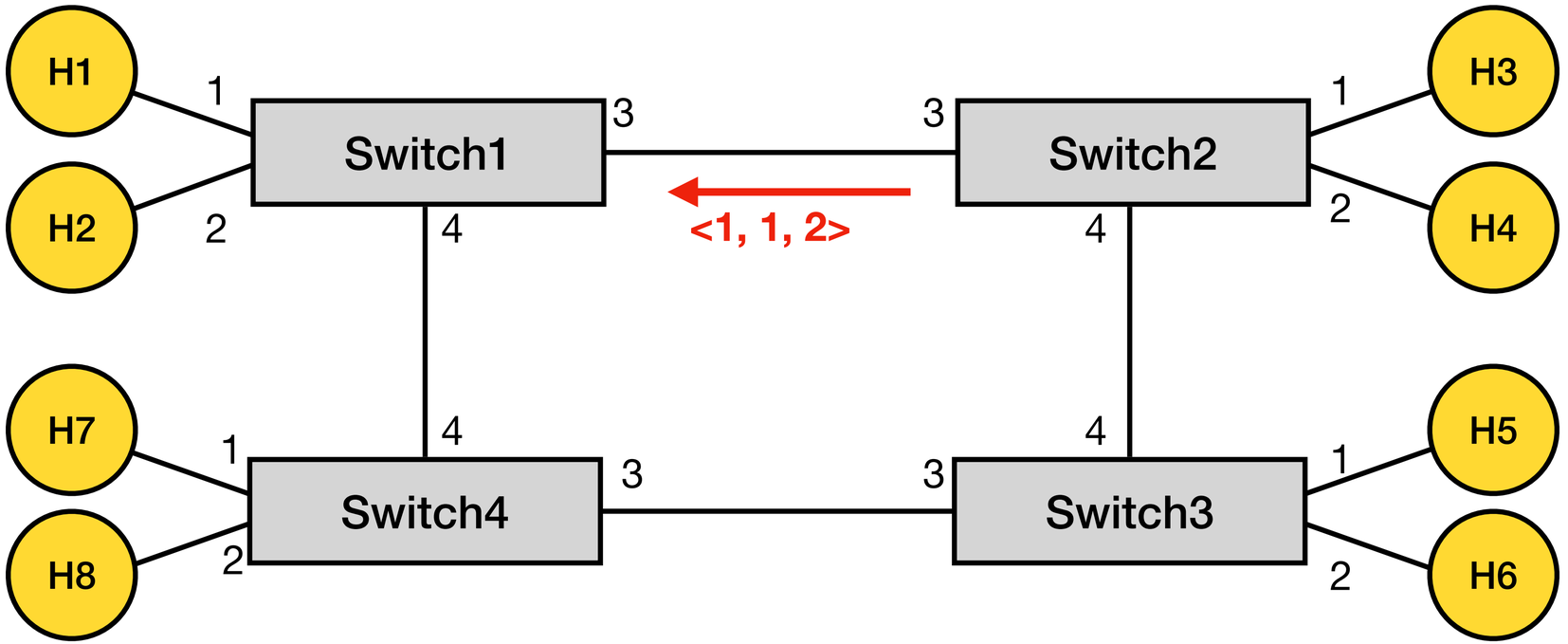
A Running Example (Switch 1 —> Switch 2)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>				
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	3	<0, N/A>				
Switch 4	4	4	<0, N/A>				



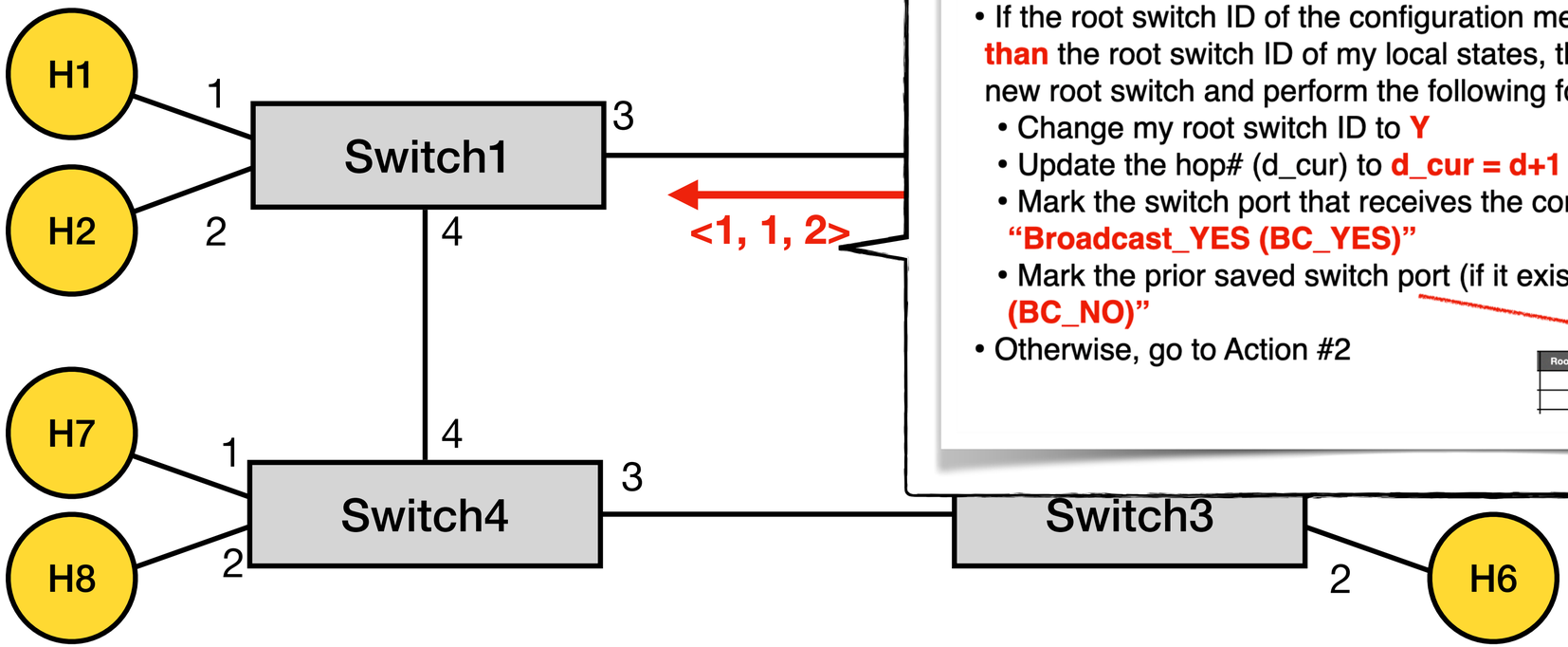
A Running Example (Switch 2 —> Switch 1)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>				
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	3	<0, N/A>				
Switch 4	4	4	<0, N/A>				



A Running Example (Switch 2 —> Switch 1)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>				
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	3	<0,				
Switch 4	4	4	<0,				



Action #1: Root Determination

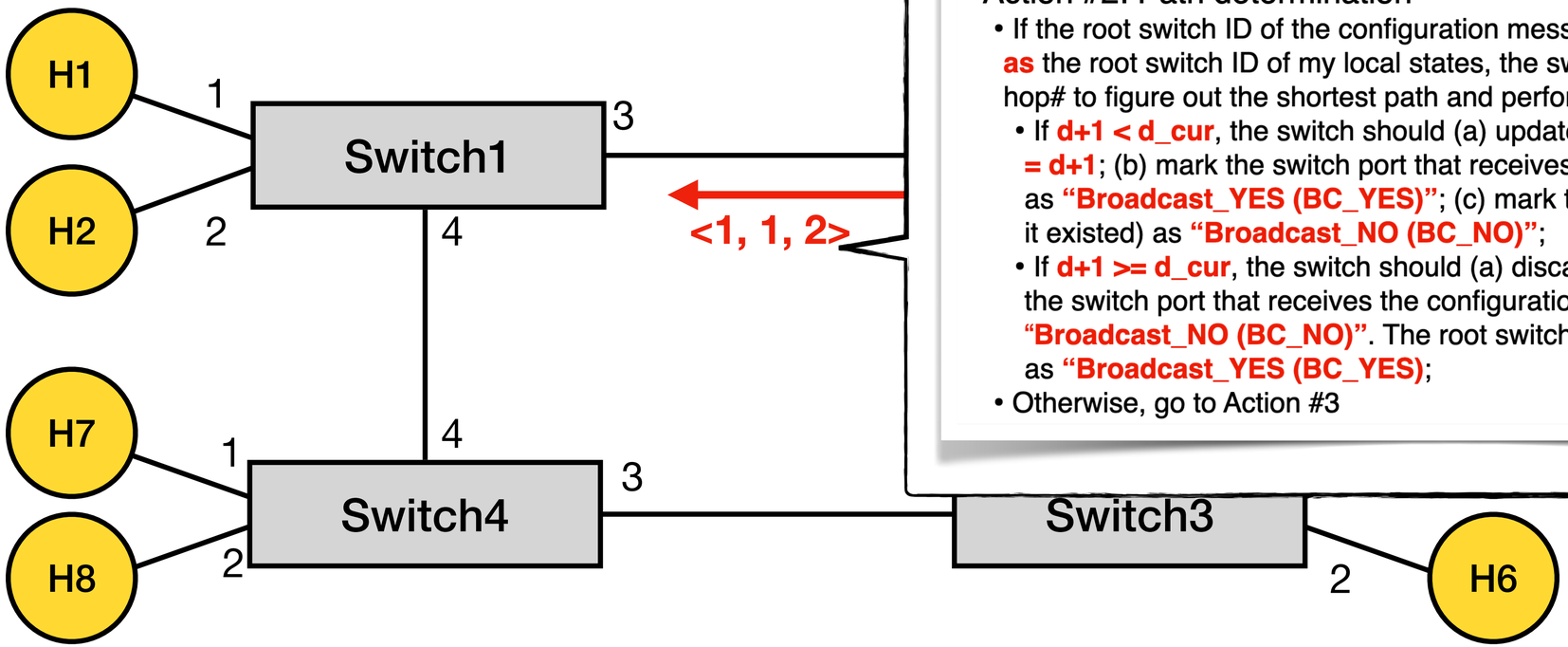
Action #1: Root determination

- If the root switch ID of the configuration message (<Y, d, X>) is **smaller than** the root switch ID of my local states, the switch should accept the new root switch and perform the following four operations:
  - Change my root switch ID to **Y**
  - Update the hop# (d_cur) to **d_cur = d+1**
  - Mark the switch port that receives the configuration message as **"Broadcast_YES (BC_YES)"**
  - Mark the prior saved switch port (if it existed) as **"Broadcast_NO (BC_NO)"**
- Otherwise, go to Action #2

Root Switch ID	<Hop#, port to root>	Port 1
1	<0, N/A>	
2	<0, N/A>	

A Running Example (Switch 2 —> Switch 1)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>				
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	3	<0,				
Switch 4	4	4	<0,				

