

Advanced Computer Networks

Addressing and Routing in Data Center Networks (I)

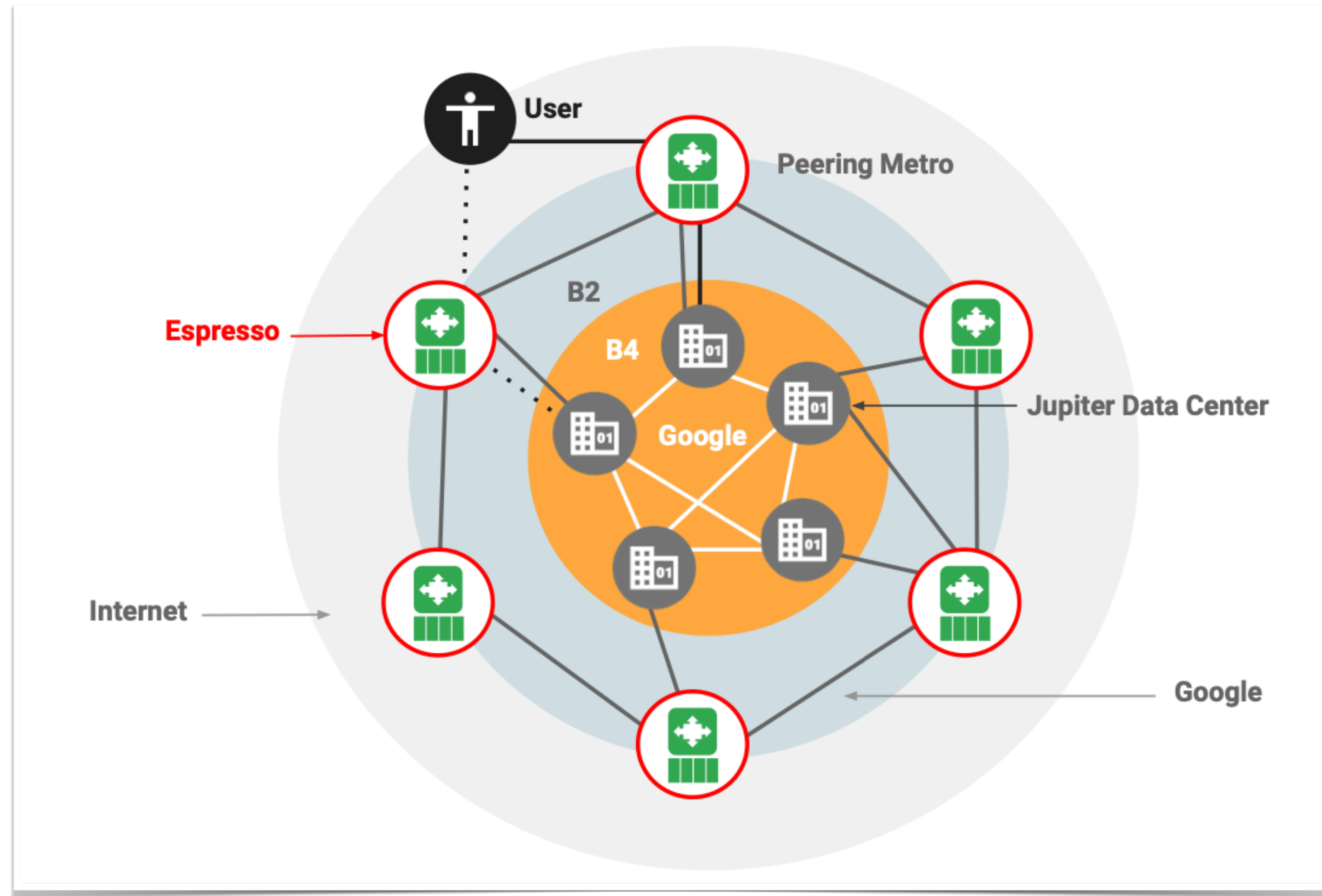
<https://pages.cs.wisc.edu/~mgliu/CS740/F26/index.html>

