Routing, Addressing, BGP

CS 168 – Fall 2022 – Discussion 5

Agenda

- L2 Routing
- Addressing
- BGP

Recall…

- What makes a routing state valid?
- For *every* destination:
  - If all paths to that destination have:
    - No dead-ends (except destination)
    - No loops
- Then the state is valid

Learning Switches

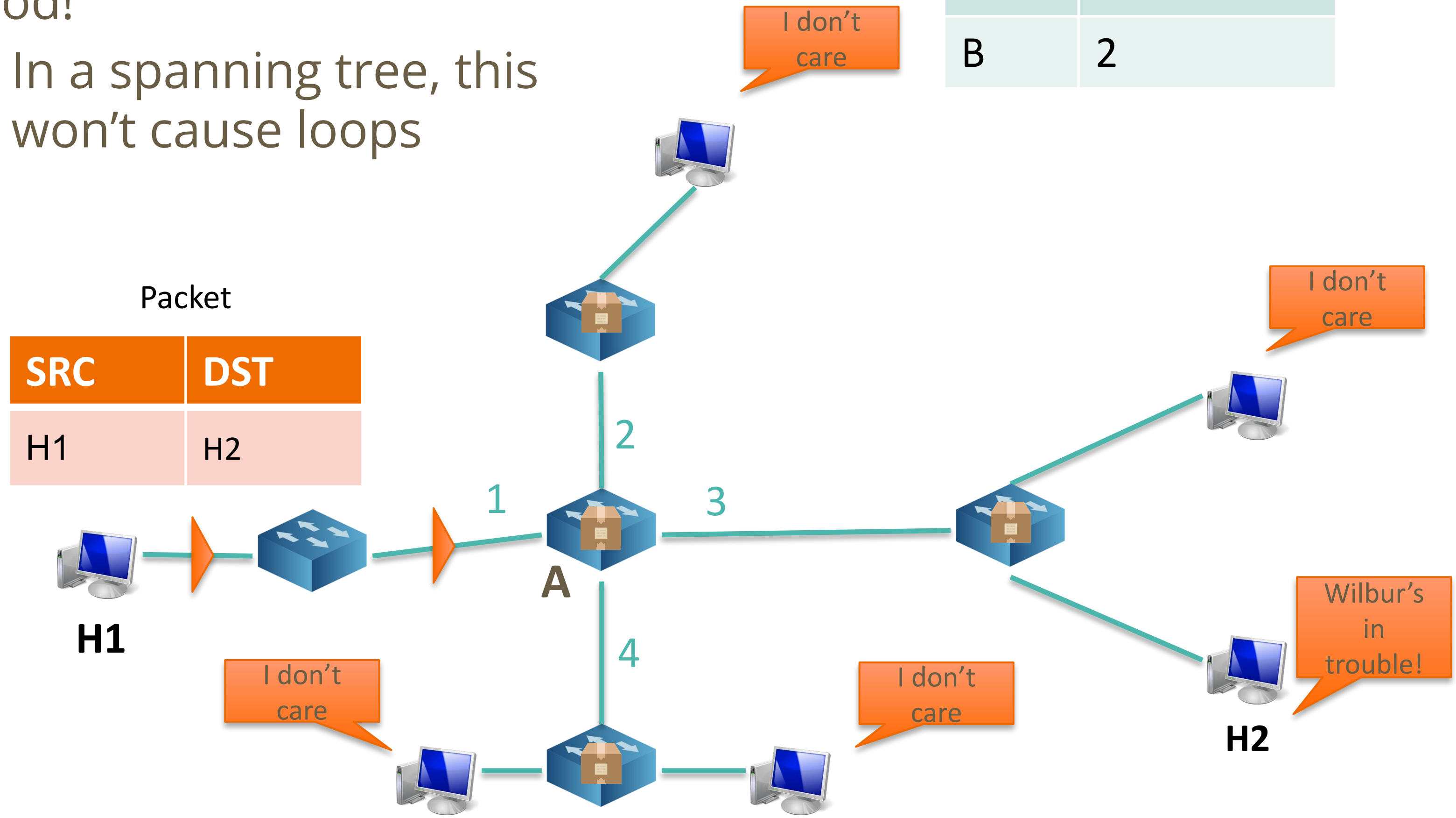
- What if the dest is not found?
  - Flood!
    - In a spanning tree, this won't cause loops

A's routing table

DEST	OUTGOING LINK
D	3
B	2

Packet

SRC	DST
H1	H2



Learning Switches

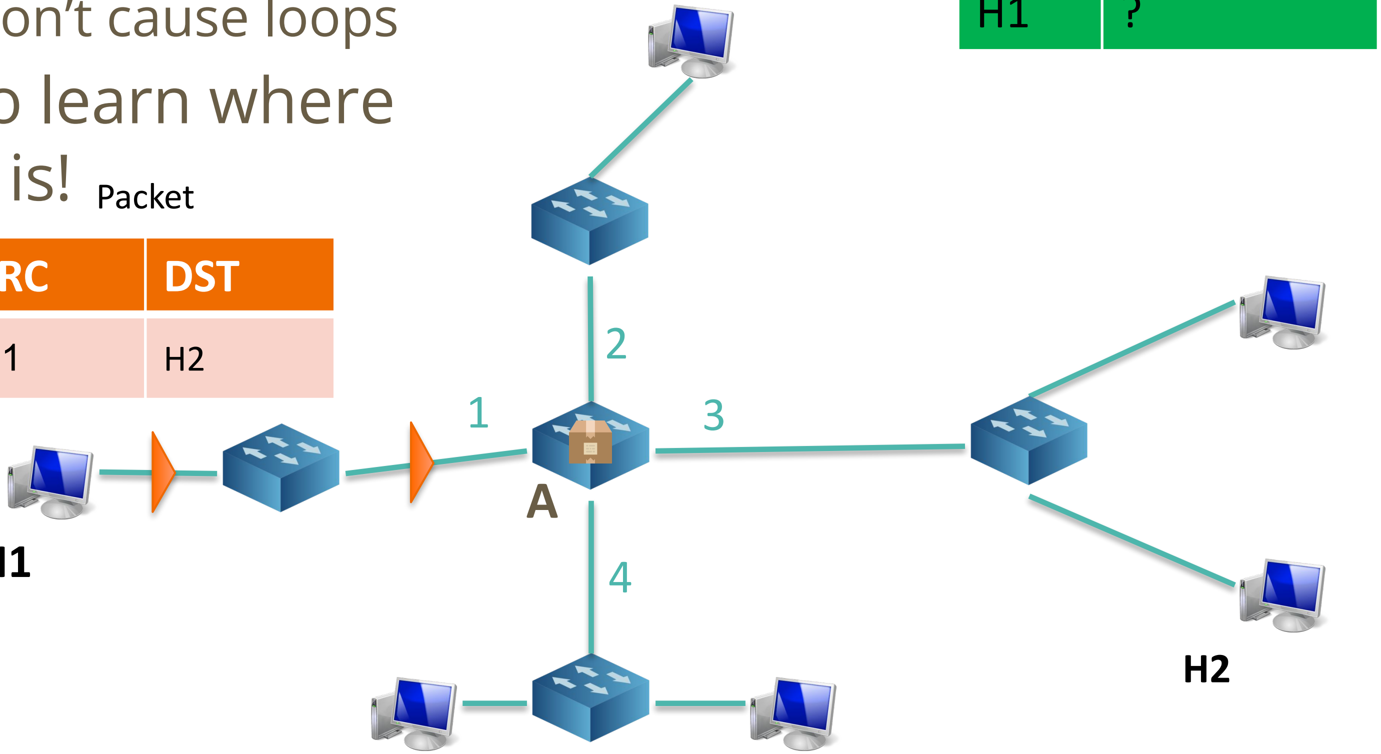
- What if the dest is not found?
  - Flood!
    - In a spanning tree, this won't cause loops
- We also learn where the src is!

A's routing table

DEST	OUTGOING LINK
D	3
B	2
H1	?

Packet

SRC	DST
H1	H2



Learning Switches

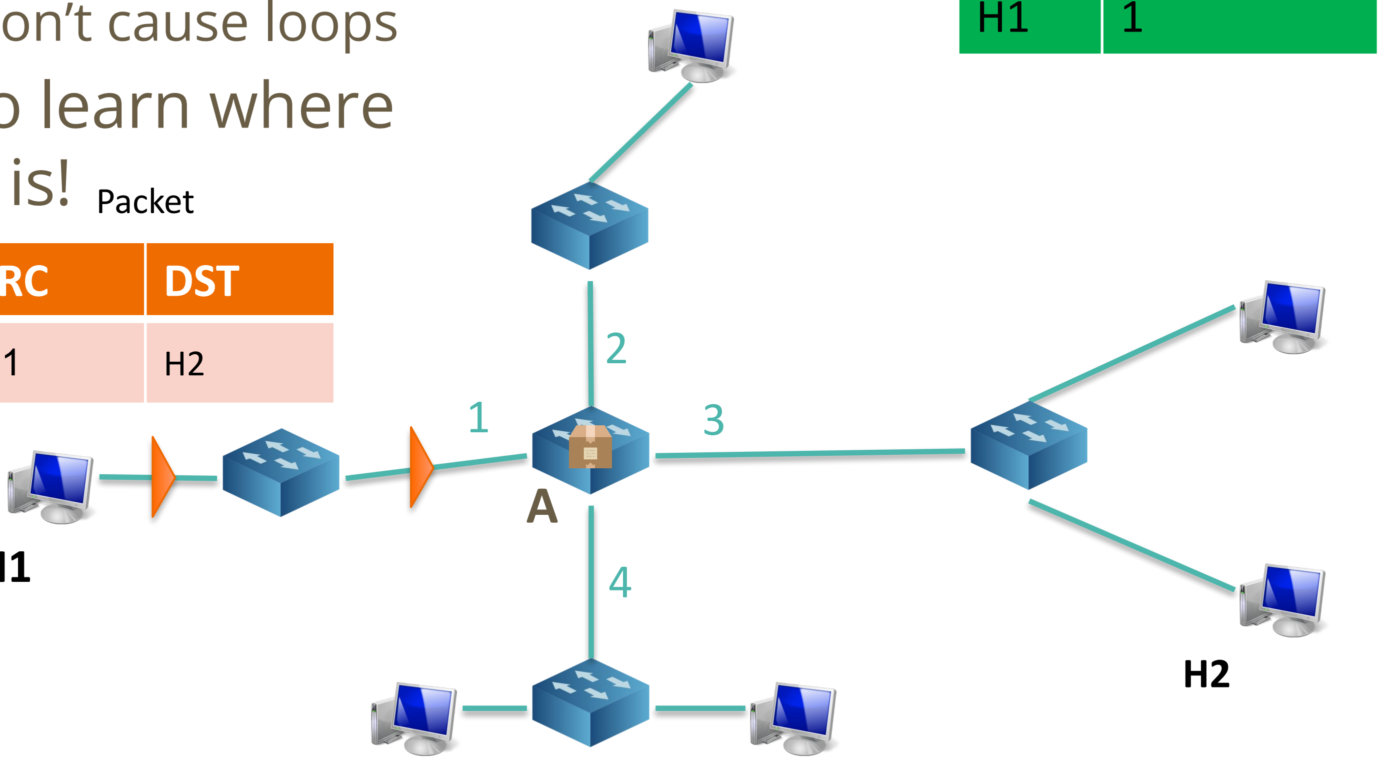
- What if the dest is not found?
  - Flood!
    - In a spanning tree, this won't cause loops
- We also learn where the src is!