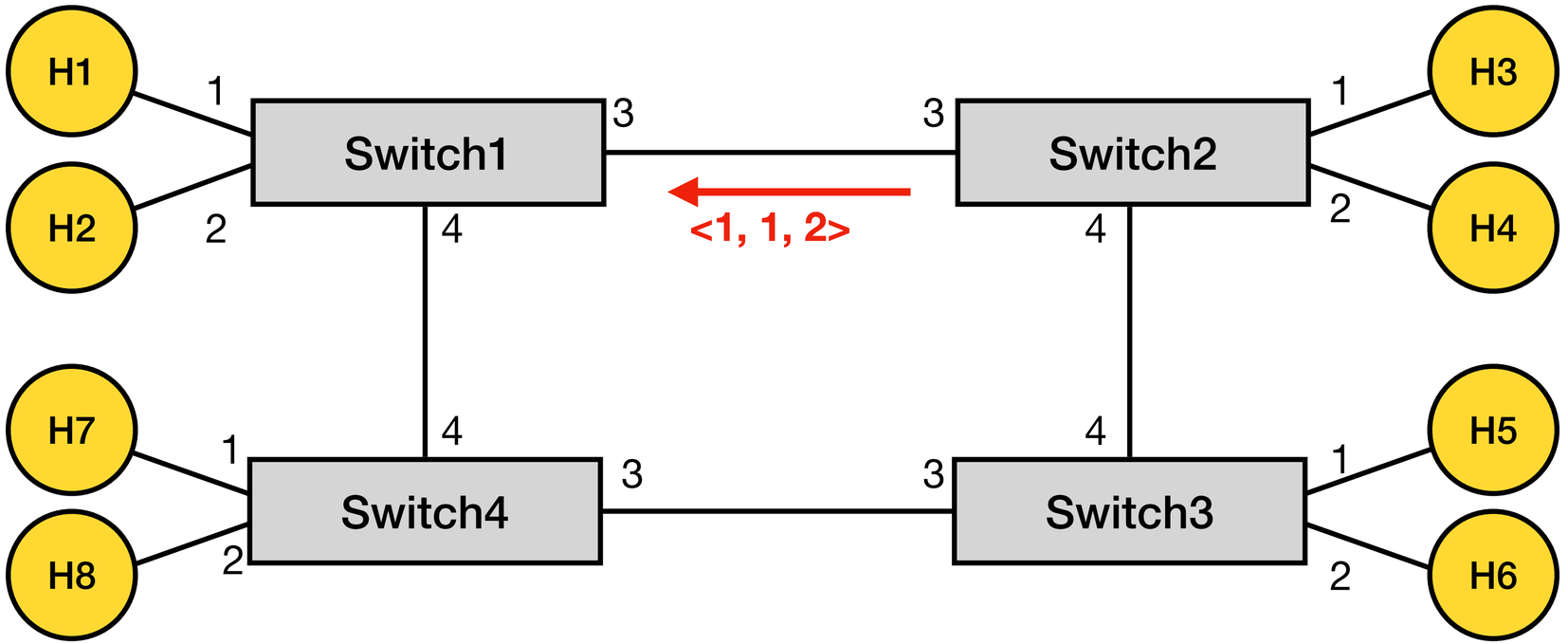


Action #2: Path determination

- Action #2: Path determination
  - If the root switch ID of the configuration message (<Y, d, X>) is **the same as** the root switch ID of my local states, the switch should examine the hop# to figure out the shortest path and perform the following operations:
    - If **d+1 < d_cur**, the switch should (a) update the hop# (d_cur) to **d_cur = d+1**; (b) mark the switch port that receives the configuration message as **"Broadcast_YES (BC_YES)"**; (c) mark the prior saved switch port (if it existed) as **"Broadcast_NO (BC_NO)"**;
    - If **d+1 >= d_cur**, the switch should (a) discard the message; (b) mark the switch port that receives the configuration message as **"Broadcast_NO (BC_NO)"**. The root switch skips (b) and marks the port as **"Broadcast_YES (BC_YES)**;
  - Otherwise, go to Action #3

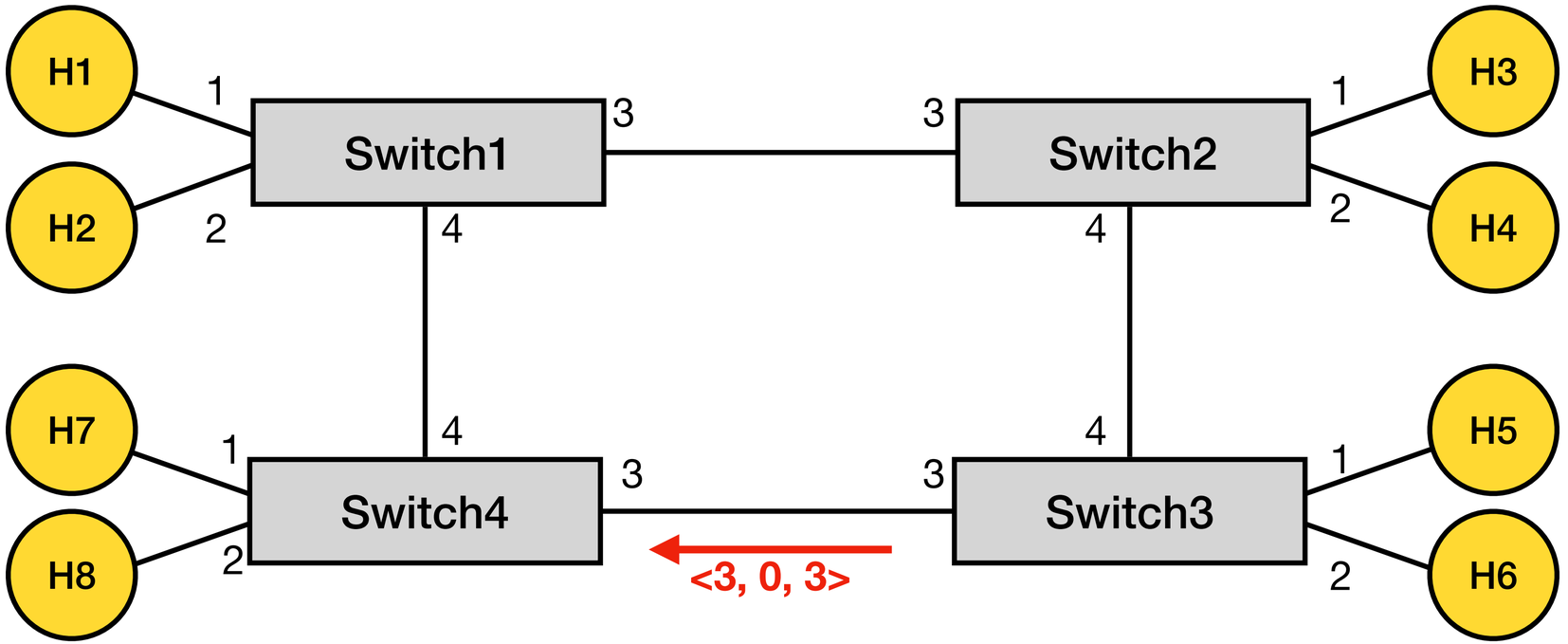
A Running Example (Switch 2 —> Switch 1)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	3	<0, N/A>				
Switch 4	4	4	<0, N/A>				



A Running Example (Switch 3 —> Switch 4)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	3	<0, N/A>				
Switch 4	4	4	<0, N/A>				



A Running Example (Switch 3 —> Switch 4)

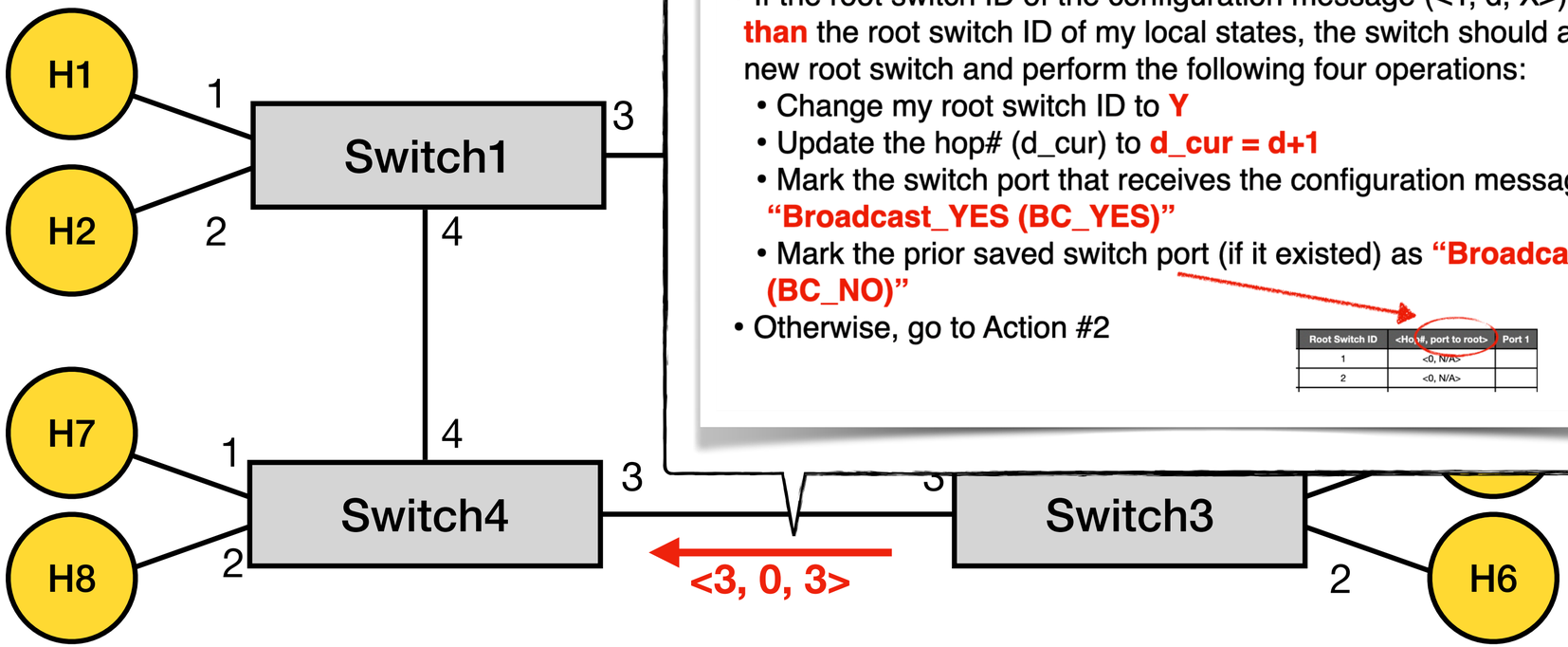
	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	3					
Switch 4	4	4					

Action #1: Root Determination

Action #1: Root determination

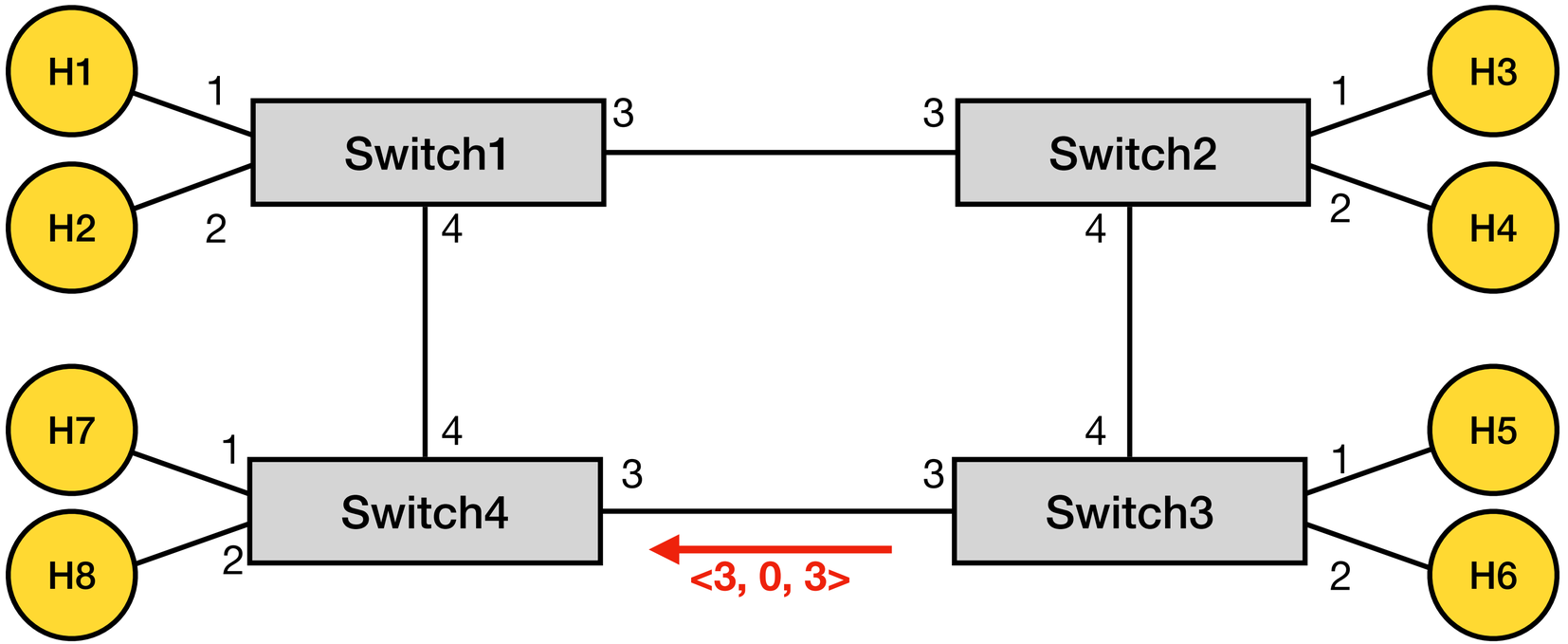
- If the root switch ID of the configuration message (<Y, d, X>) is **smaller than** the root switch ID of my local states, the switch should accept the new root switch and perform the following four operations:
  - Change my root switch ID to **Y**
  - Update the hop# (d_cur) to **d_cur = d+1**
  - Mark the switch port that receives the configuration message as **"Broadcast_YES (BC_YES)"**
  - Mark the prior saved switch port (if it existed) as **"Broadcast_NO (BC_NO)"**
- Otherwise, go to Action #2