Ming Liu
mgliu@cs.wisc.edu

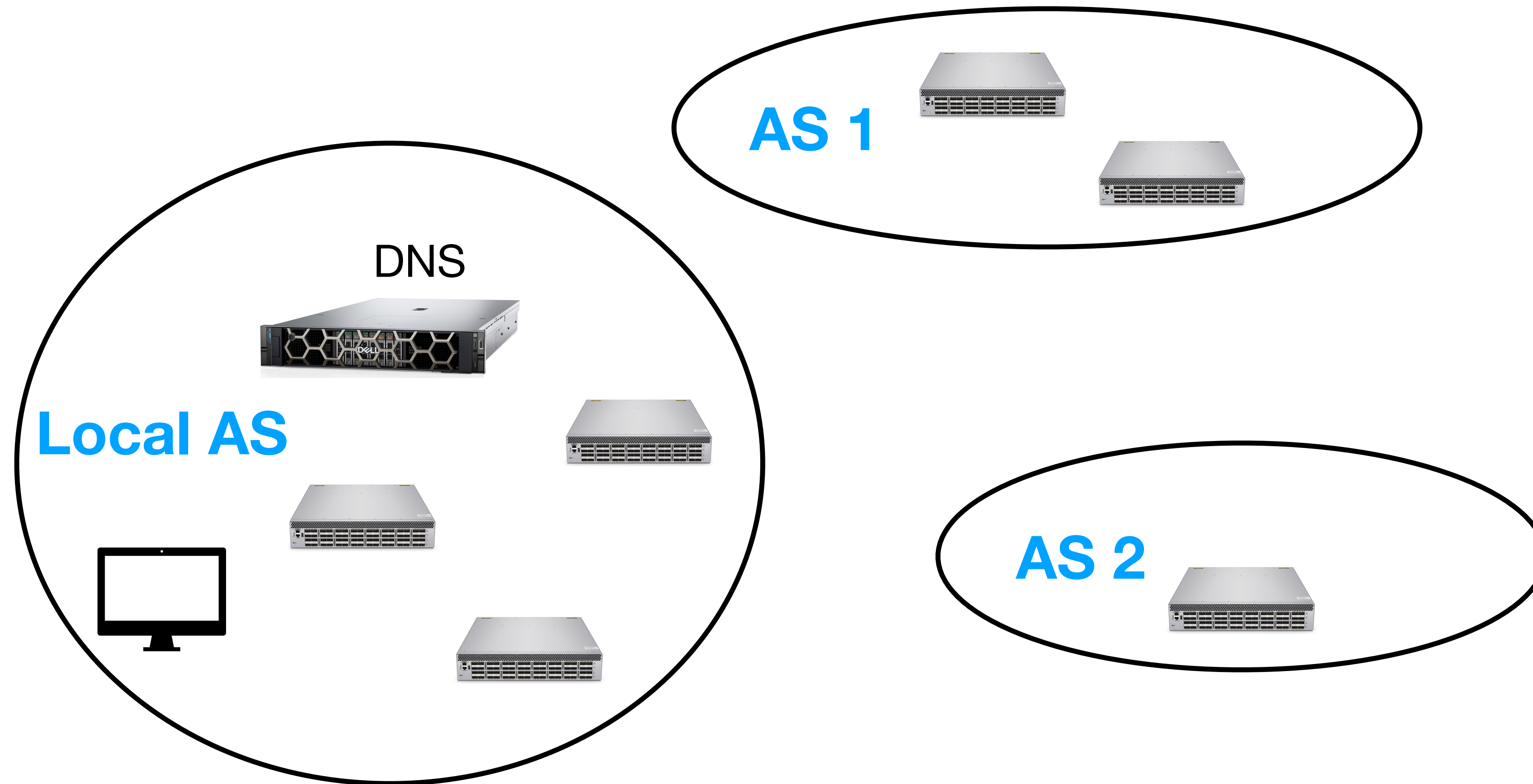
Outline

- Last lecture
 - Physical connectivity beyond the data center
- Today
 - Addressing and routing in data center networks (I)
- Announcements
 - Lab1 due 10/09/2026 11:59 PM
 - No class this Thursday