A's routing table

DEST	OUTGOING LINK
D	3
B	2
H1	1

Packet

SRC	DST
H1	H2



Learning Switch: Pseudocode

When a learning switch receives a packet:

```
Look up the DEST in the switch table

if entry found for destination {
    if dest on link from which packet arrived
        then drop packet
    else forward packet on link indicated
}

else flood
```

Learning Switch: Pseudocode

When a learning switch receives a packet:

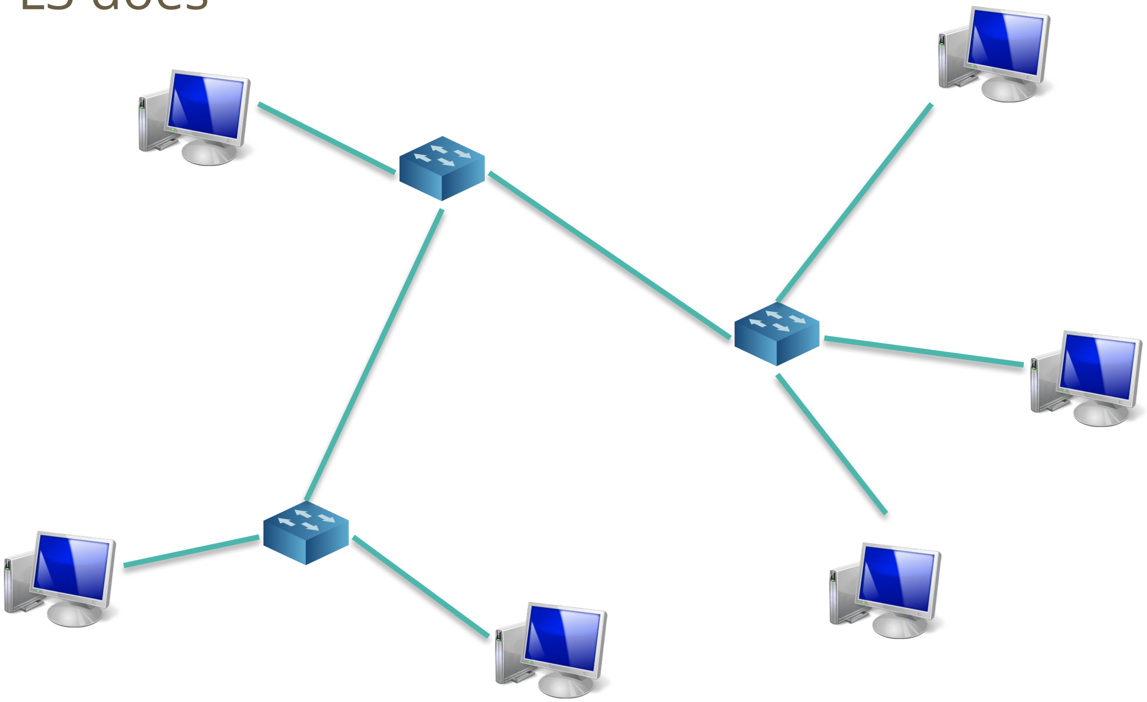
```
if dest not in switch_table:
    flood(packet)

link = switch_table[dest]
if link == incoming_link:
    drop(packet)
else:
    forward(packet, link)
```

Worksheet: Q1

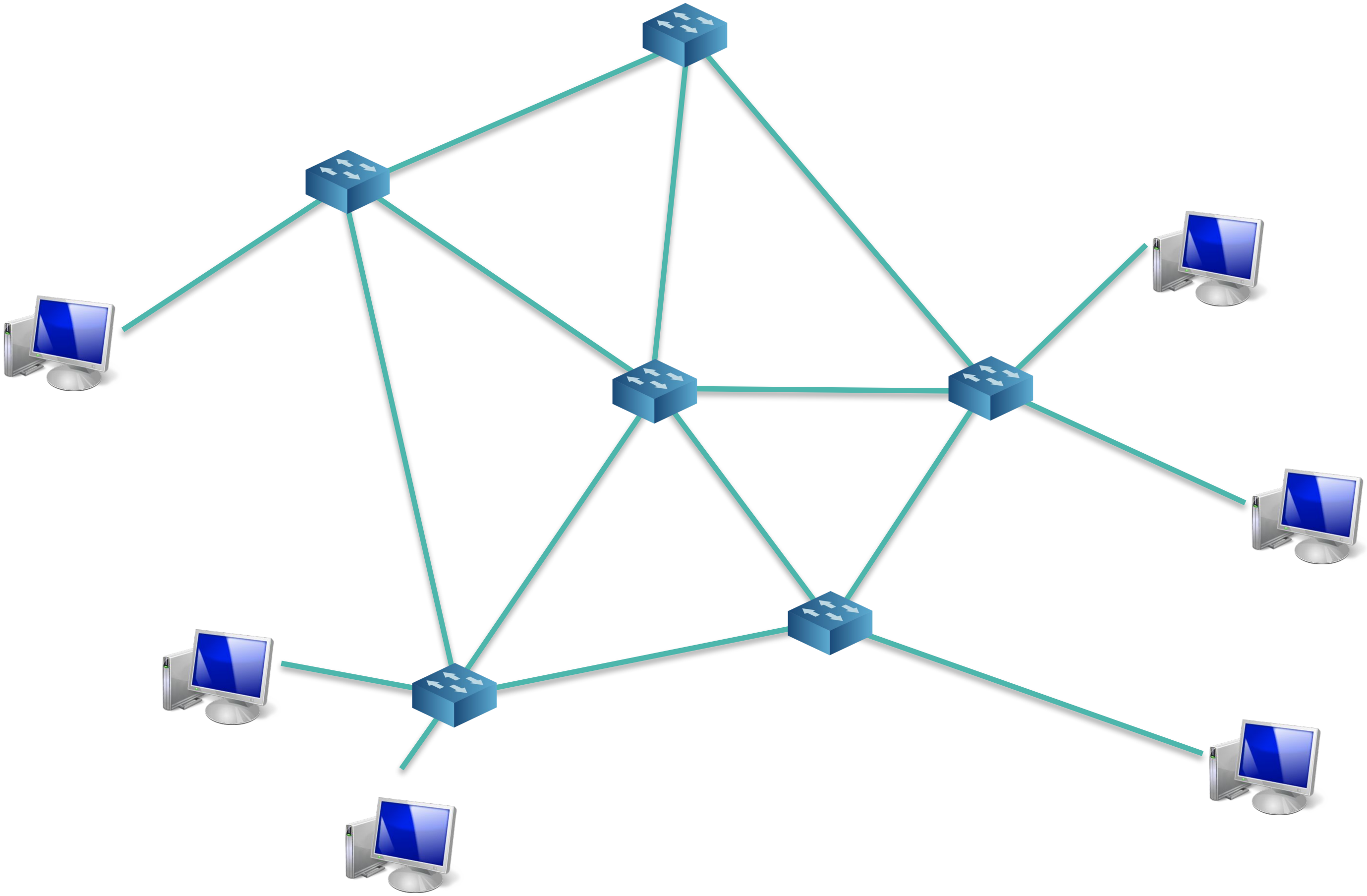
Spanning Trees

- One way to solve loops:
  – Use a topology that doesn't have any!
- Spanning Tree
  – Used by Ethernet networks (L2) which don't require much scalability like L3 does