Root Switch ID	<Hop#, port to root>	Port 1
1	<0, N/A>	
2	<0, N/A>	



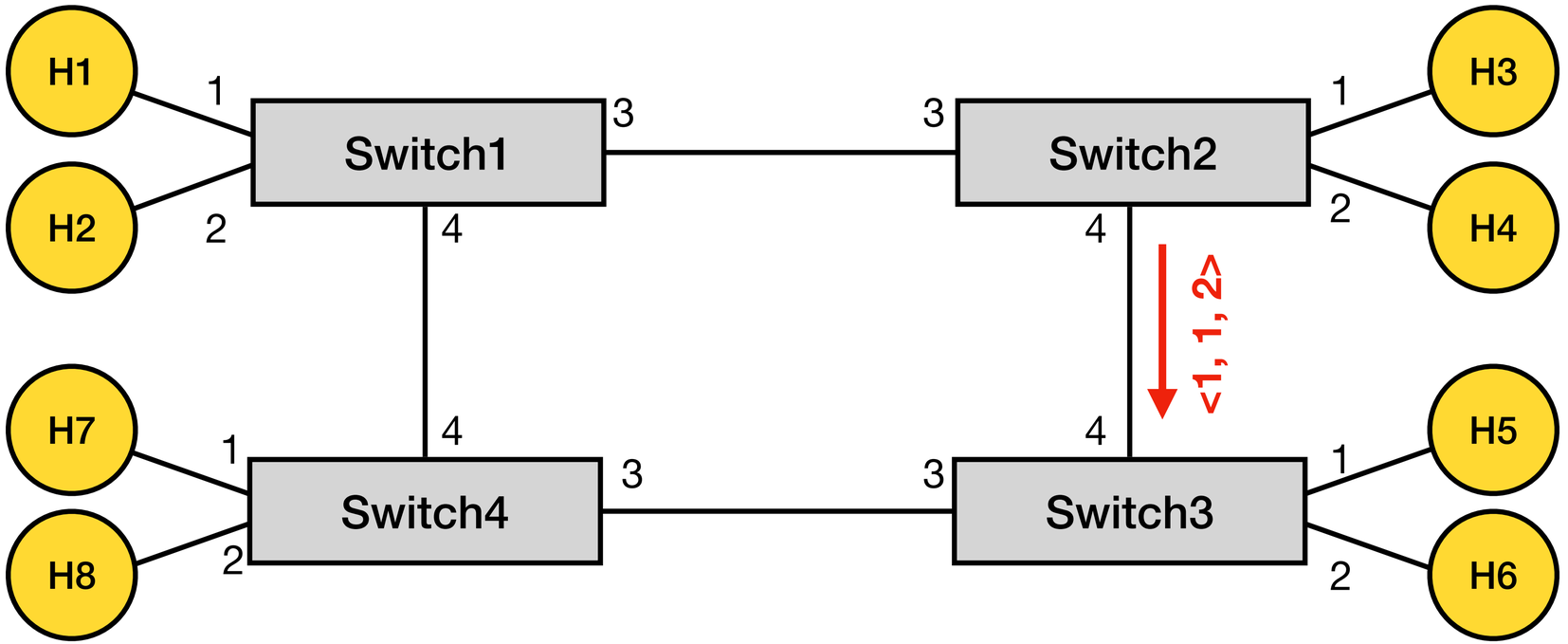
A Running Example (Switch 3 —> Switch 4)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	3	<0, N/A>				
Switch 4	4	3	<1, 3>			BC_YES	



A Running Example (Switch 2 —> Switch 3)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	3	<0, N/A>				
Switch 4	4	3	<1, 3>			BC_YES	



A Running Example (Switch 2 —> Switch 3)

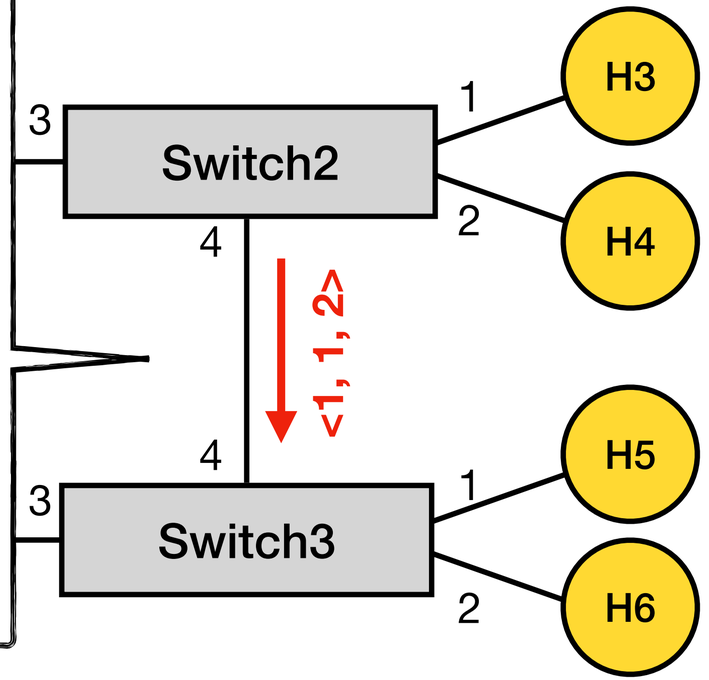
	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	3	<0, N/A>				
[illegible]	[illegible]	[illegible]	3>			BC_YES	

Action #1: Root Determination

Action #1: Root determination

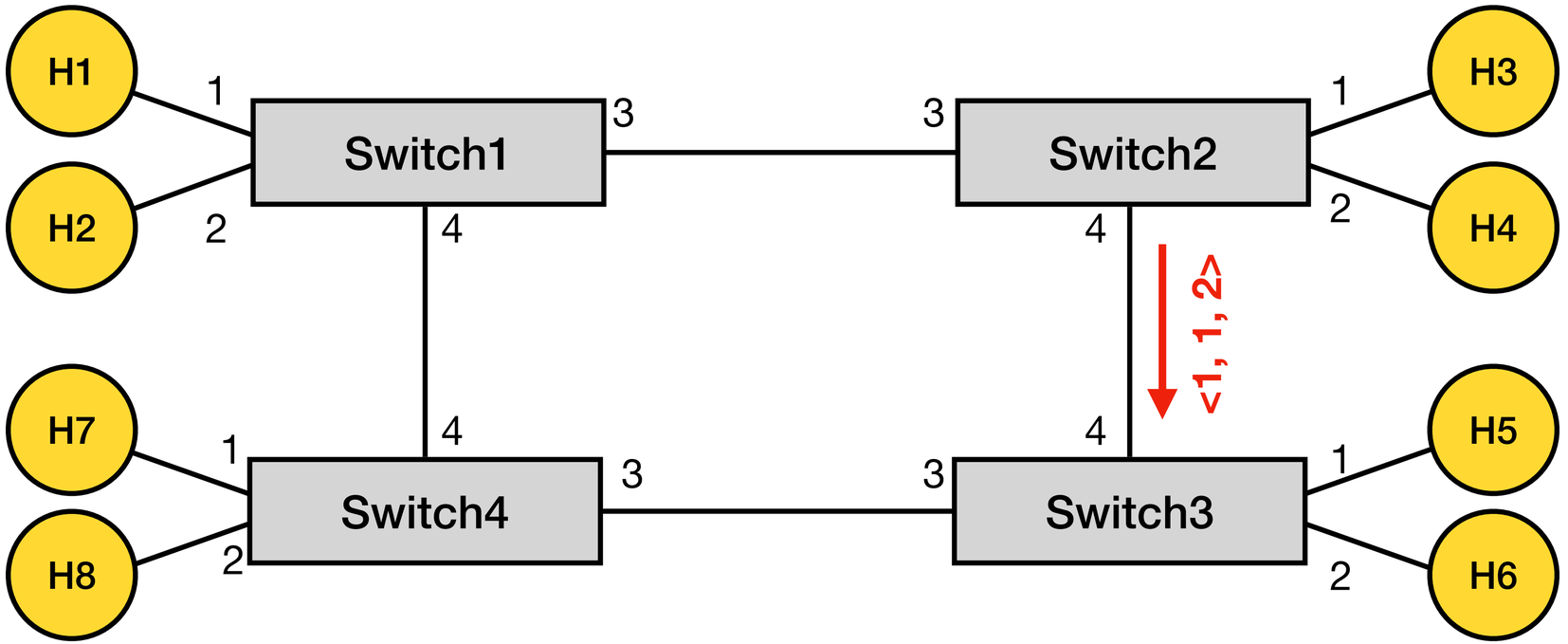
- If the root switch ID of the configuration message (<Y, d, X>) is **smaller than** the root switch ID of my local states, the switch should accept the new root switch and perform the following four operations:
  - Change my root switch ID to **Y**
  - Update the hop# (d_cur) to **d_cur = d+1**
  - Mark the switch port that receives the configuration message as **"Broadcast_YES (BC_YES)"**
  - Mark the prior saved switch port (if it existed) as **"Broadcast_NO (BC_NO)"**
- Otherwise, go to Action #2