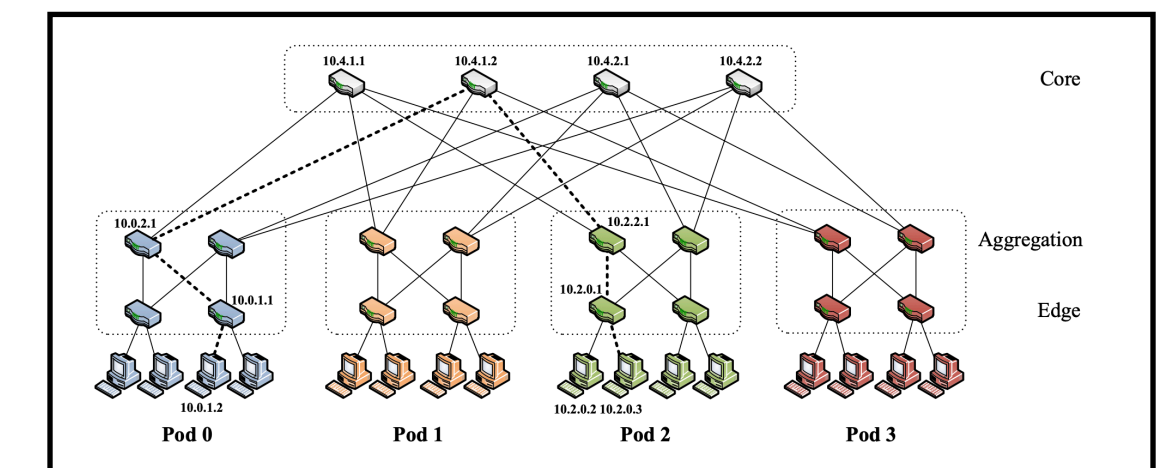
Data Center Communications



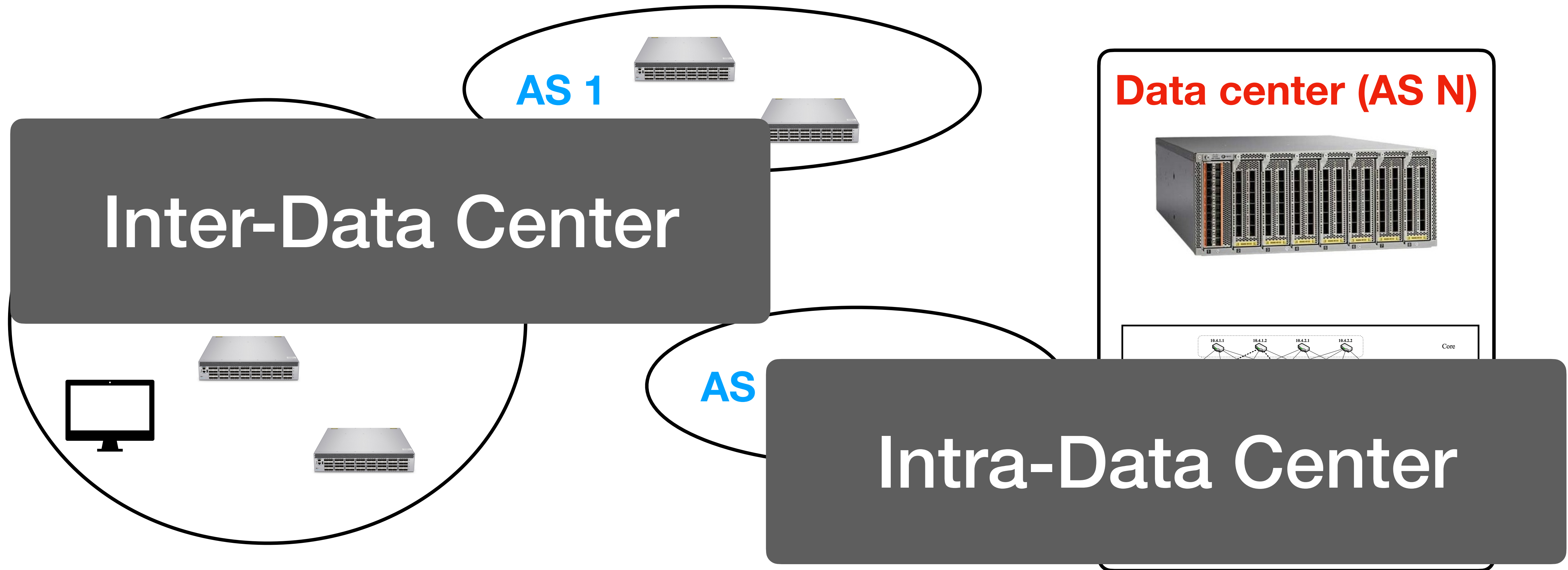
Dissect the Communication Path



Data center (AS N)



Dissect the Communication Path



Data Movement

- Three types of data movements
 - #1: User $\longleftrightarrow$ Data center gateway (Inter-DC)
 - #2: Data center gateway $\longleftrightarrow$ Data center server (Intra-DC)
 - #3: Data center server $\longleftrightarrow$ Data center server (Intra-DC)

Data Movement