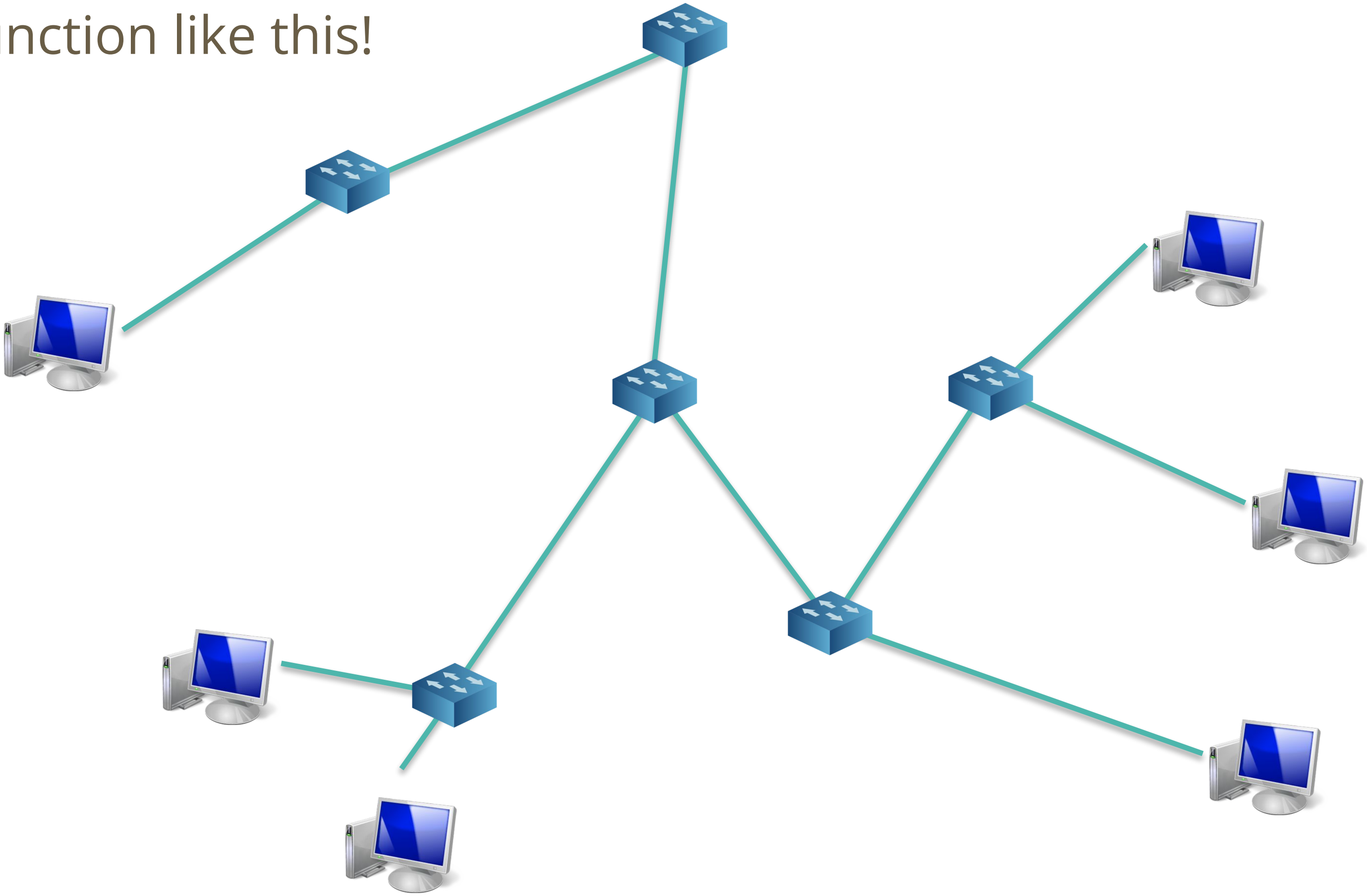
Spanning Tree Protocol (STP)

- Goal:
  - To make this network…



Spanning Tree Protocol (STP)

- Goal:
  - To make this network…
  - Function like this!



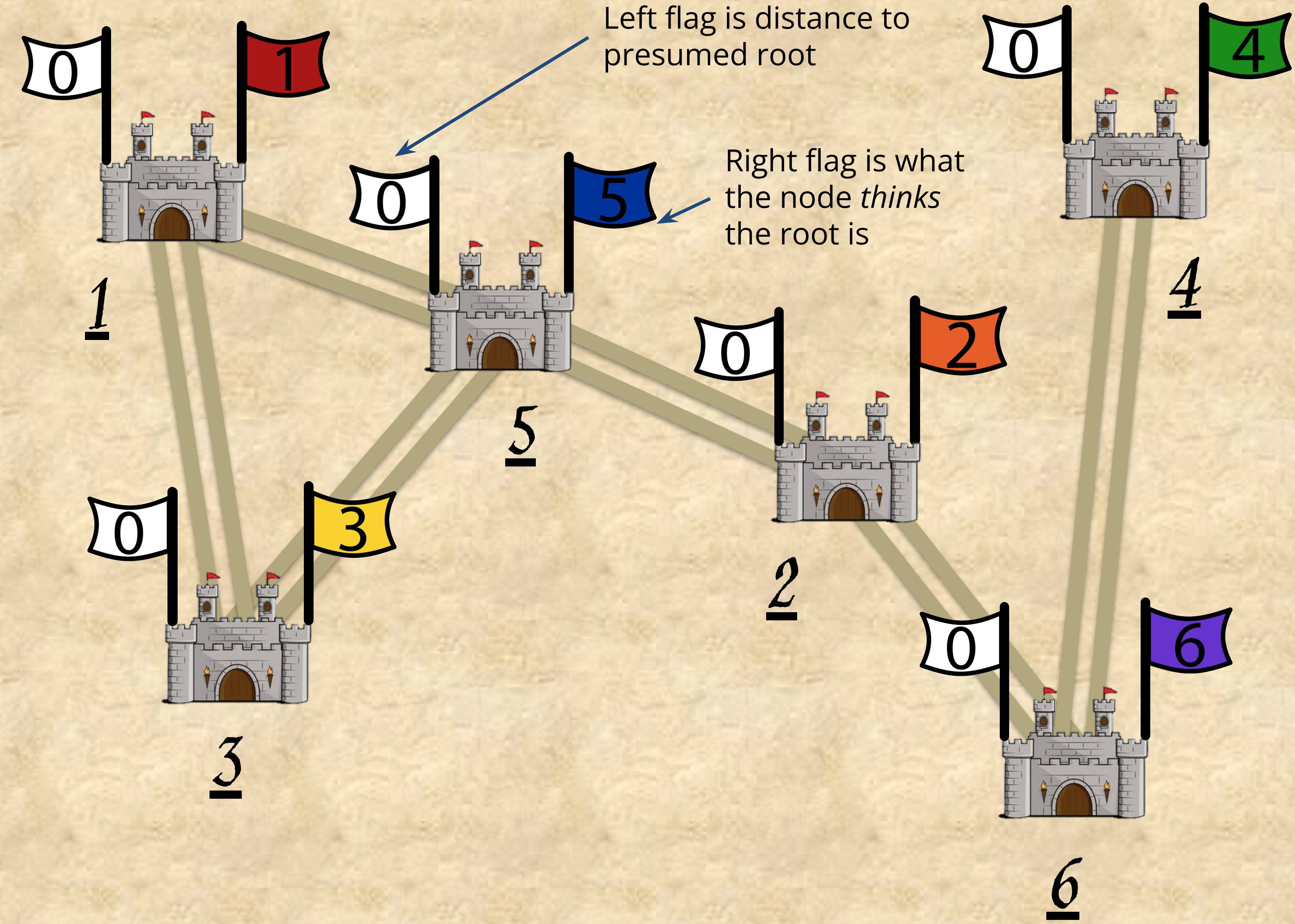
Spanning Tree Protocol (STP)

- Algorithm in two parts:
  1. Pick a root (by lowest address)
  2. Compute shortest paths to that root
     - Only keep the links on the shortest paths
     - Break ties by neighbor with smaller address

Spanning Tree Protocol (STP)

1. Each router sends a message to its neighbors
   - (I think **4** is the root. My distance to 4 is **0**. I am **4**.)
     - Root, Distance, Src
2. When routers receive a message, they update their current knowledge of the root and distance to it.
   - If 4 receives (**3**, **0**, **3**):
     - It will now think (**3** is the root. My distance is **1**. I am **4**.)
3. Repeat until everyone agrees.