Root Switch ID	<Hop#, port to root>	Port 1
1	<0, N/A>	
2	<0, N/A>	



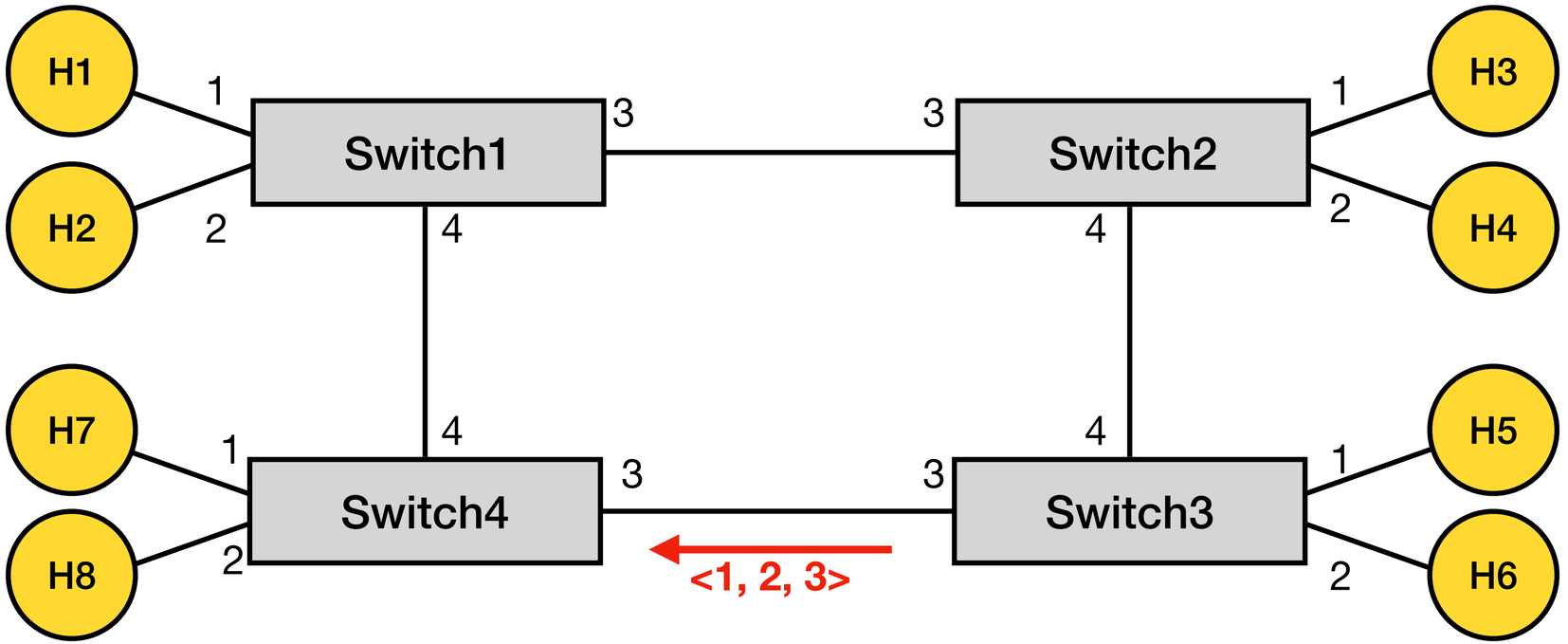
A Running Example (Switch 2 —> Switch 3)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	1	<2, 4>				BC_YES
Switch 4	4	3	<1, 3>			BC_YES	



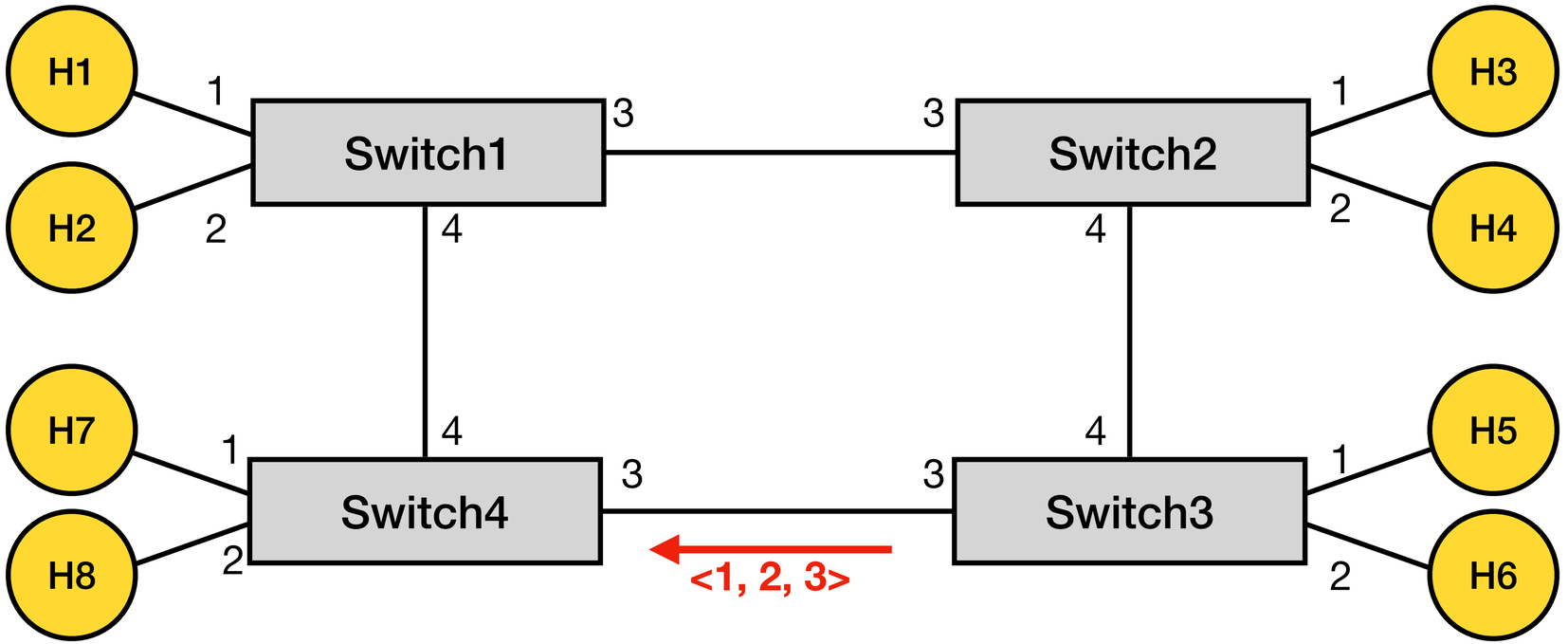
A Running Example (Switch 3 —> Switch 4)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	1	<2, 4>				BC_YES
Switch 4	4	3	<1, 3>			BC_YES	



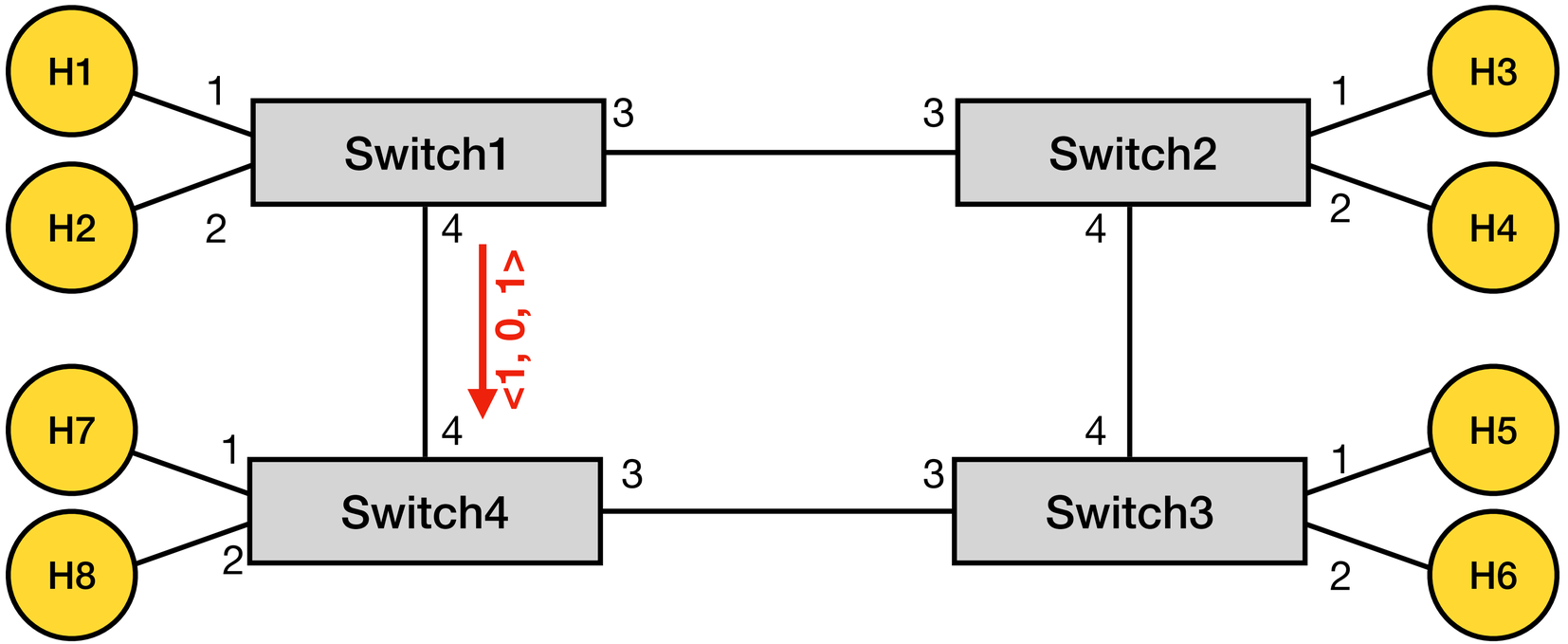
A Running Example (Switch 3 —> Switch 4)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	1	<2, 4>				BC_YES
Switch 4	4	1	<3, 3>			BC_YES	



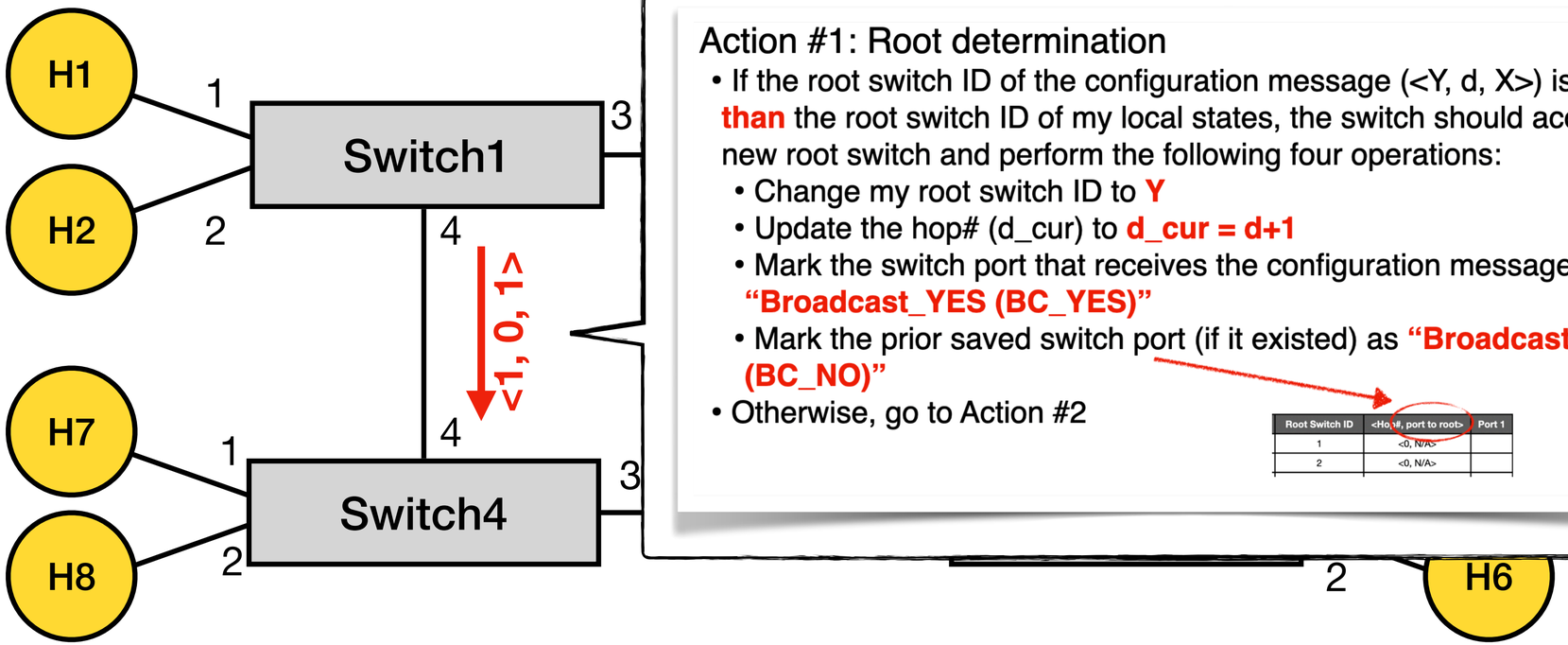
A Running Example (Switch 1 —> Switch 4)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	1	<2, 4>				BC_YES
Switch 4	4	1	<3, 3>			BC_YES	