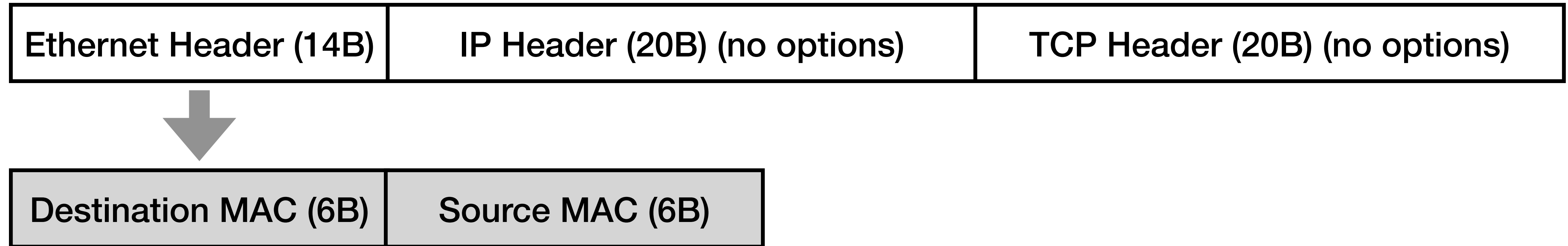
- Three types of data movements
 - #1: User $\longleftrightarrow$ Data center gateway (Inter-DC)
 - #2: Data center gateway $\longleftrightarrow$ Data center server (Intra-DC)
 - #3: Data center server $\longleftrightarrow$ Data center server (Intra-DC)

- Data = Packet
- Three aspects of data movement:
 - Source: What is the source address?
 - Destination: What is the destination address?
 - Path: Which routing path is taken?