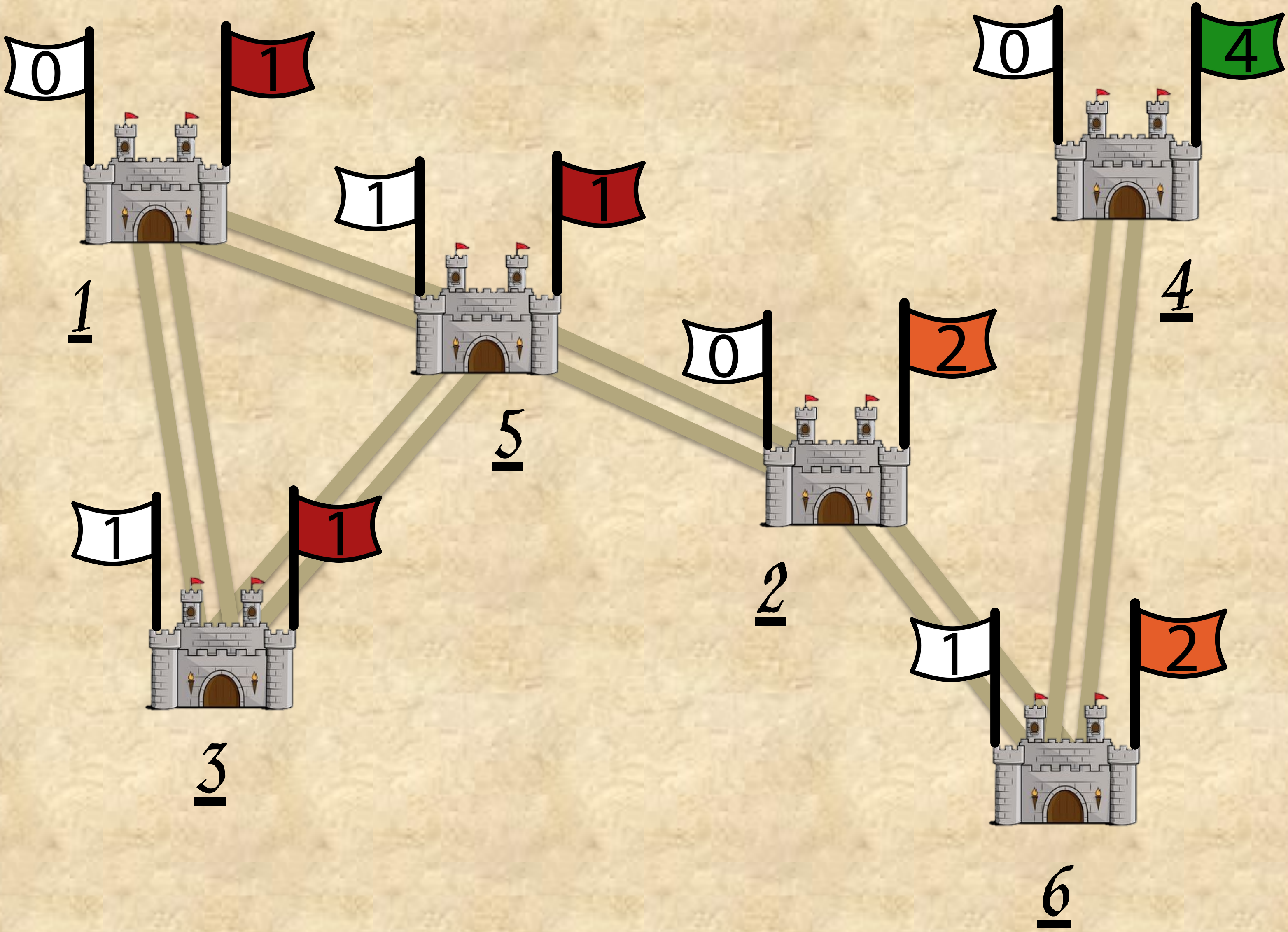
The Spanning Tree Battlefield



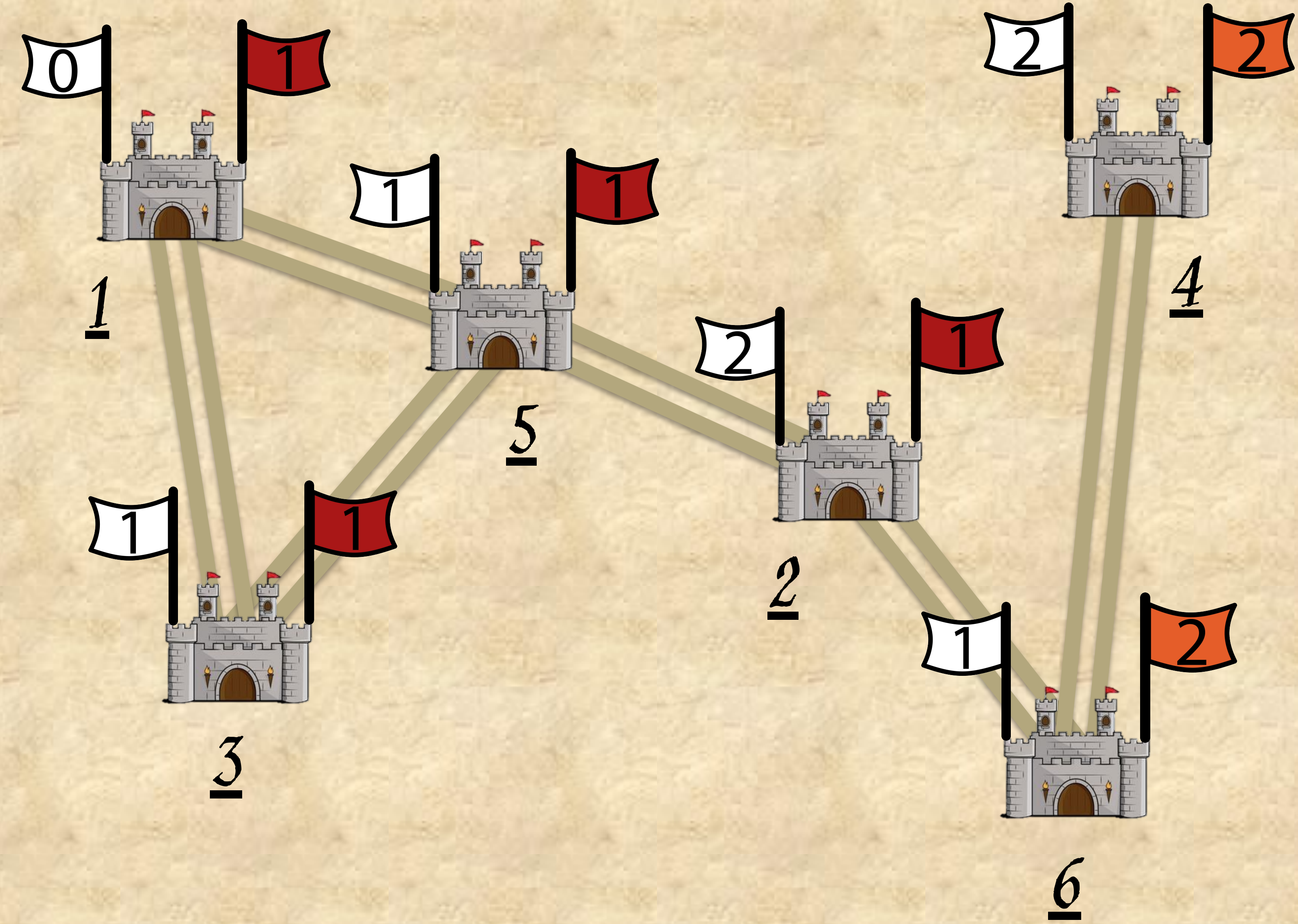
Left flag is distance to presumed root

Right flag is what the node *thinks* the root is

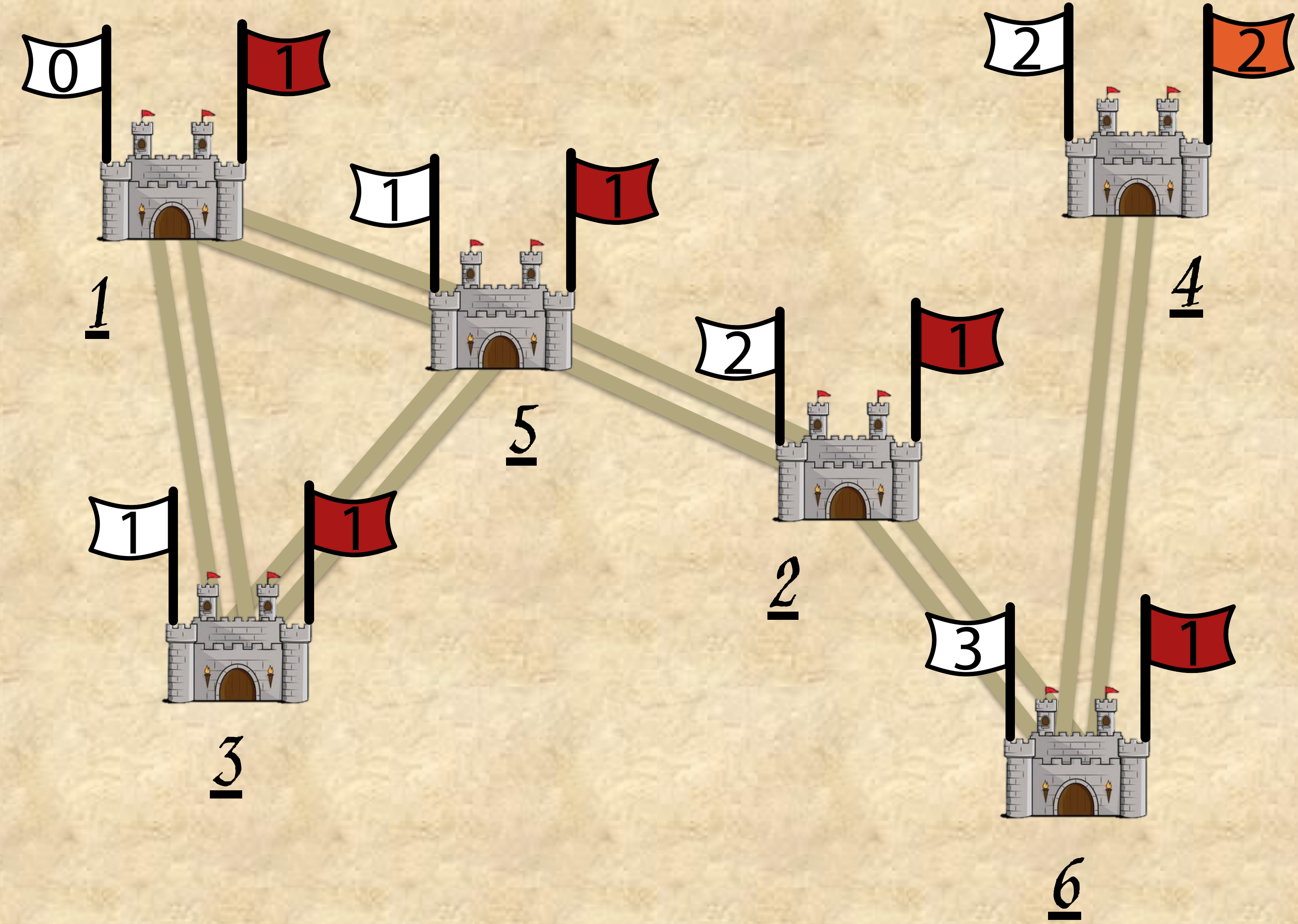
The Spanning Tree Battlefield



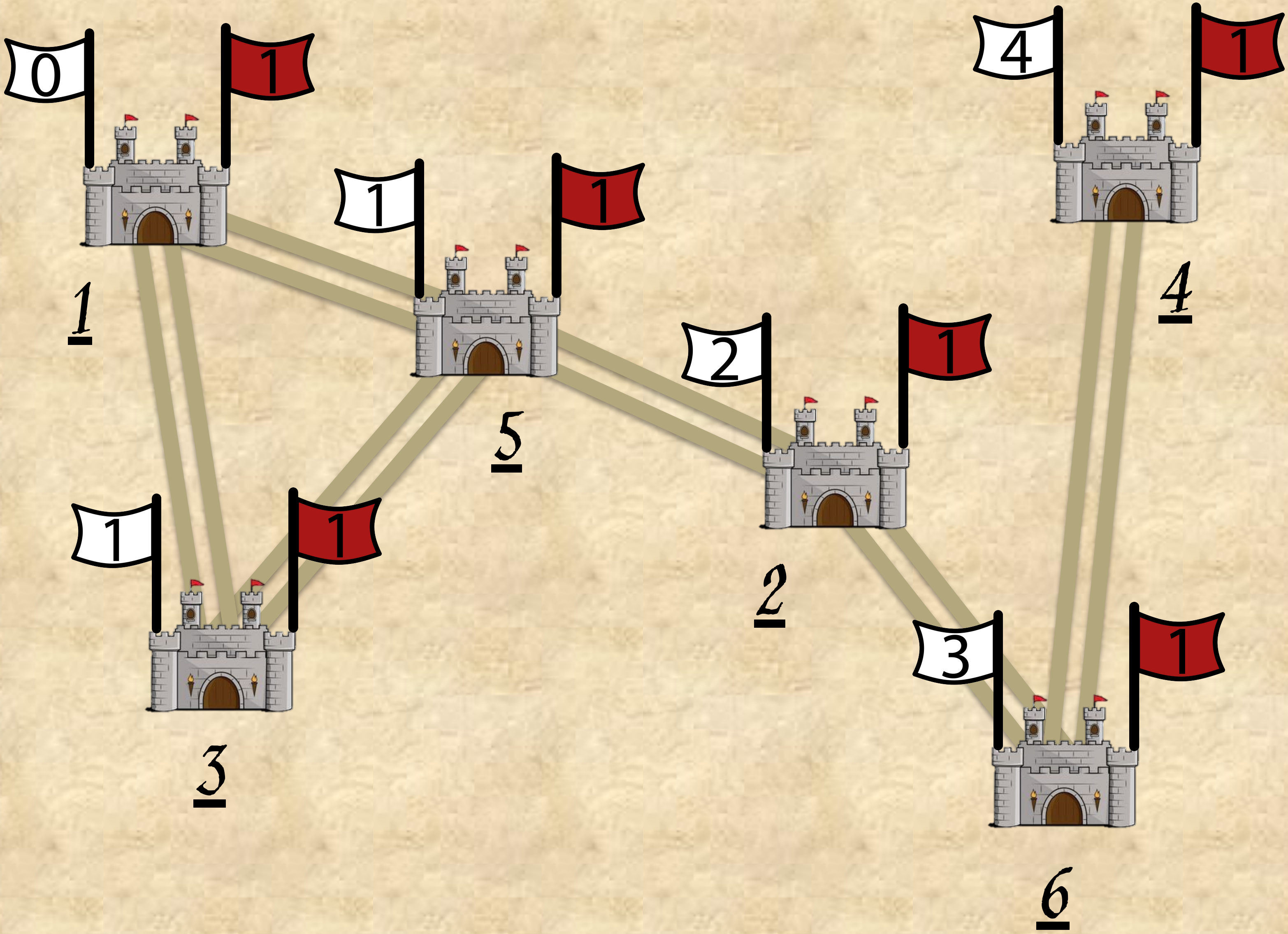