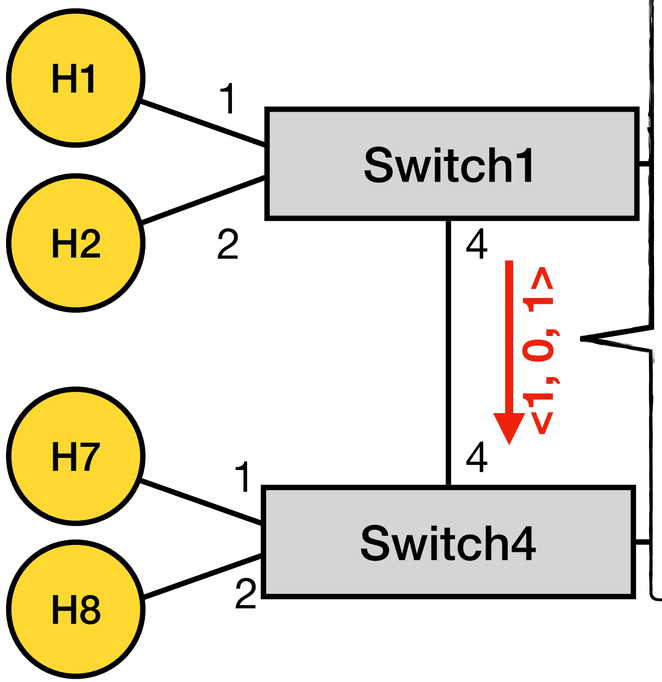
A Running Example (Switch 1 —> Switch 4)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	1	<2, 4>				BC_YES
Switch 4	4	1					



A Running Example (Switch 1 —> Switch 4)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	1	<2, 4>				BC_YES
Switch 4	4	1					

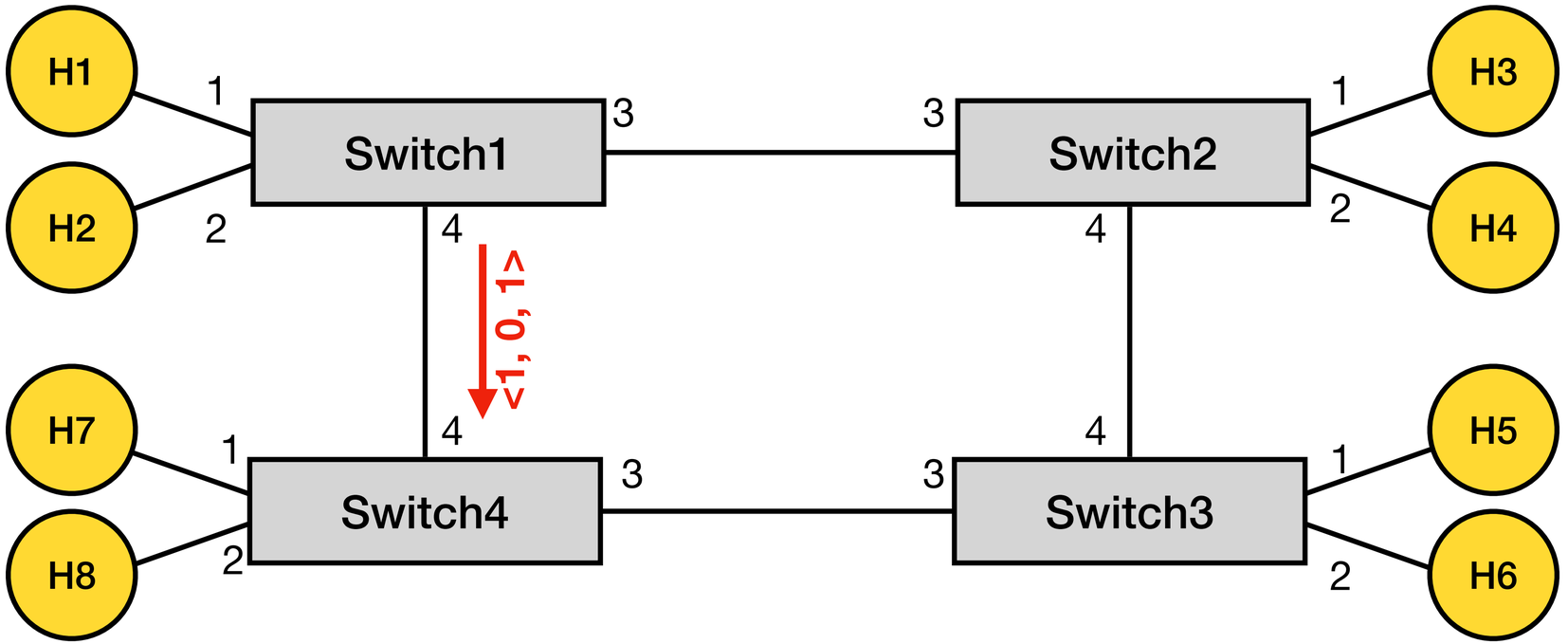


Action #2: Path determination

- Action #2: Path determination
  - If the root switch ID of the configuration message (<Y, d, X>) is **the same as** the root switch ID of my local states, the switch should examine the hop# to figure out the shortest path and perform the following operations:
    - If **d+1 < d_cur**, the switch should (a) update the hop# (d_cur) to **d_cur = d+1**; (b) mark the switch port that receives the configuration message as **"Broadcast_YES (BC_YES)"**; (c) mark the prior saved switch port (if it existed) as **"Broadcast_NO (BC_NO)"**;
    - If **d+1 >= d_cur**, the switch should (a) discard the message; (b) mark the switch port that receives the configuration message as **"Broadcast_NO (BC_NO)"**. The root switch skips (b) and marks the port as **"Broadcast_YES (BC_YES)**;
  - Otherwise, go to Action #3

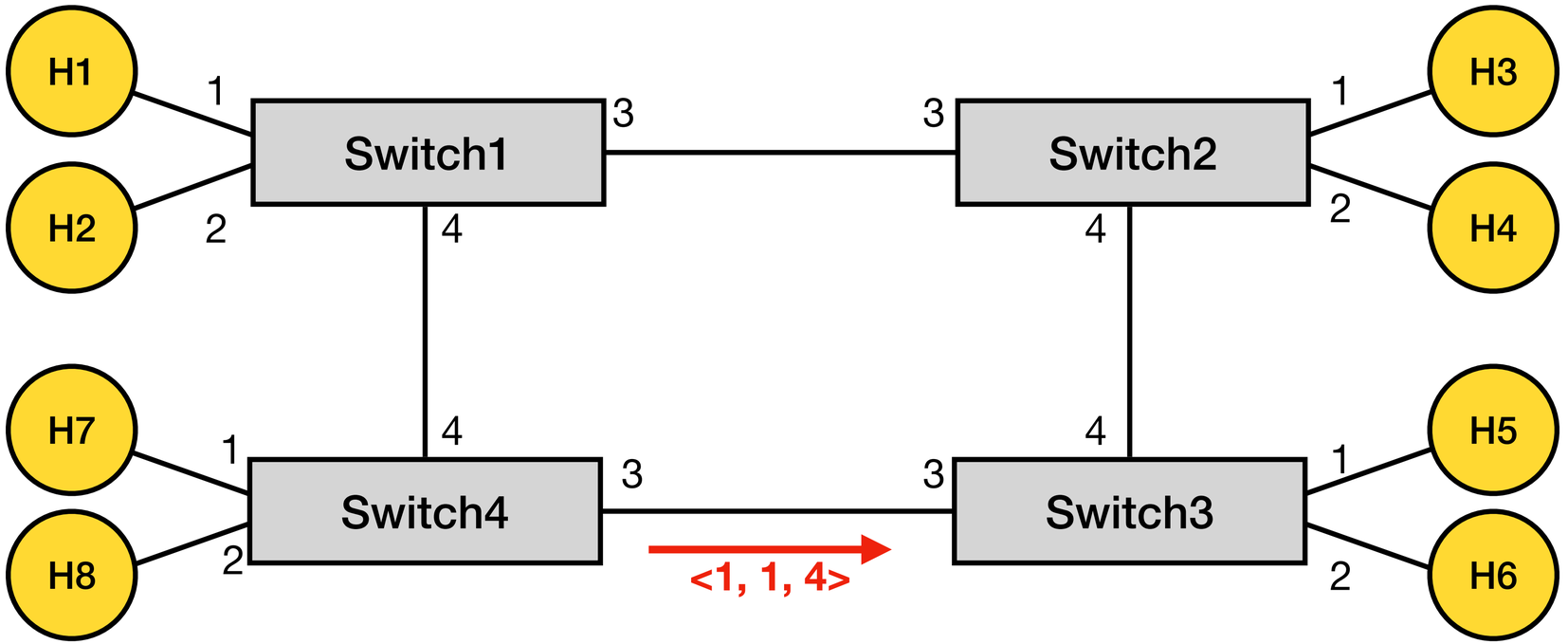
A Running Example (Switch 1 —> Switch 4)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	1	<2, 4>				BC_YES
Switch 4	4	1	<1, 4>			BC_NO	BC_YES



A Running Example (Switch 4 —> Switch 3)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	1	<2, 4>				BC_YES
Switch 4	4	1	<1, 4>			BC_NO	BC_YES



A Running Example (Switch 4 —> Switch 3)

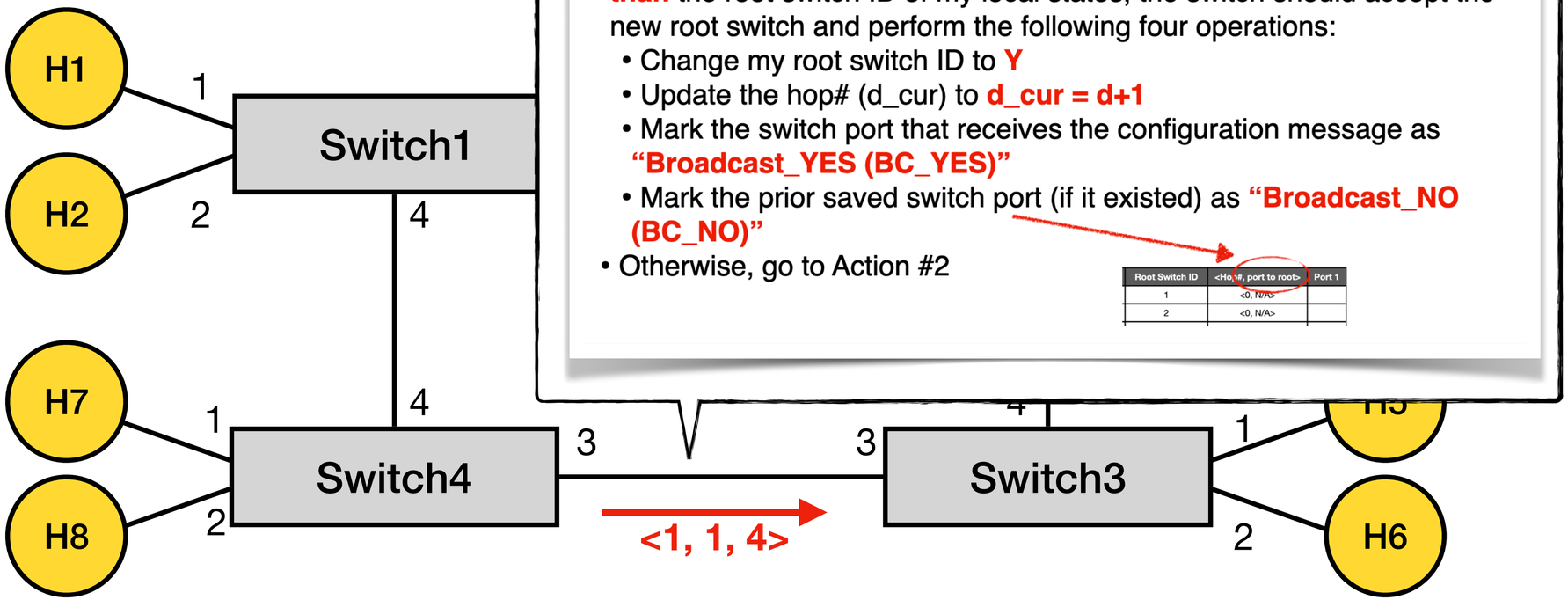
	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1					
Switch 3	3	1					BC_YES
Switch 4	4	1					BC_YES