Recap: Packet Format

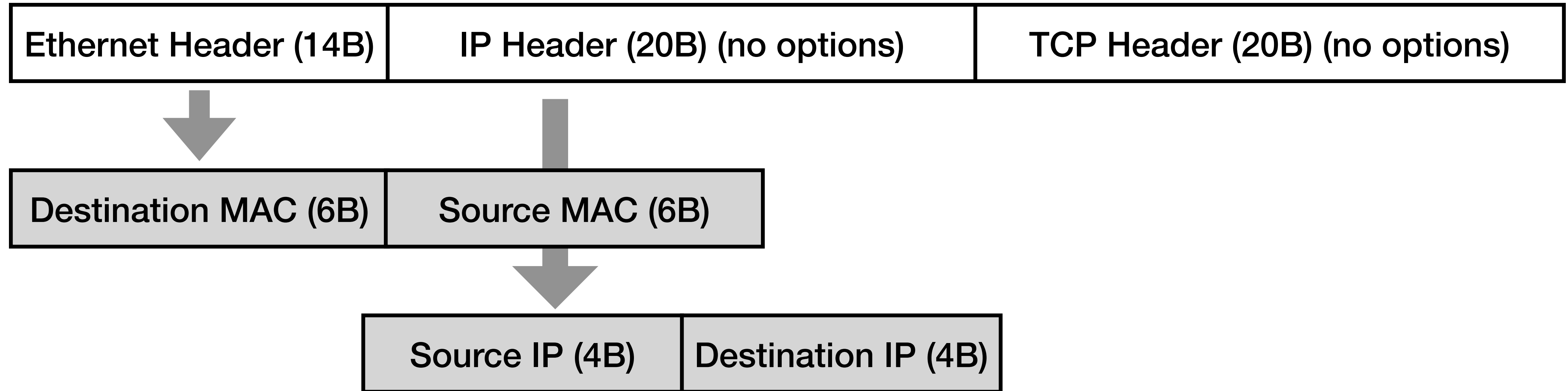
Ethernet Header (14B)	IP Header (20B) (no options)	TCP Header (20B) (no options)
-----------------------	------------------------------	-------------------------------

Recap: Packet Format



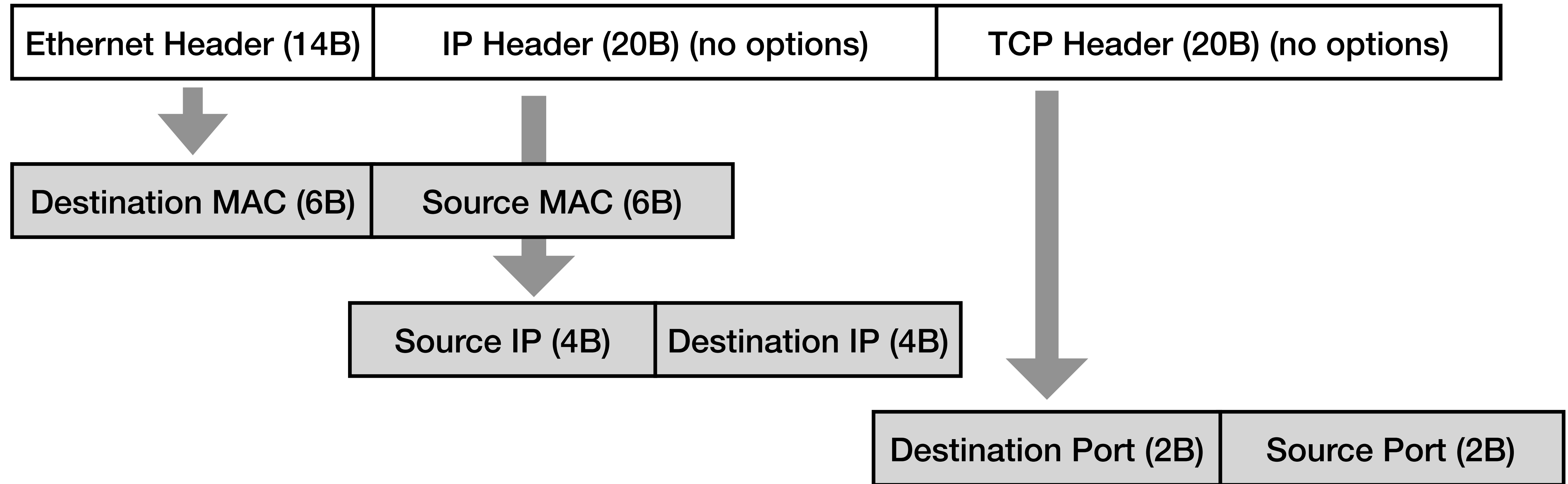
- MAC address: physical host (NIC) identification

Recap: Packet Format



- MAC address: physical host (NIC) identification
- IP address: virtual host identification