The Spanning Tree Battlefield



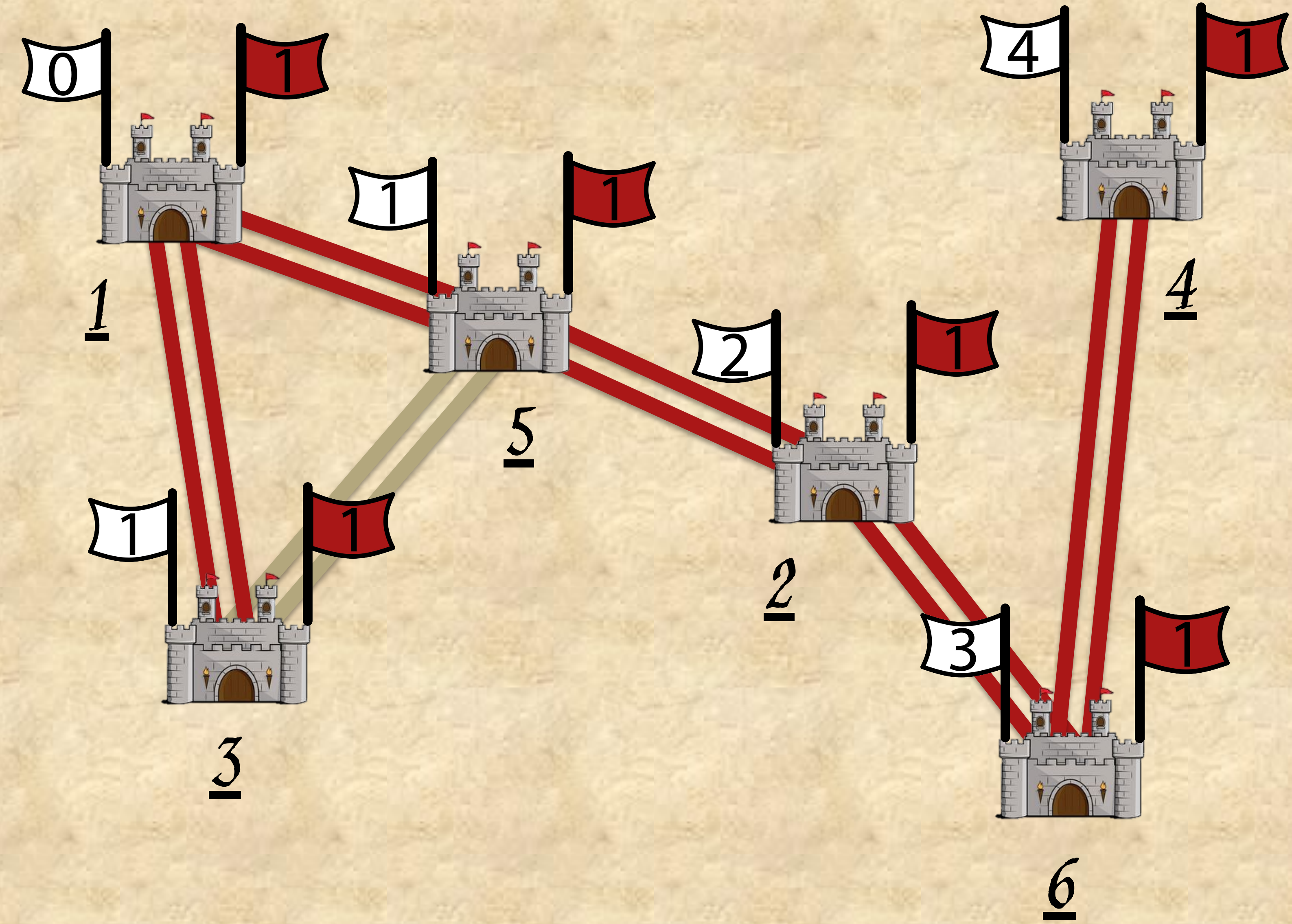
The Spanning Tree Battlefield



The Spanning Tree Battlefield

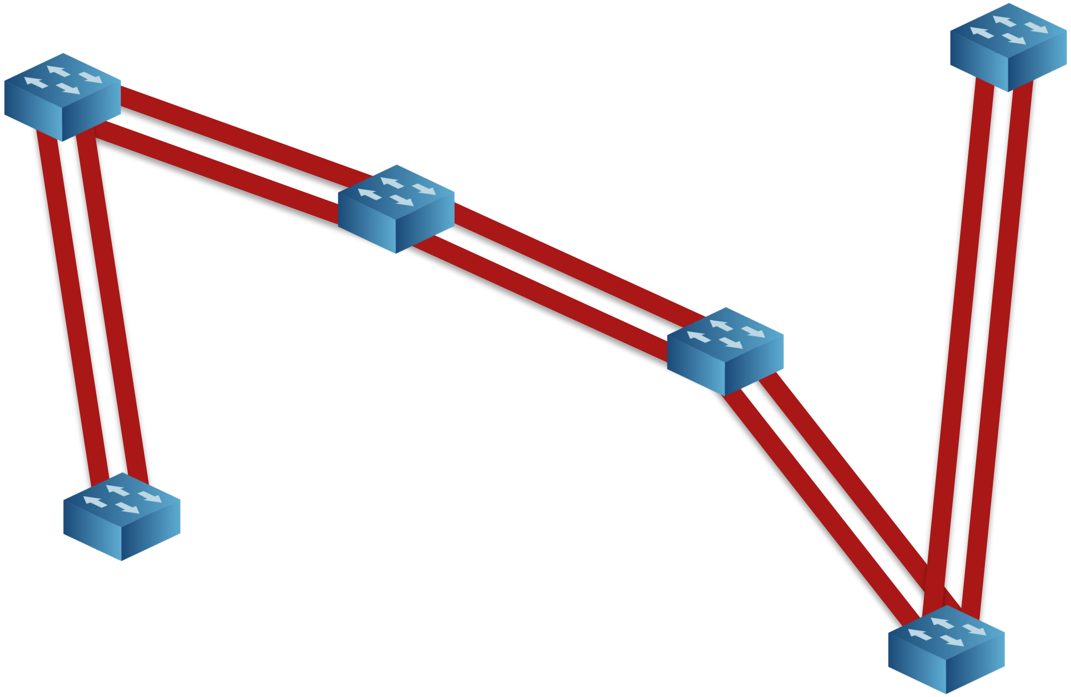


The Spanning Tree Battlefield



The Spanning Tree Battlefield

- Now the links form a spanning tree…
- And we can deploy our learning switches and flood!