Action #1: Root Determination

Action #1: Root determination

- If the root switch ID of the configuration message (<Y, d, X>) is **smaller than** the root switch ID of my local states, the switch should accept the new root switch and perform the following four operations:
  - Change my root switch ID to **Y**
  - Update the hop# (d_cur) to **d_cur = d+1**
  - Mark the switch port that receives the configuration message as **"Broadcast_YES (BC_YES)"**
  - Mark the prior saved switch port (if it existed) as **"Broadcast_NO (BC_NO)"**
- Otherwise, go to Action #2

Root Switch ID	<Hop#, port to root>	Port 1
1	<0, N/A>	
2	<0, N/A>	

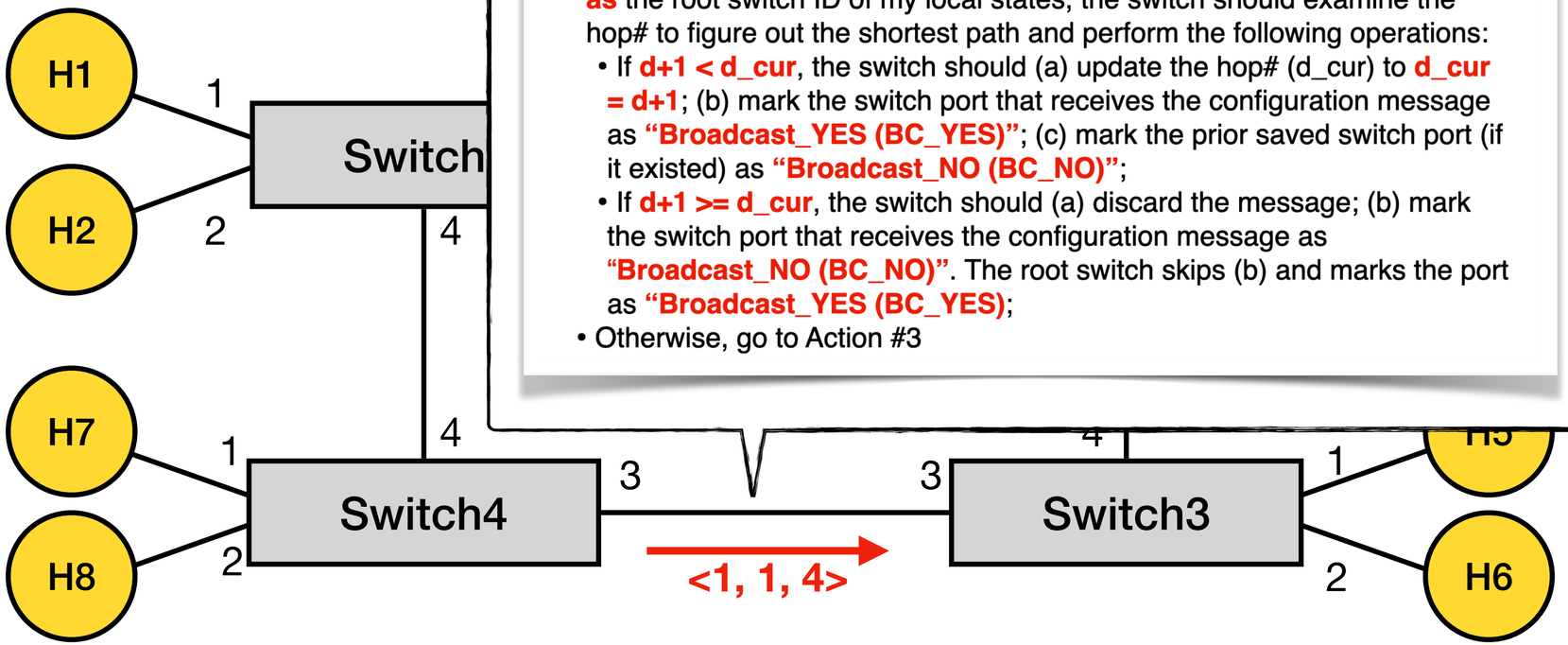


A Running Example (Switch 4 —> Switch 3)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2					ES	
Switch 3	3						BC_YES
Switch 4	4					NO	BC_YES

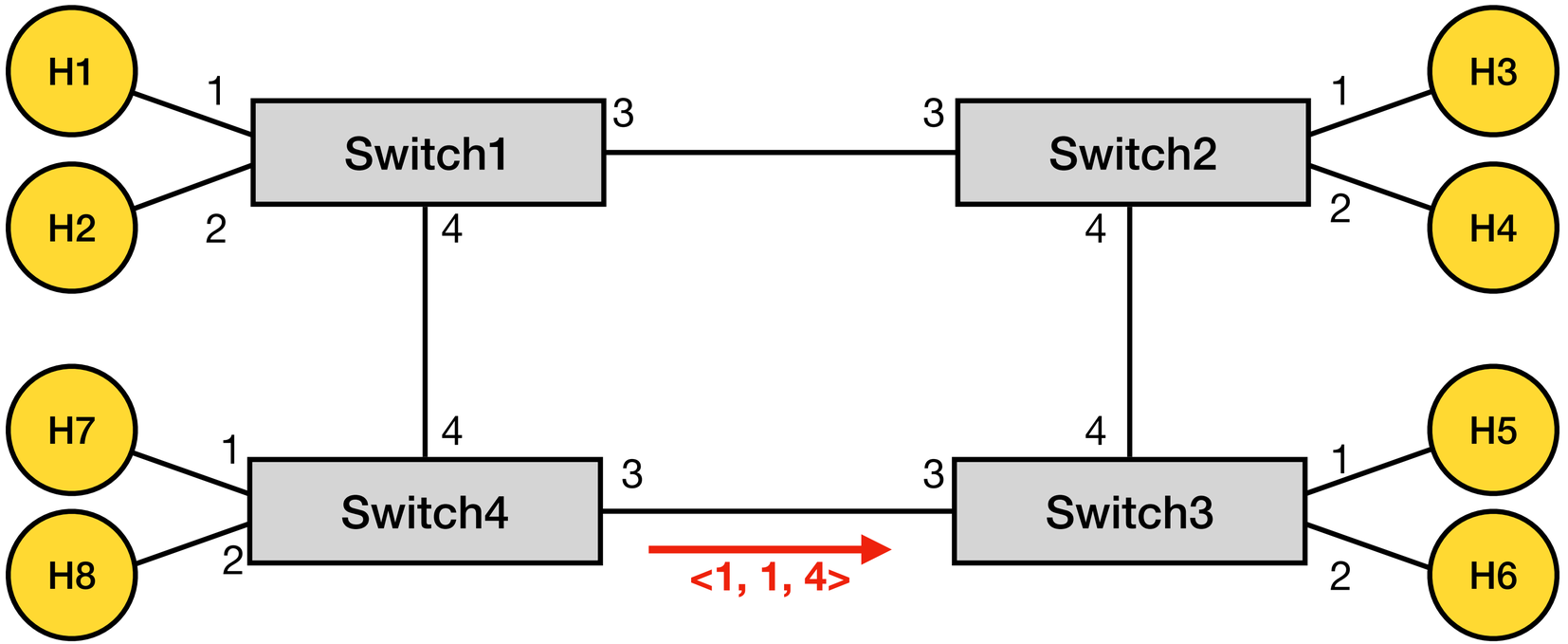
Action #2: Path determination

- Action #2: Path determination
  - If the root switch ID of the configuration message (<Y, d, X>) is **the same as** the root switch ID of my local states, the switch should examine the hop# to figure out the shortest path and perform the following operations:
    - If **d+1 < d_cur**, the switch should (a) update the hop# (d_cur) to **d_cur = d+1**; (b) mark the switch port that receives the configuration message as **"Broadcast_YES (BC_YES)"**; (c) mark the prior saved switch port (if it existed) as **"Broadcast_NO (BC_NO)"**;
    - If **d+1 >= d_cur**, the switch should (a) discard the message; (b) mark the switch port that receives the configuration message as **"Broadcast_NO (BC_NO)"**. The root switch skips (b) and marks the port as **"Broadcast_YES (BC_YES)**;
  - Otherwise, go to Action #3



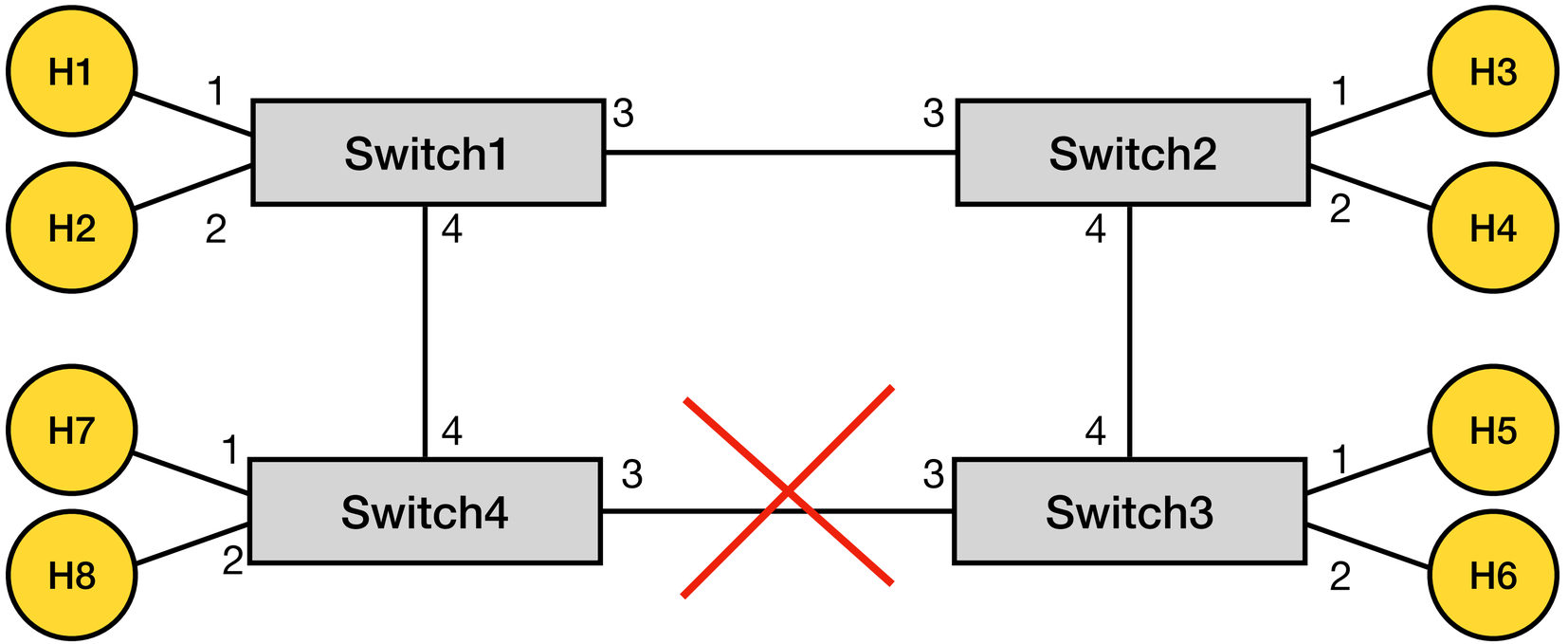
A Running Example (Switch 4 —> Switch 3)