Recap: Packet Format

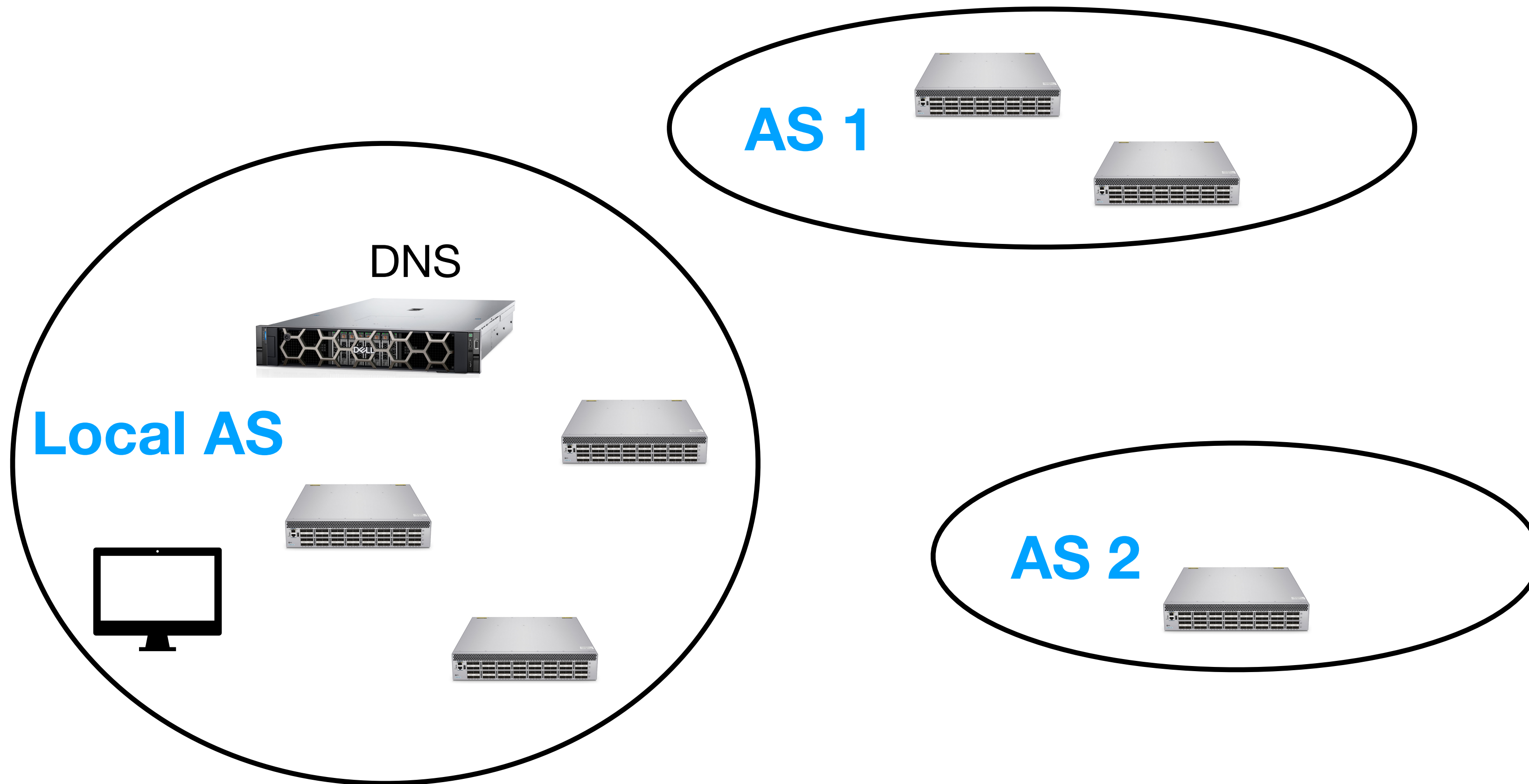


- MAC address: physical host (NIC) identification
- IP address: virtual host identification
- Port number: application process identification