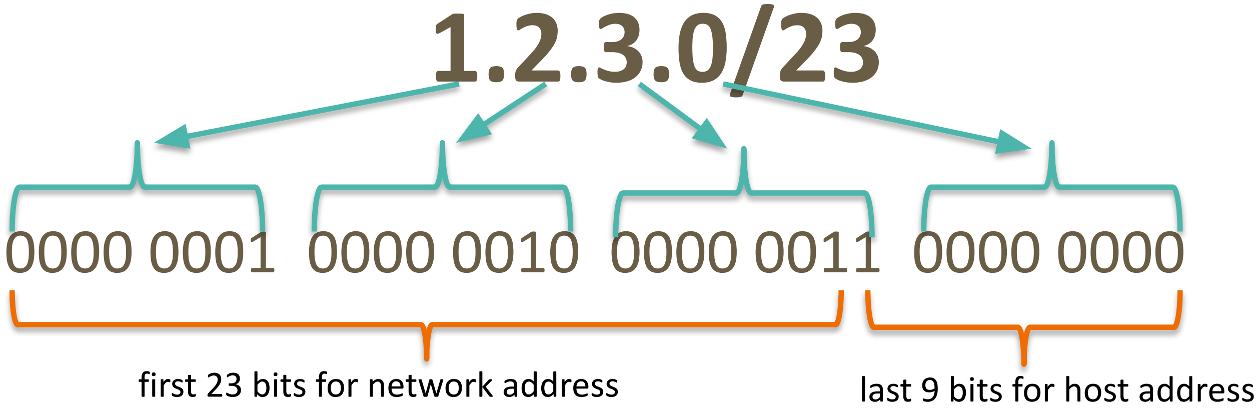
Worksheet: Q2

Requirements of Addressing

- Scalable Routing
  - Minimize state exchange needed to create paths
- Efficient Forwarding
  - Small forwarding tables
  - Fast lookups
- Host must be able to recognize packet is for them
  - An end-to-end check on routing
  - L3: IP addresses (dynamically assigned)

IP Address

- 32 bits (for IPv4), split into 4 bytes, written in decimal
- Network prefix: /<bits>
  - Size of network address, counting from the leftmost bit
  - Example: 1.2.3.0/23



1.2.3.0/23
0000 0001
0000 0010
0000 0011
0000 0000
first 23 bits for network address
last 9 bits for host address

Network prefixes (netmasks)

- Prefix dedicated to network address
- How can we tell if a host is in a network?
  - Check if the prefix matches!
  - (bitwise AND: `addr & mask == mask`)

Mask: 123.96.0.0/12

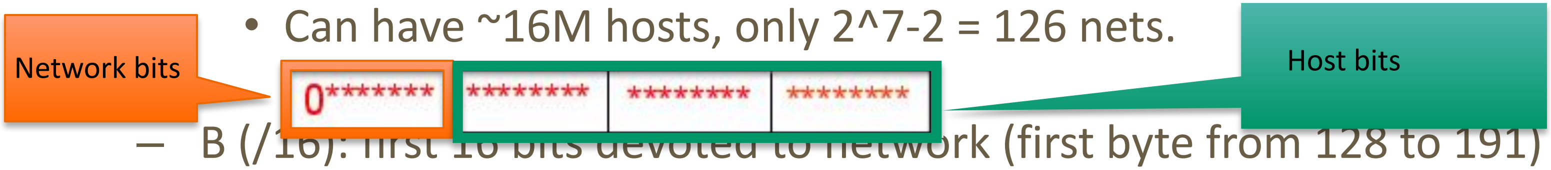
```
01111011 . 01100000 . 00000000 . 00000000
```

Addr: 123.100.42.6

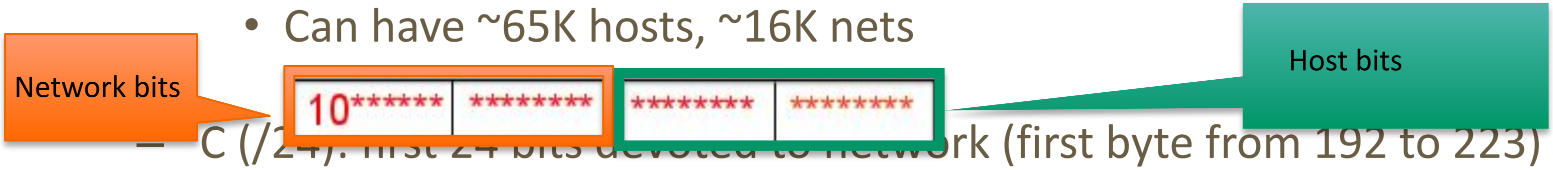
```
01111011 . 01100100 . 00101010 . 00000110
```

Classful Addressing

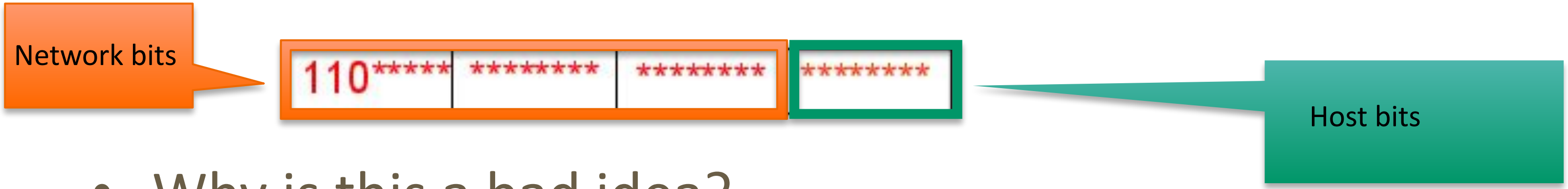
- Network classes:
  - A (/8): first 8 bits devoted to network
    - First bit is fixed to **0**.
    - first byte from 1 to 126 (0 and 127 are reserved)
    - Can have ~16M hosts, only 2^7-2 = 126 nets.



  - B (/16): first 16 bits devoted to network (first byte from 128 to 191)
    - First two bits are fixed to **10**
    - Can have ~65K hosts, ~16K nets



  - C (/24): first 24 bits devoted to network (first byte from 192 to 223)
    - First three bits are fixed to **110**
    - Can have only 254 hosts (255 is reserved for last byte) ~2M nets

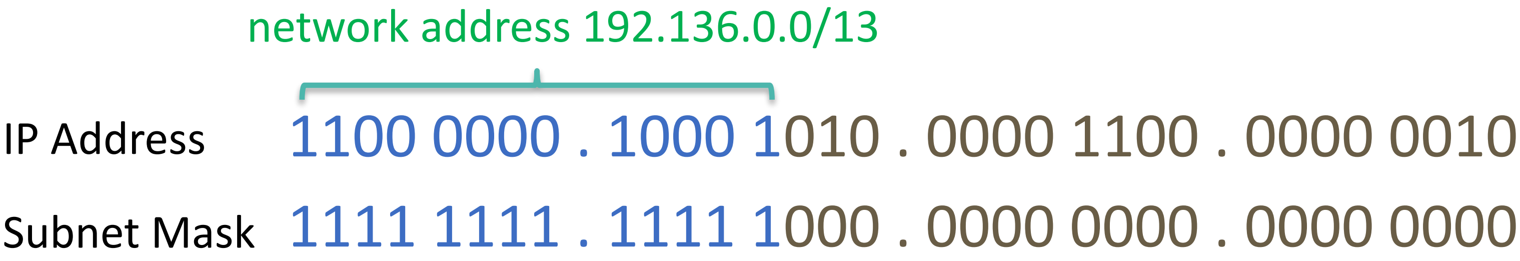


- Why is this a bad idea?

  Very limited choices lead to waste of addresses

Classless Inter-Domain Routing (CIDR)

- Use two 32-bit numbers to represent a network
  - Network address = IP Address **bitwise AND** Subnet Mask
    - IP Address is 192.138.12.2
    - Subnet Mask is 255.248.0.0



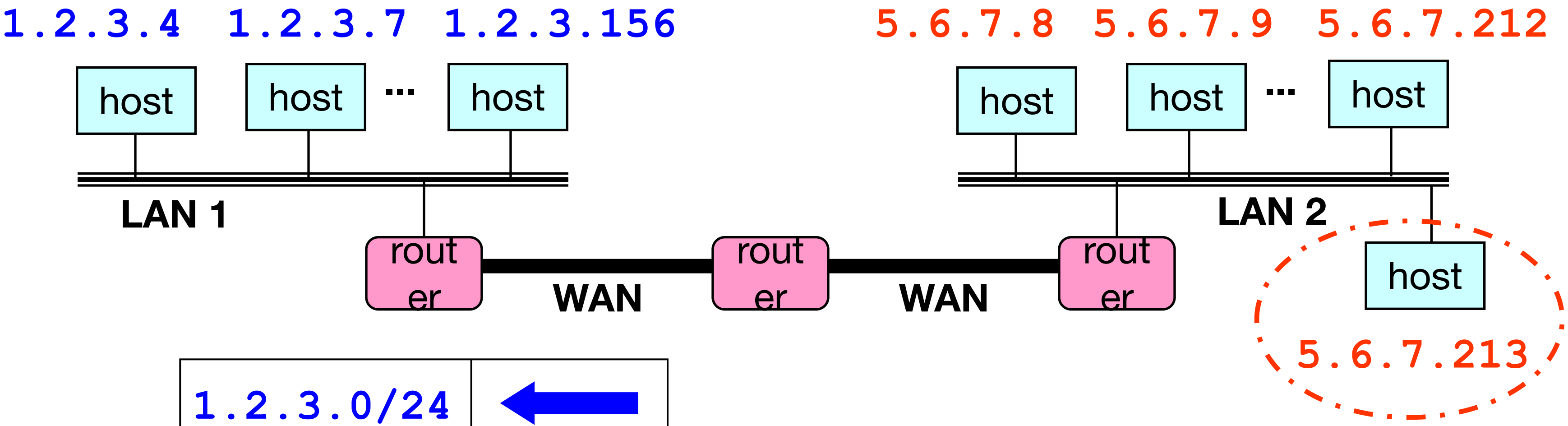
network address 192.136.0.0/13

IP Address	1100 0000 . 1000 1010 . 0000 1100 . 0000 0010
Subnet Mask	1111 1111 . 1111 1000 . 0000 0000 . 0000 0000

- Flexible division of bits:
  - More choices for the size of the network and hosts
- Offers better size routing table and efficient IP address space

Prefixes

- Easy to Add New Hosts
  - New host (5.6.7.213)
  - Forwarding table doesn't need to be updated!