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	1	<2, 4>			BC_NO	BC_YES
Switch 4	4	1	<1, 4>			BC_NO	BC_YES



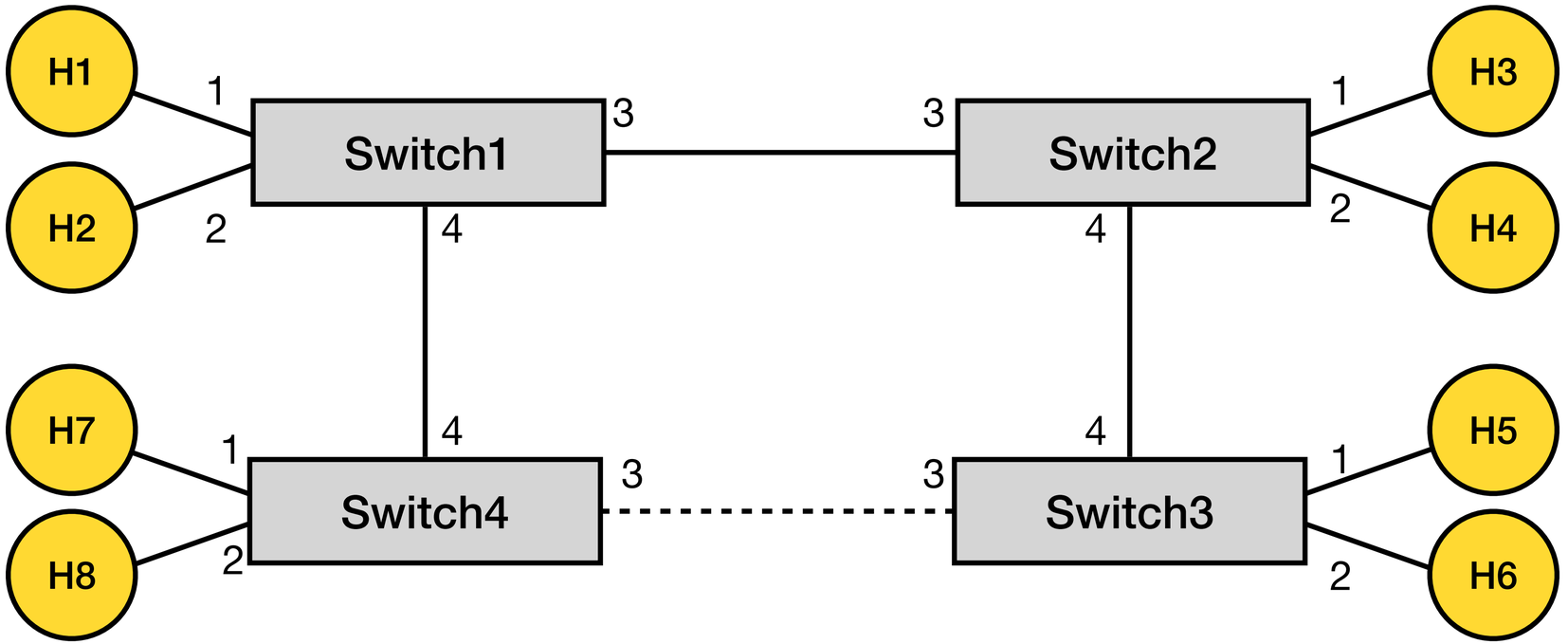
A Running Example: Avoid Loops

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	
Switch 2	2	1	<1, 3>			BC_YES	
Switch 3	3	1	<2, 4>			BC_NO	BC_YES
Switch 4	4	1	<1, 4>			BC_NO	BC_YES



A Running Example: Avoid Loops

	Local Switch ID	Root Switch ID	<Hop#, port to root>	Port 1	Port 2	Port 3	Port4
Switch 1	1	1	<0, N/A>			BC_YES	BC_YES
Switch 2	2	1	<1, 3>			BC_YES	BC_YES
Switch 3	3	1	<2, 4>			BC_NO	BC_YES
Switch 4	4	1	<1, 4>			BC_NO	BC_YES



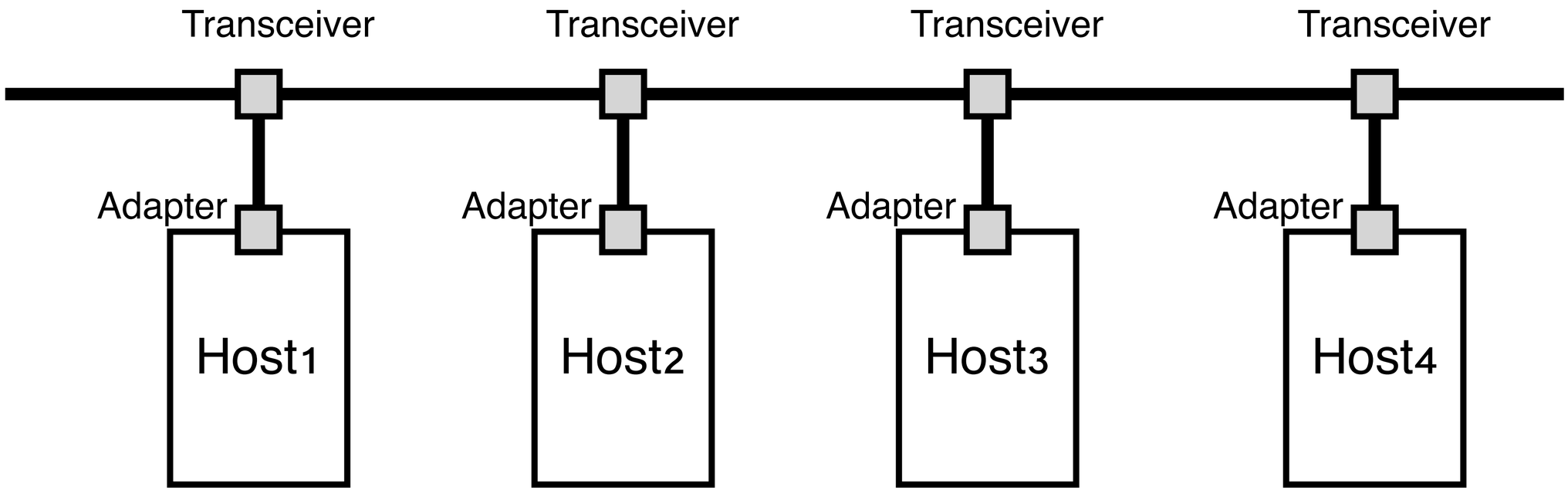
Ethernet Dominates Local Area Network (LAN)

- LAN connects computers within a limited area
  - Popular in residence, campus, and building
  - Great competition in the 1980s, e.g., token ring, FDDI, ATM, Ethernet

- Why Ethernet succeeds
  - #1: The first widely deployed high-speed LAN
  - #2: Simple and cheap
  - #3: Sustain a higher data rate for each version
  - #4: Ethernet hardware becomes commodity

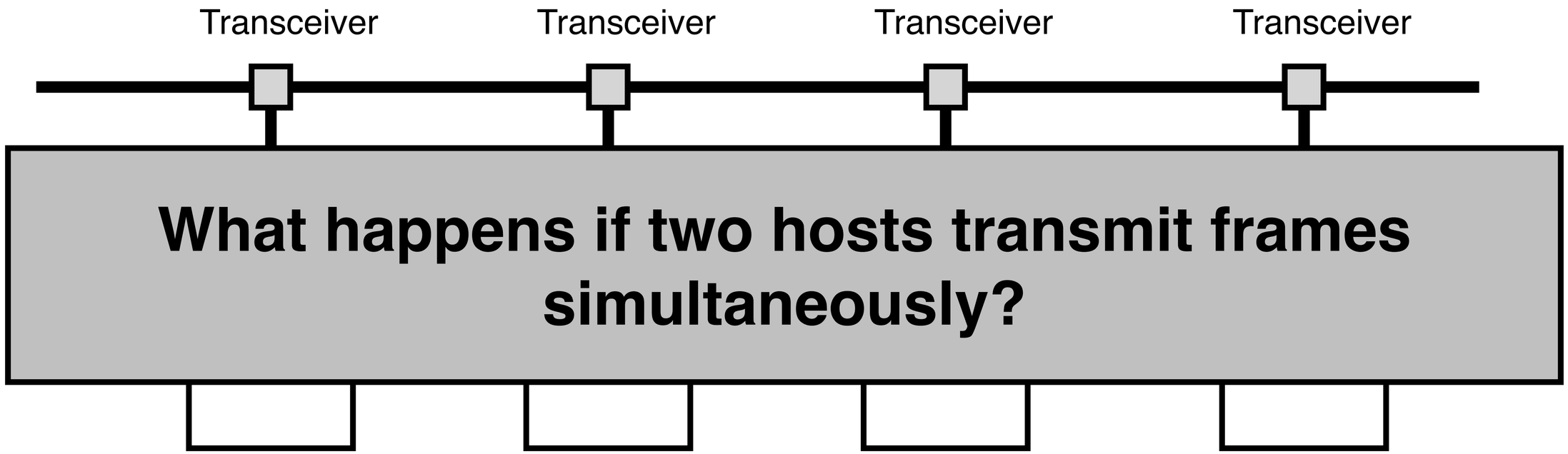
Ethernet (1970s to mid-1990s)

- Design for multiple access over a shared physical medium
  - Bus topology
  - All transmitted frames travel to and processed by all adapters connected to the bus



Ethernet (1970s to mid-1990s)

- Design for multiple access over a shared physical medium
  - Bus topology
  - All transmitted frames travel to and processed by all adapters connected to the bus



CSMA/CD Overview

- Key idea: access the channel in a “random” fashion
  - When collisions occur, wait for a random interval and retransmit

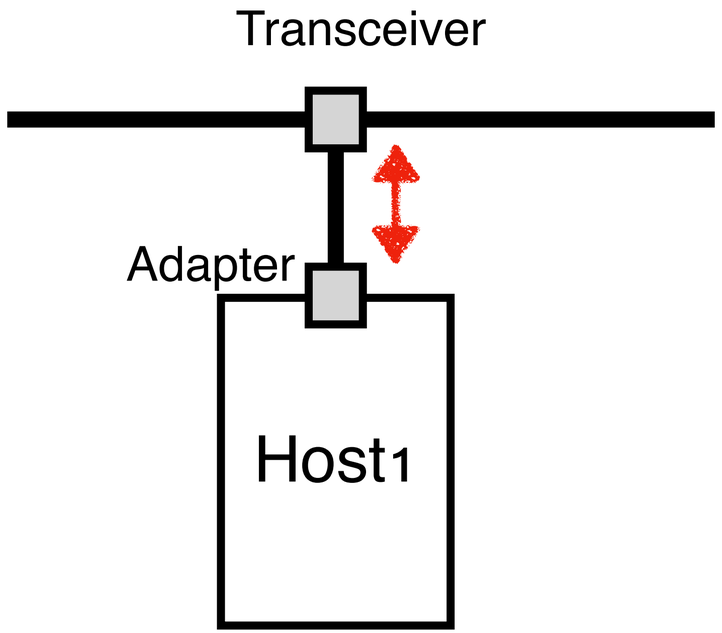
CSMA/CD Overview

- Key idea: access the channel in a “random” fashion
  - When collisions occur, wait for a random interval and retransmit

- CSMA/CD Scheme:
  - #1: Carrier Sense (CS)
  - #2: Multiple Access (MA)
  - #3: Collision Detection (CD)