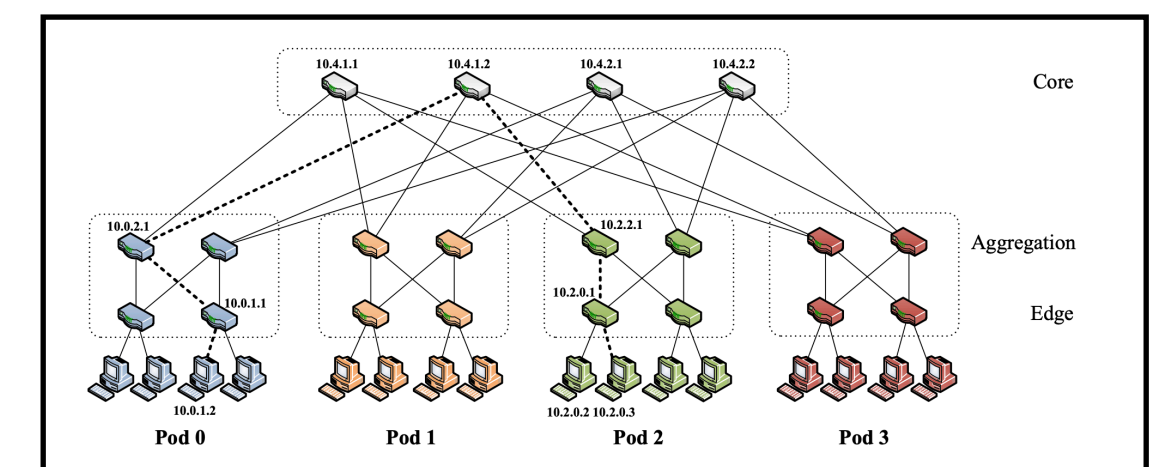
Which address should we use for routing?

Type #1: User $\leftrightarrow$ Data Center Gateway

- Similar to what we have learned in CS640



Data center (AS N)



Suppose a **process A** running on a **PC X** communicating with a **process B** running on a **server Y** (inside the data center),

Suppose a **process A** running on a **PC X** communicating with a **process B** running on a **server Y** (inside the data center),

- Addressing (User —> Data center gateway)
 - Source MAC
 - Destination MAC
 - Source IP
 - Destination IP
 - Source Port
 - Destination Port

Suppose a **process A** running on a **PC X** communicating with a **process B** running on a **server Y** (inside the data center),

- Addressing (User —> Data center gateway)
 - Source MAC = X's NIC MAC
 - Destination MAC = X's local gateway's NIC MAC
 - Source IP
 - Destination IP
 - Source Port
 - Destination Port

Suppose a **process A** running on a **PC X** communicating with a **process B** running on a **server Y (inside the data center)**,

- Addressing (User —> Data center gateway)
 - Source MAC = X's NIC MAC
 - Destination MAC = X's local gateway's NIC MAC
 - Source IP = X's IP address
 - Destination IP = The IP provided by DNS = Data center gateway's IP
 - Source Port
 - Destination Port

Suppose a **process A** running on a **PC X** communicating with a **process B** running on a **server Y** (inside the data center),

- Addressing (User —> Data center gateway)
 - Source MAC = X's NIC MAC
 - Destination MAC = X's local gateway's NIC MAC
 - Source IP = X's IP address
 - Destination IP = The IP provided by DNS = Data center gateway's IP
 - Source Port = A's port assigned by its TCP/IP stack
 - Destination Port = The service port of B

Suppose a **process A** running on a **PC X** communicating with a **process B** running on a **server Y** (inside the data center),

- Routing (User —> Data center gateway)
 - X —> Local Gateway
 - Local Gateway —> X's AS's Border Router
 - X's AS —> Data Center Gateway

Suppose a **process A** running on a **PC X** communicating with a **process B** running on a **server Y (inside the data center)**,

- Routing (User —> Data center gateway)
 - X —> Local Gateway: Destination MAC
 - Local Gateway —> X's AS's Border Router: Destination IP
 - X's AS —> Data Center Gateway: Destination IP