1.2.3.0/24	←
5.6.7.0/24	→

forwarding table

Worksheet: Q3

Interdomain Routing

- **Inter**domain routing is between autonomous systems (AS)
  - Similar goals as intradomain routing with scalability + policy compliance
  - Autonomous systems want privacy and autonomy
- Border gateway protocol (BGP) is current design
  - Extends on top of DV (with some crucial differences)

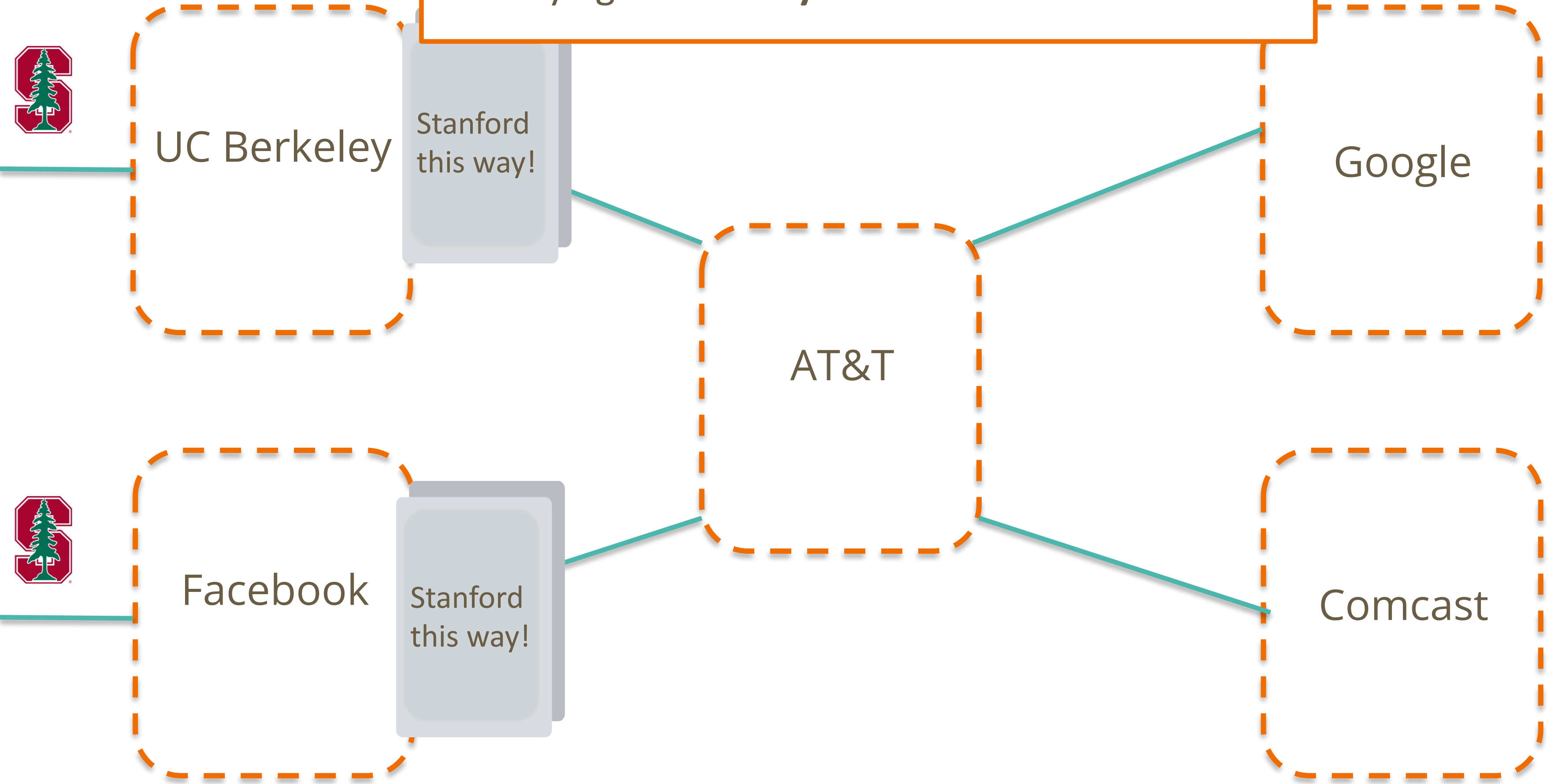
Export & Selection

- If you are an AS:
  - Route Selection
    - **Where you send your packets**
    - Determine how to choose a valid route to a given IP prefix, when multiple paths through ASes
  - Route Export
    - **Which ASes will receive your route**
    - Other ASes will *select* your route and **send traffic to you**

Export & Selection

UCB and Facebook **export** their route to Stanford

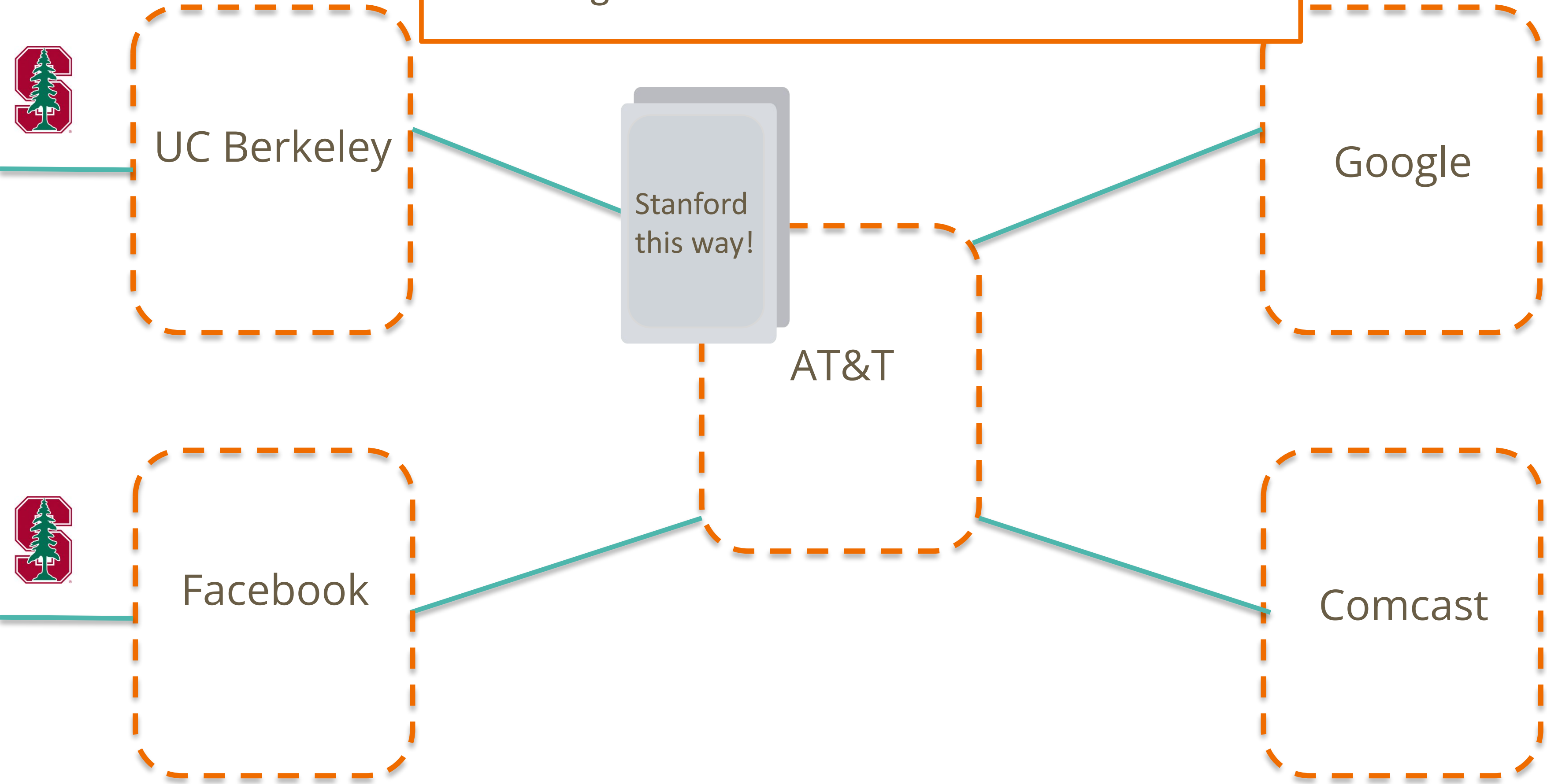
They agree to **carry** AT&T's traffic to Stanford



Export & Selection

AT&T **selects** route through UC Berkeley

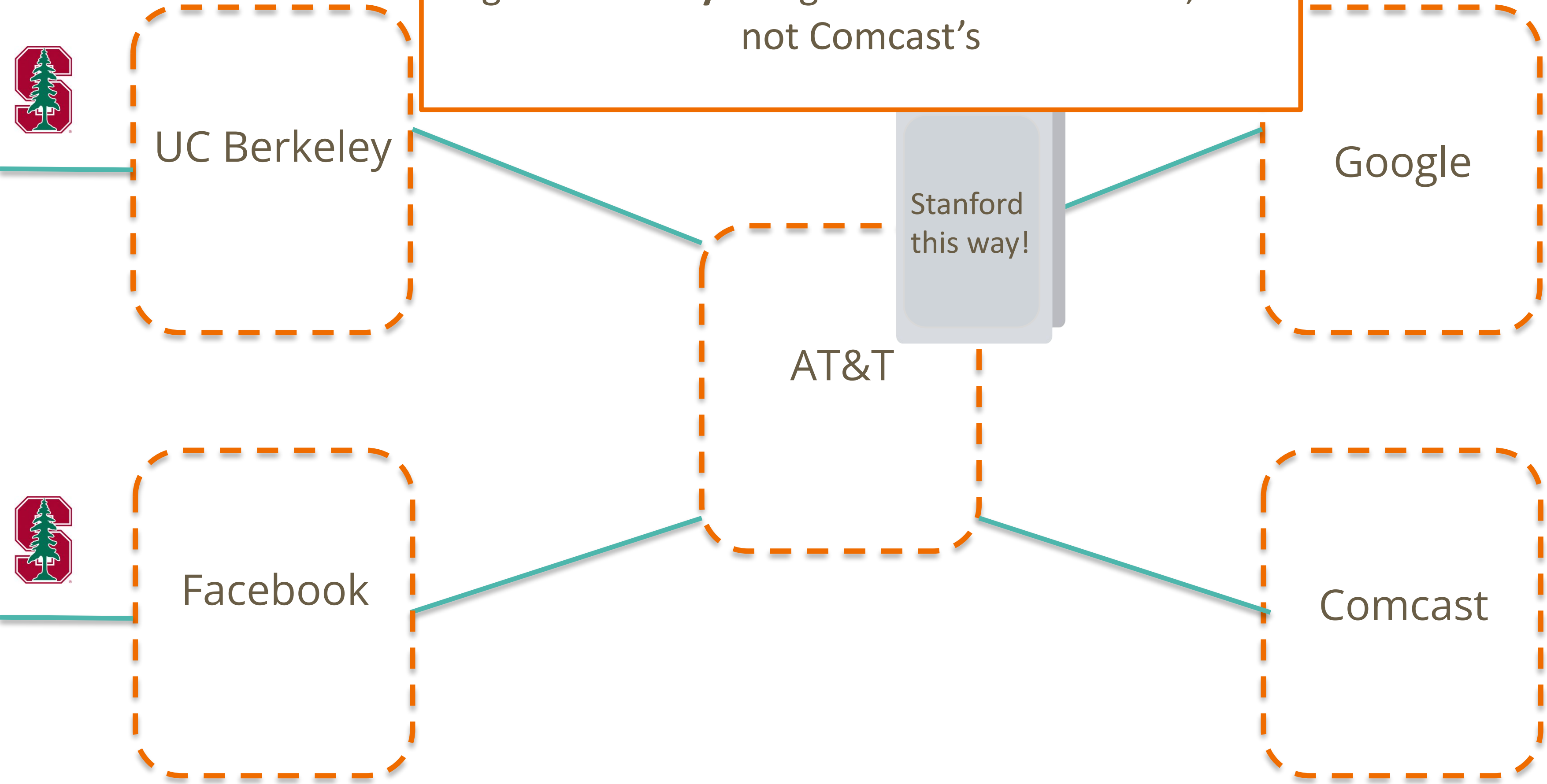
AT&T might now **send** traffic to Stanford via UCB