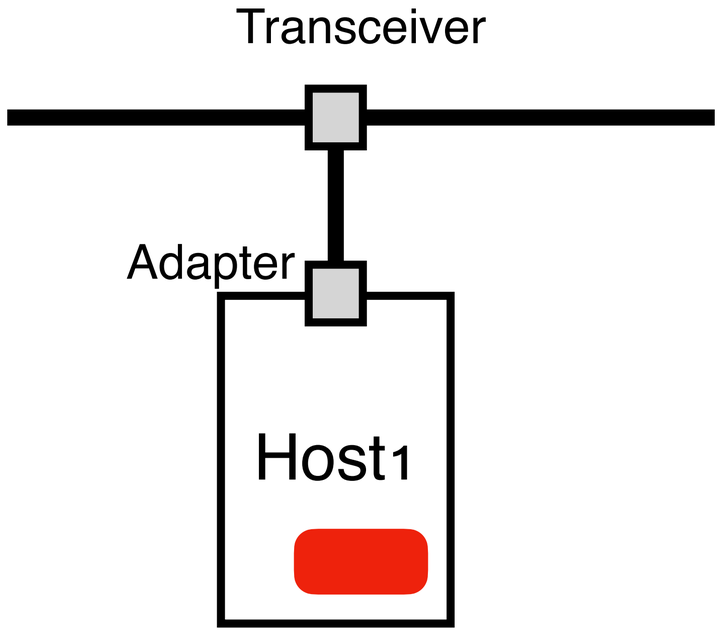
#1: Carrier Sense (CS)

- The transmitter senses the state of the communication carrier
  - Idle or busy
  - We use “transmitter” here to describe the host that sends a frame



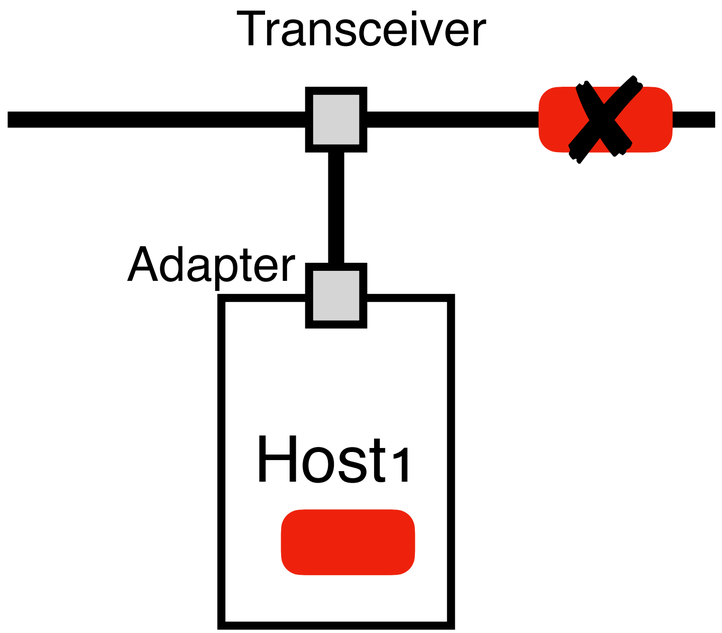
#2: Multiple Access (MA)

- The transmitter sends the frame with a probability p when idle
  - No coordination
  - The device makes the decision at the beginning of each time slot
  - p is pre-determined



#3: Collision Detection (CD)

- Under collision, the transmitter aborts the communication, performs exponential back-off, and sends the frame again
  - Postpone the transmission by an interval of *T*
  - The length of the interval *T* increases with every collision
    - T = 2 ^ (i-1) * X, where *i* is the number of retries
  - X was configured as 51.2us originally



Early Ethernet Standards

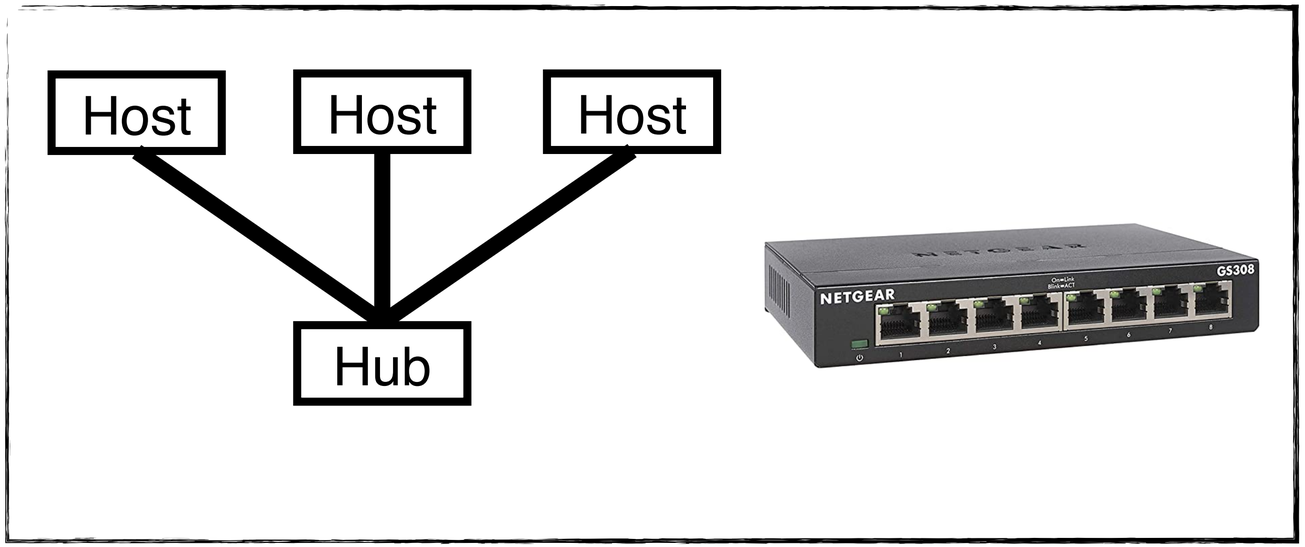
- Ethernet defines both the physical and link layers
  - 10BASE-T, 10BASE-2, 10BASE-5

- Naming
  - #1: “10” —> data rate: 10Mbps
  - #2: BASE —> Baseband Ethernet, i.e., the physical layer only carries Ethernet traffic
  - #3: Physical media itself
    - T —> Twisted-pair copper wires
    - 2 —> Thin coaxial cables
    - 5 —> Thick coaxial cables

Ethernet (mid-1990s to late-1990s)

- Ethernet is widely deployed at companies and universities
  - Bus topology —> Hub-based star topology
  - An Ethernet segment originally could support 500m at most
  - CSMA/CD is still needed

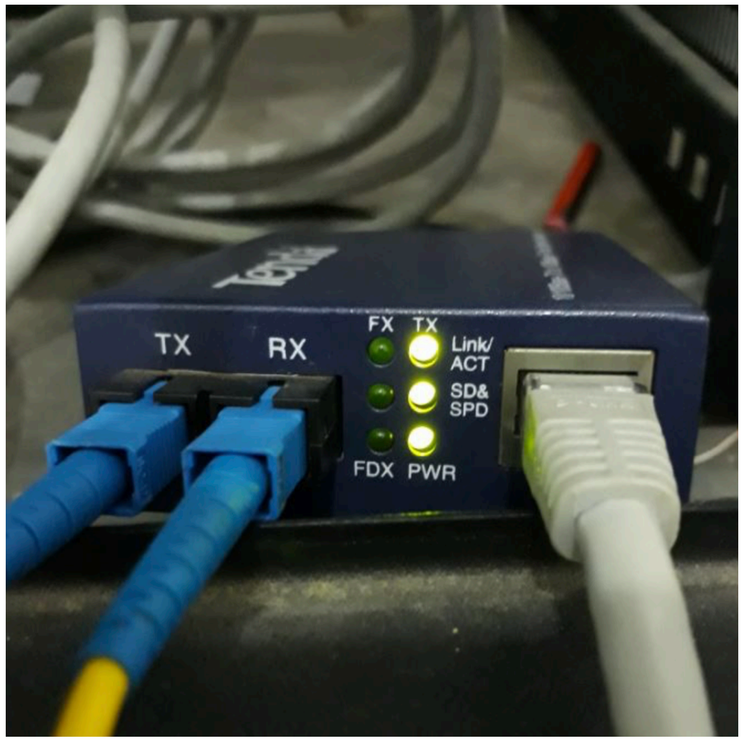
Network repeater



Network Hub

Ethernet Standards Envolving

- 100Mbps becomes widely used
  - Twisted-pair copper wires (100BASE-T)
  - Fiber optical (100BASE-FX, 100BASE-SX, 100BASE-BX)



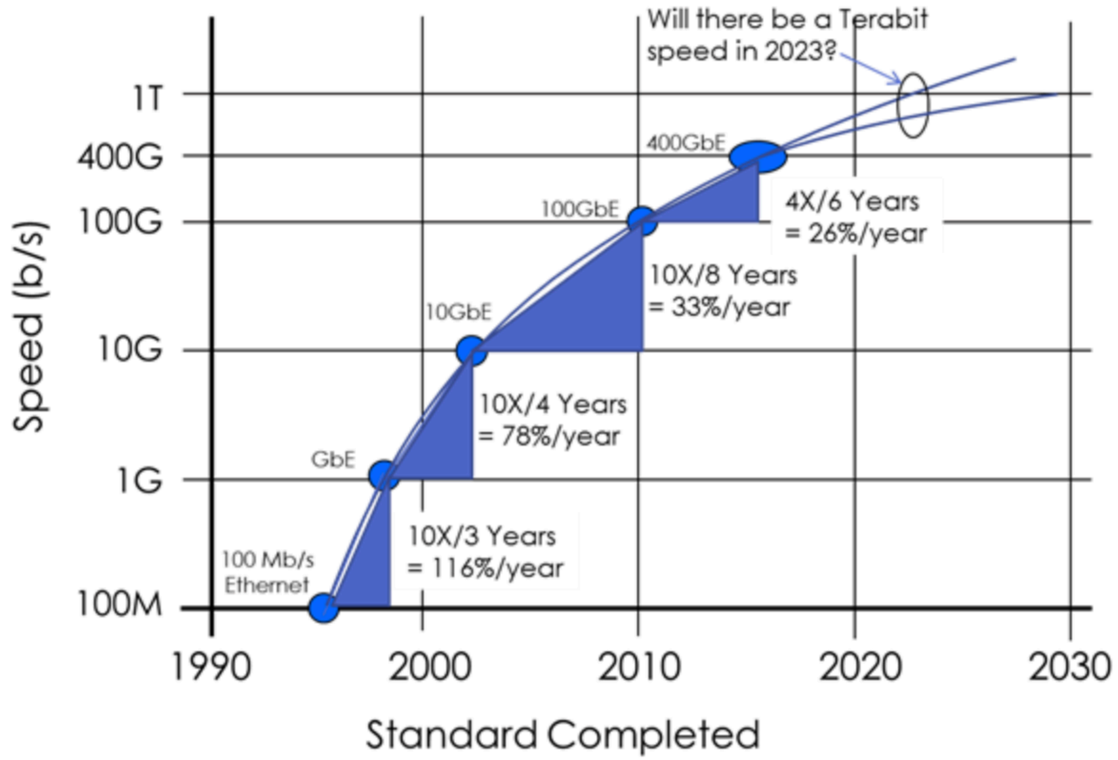
Ethernet (Since 2000s)

- Gigabit Ethernet
  - Ethernet switches are used and dominated
  - Full-duplex (send and receive happen simultaneously) device is common

- More importantly, CSMA/CD is no longer needed
  - Hub-based star topology —> Switch-based star topology
  - Point-to-point communication —> Isolated communication domain
  - Full-duplex mode —> No send-and-receive interference

Rising Ethernet Data Rate

- 800GBASE is standardized
  - QSFP-DD and QSFP-112 transceivers are available
  - 1.6 Terabit Ethernet is under development



Summary

- Today
  - STP
  - Ethernet

- Next lecture
  - Reliable communication at the link layer