Export & Selection

AT&T **exports** route to Google only

Agrees to **carry** Google's traffic to Stanford, but not Comcast's



Types of ASes (domains)

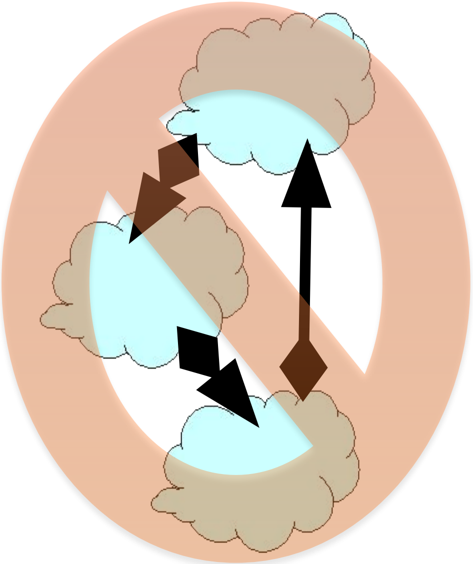
- **Stub**: only sends/receives traffic for its users
  - companies, universities, etc.
- **Transit**: carries traffic for other ASes
  - Global ISPs (Tier 1): fully connected mesh
  - Regional ISPs (Tier 2)
  - Local ISPs (Tier 3)
- Lower tiers buy service from higher tiers
- What’s the relationship between AS and ISP?
  - All ISPs are ASes, but not all ASes are ISPs
  - E.g. UC Berkeley is not an ISP but it is an AS

Business Relationship among ASes

- Two ASes will **connect** only if they have business relationship:
  - **Customer-Provider**
    - *Provider* B carries *customer* A's traffic **for a fee**
  - **Peers**
    - *Peers* A, B carry each other's traffic **for free**
- What roles can a global ISP (Tier 1) have?
  - Provider to Tier 2 or Tier 3
  - Peer to other global ISP (tier 1)
  - Not a customer!

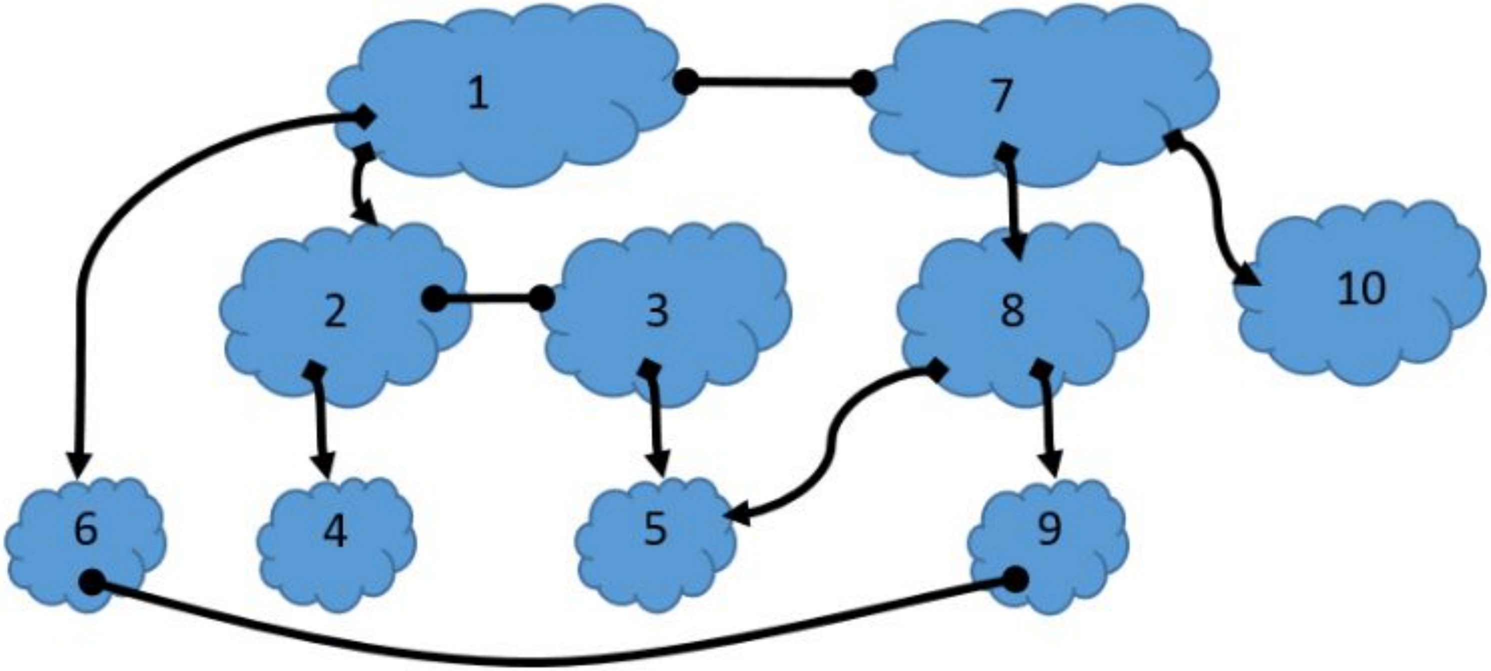
Business Relationship Restrictions

- The graph of **peering** relations can be ***cyclic***
  - The peer of my peer can also be my peer
  - For example, global ISPs all peer with each other
- The graph of **customer-provider** relations must be ***acyclic***

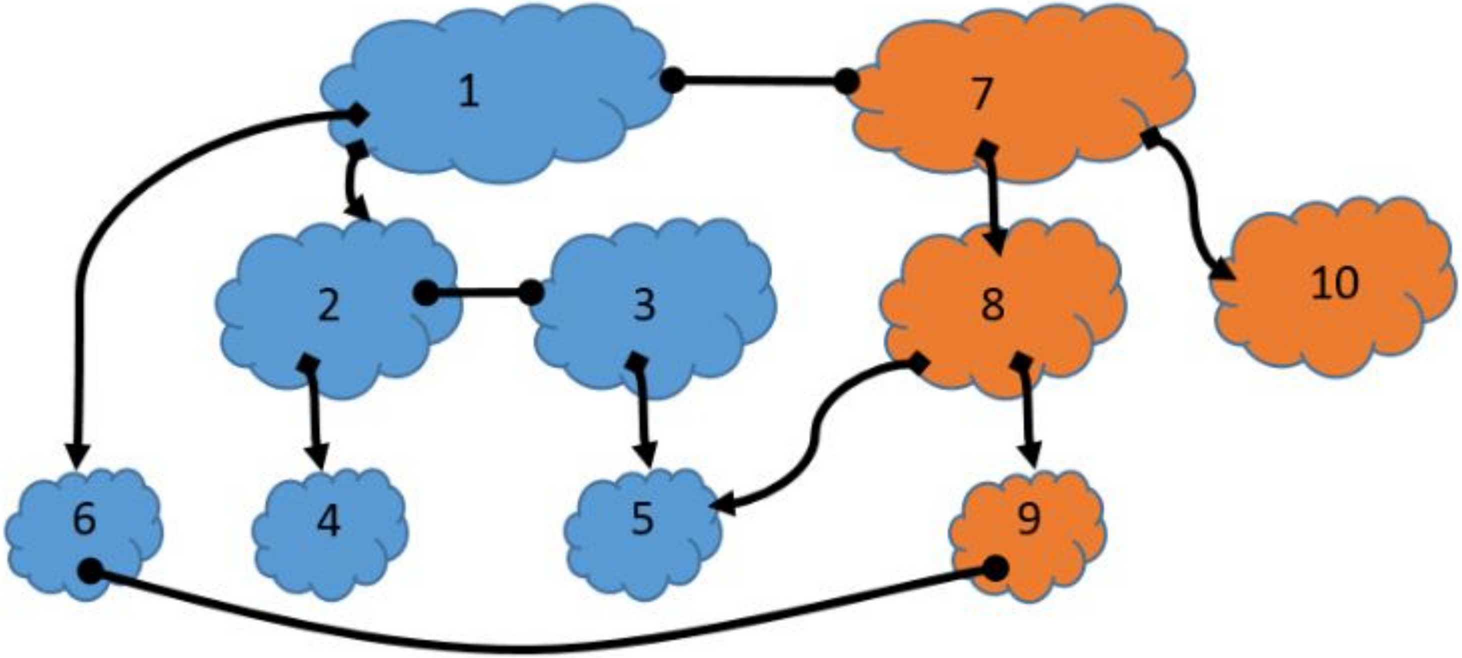


Worksheet: Q4

Question 3



Question 3


1
7
2
3
8
10
6
4
5
9