Suppose a **process A** running on a **PC X** communicating with **process B** running on

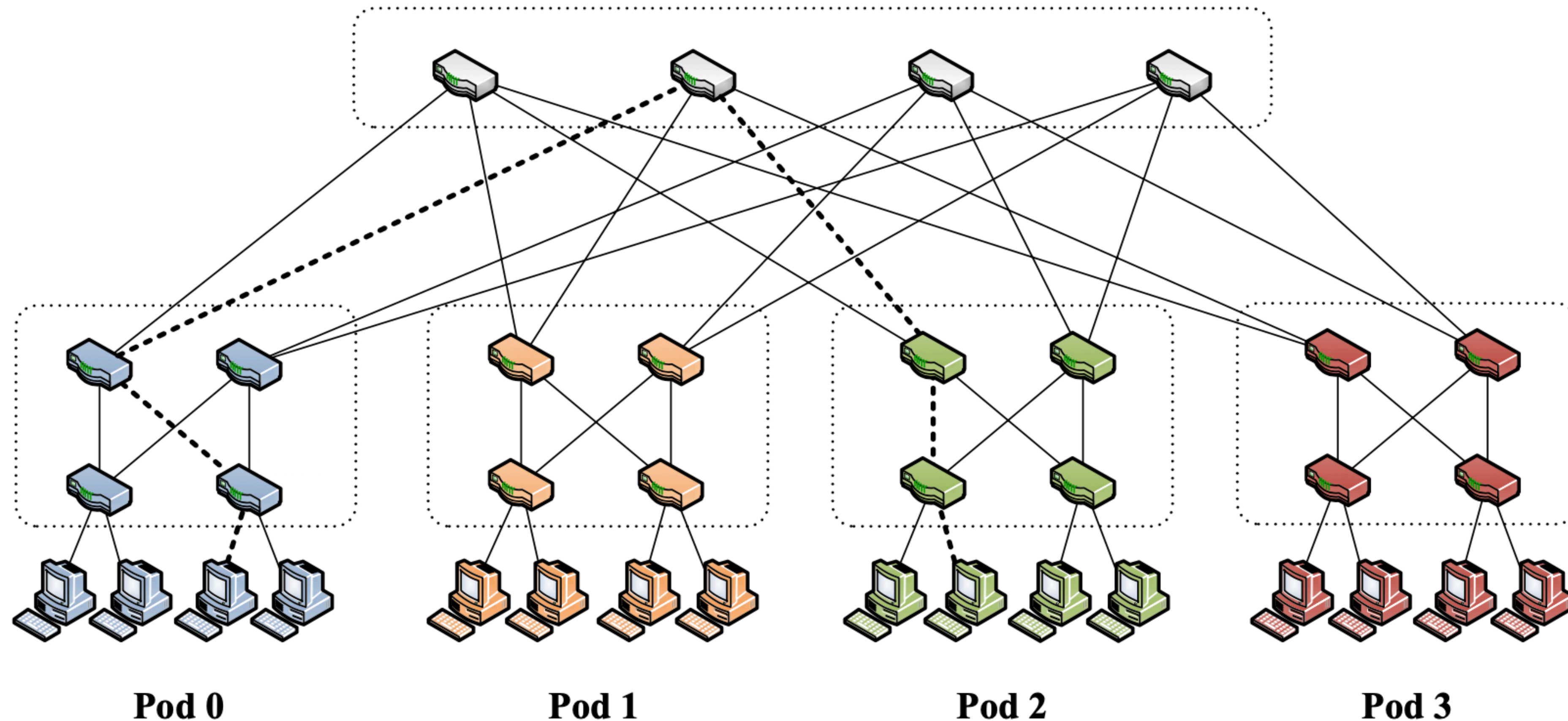
Table lookup

MAC	Port
01:02:03:04:05:06	4
07:08:09:0A:0F:0E	5

IP	Port
128.105.146.64	6
128.105.146.65	7

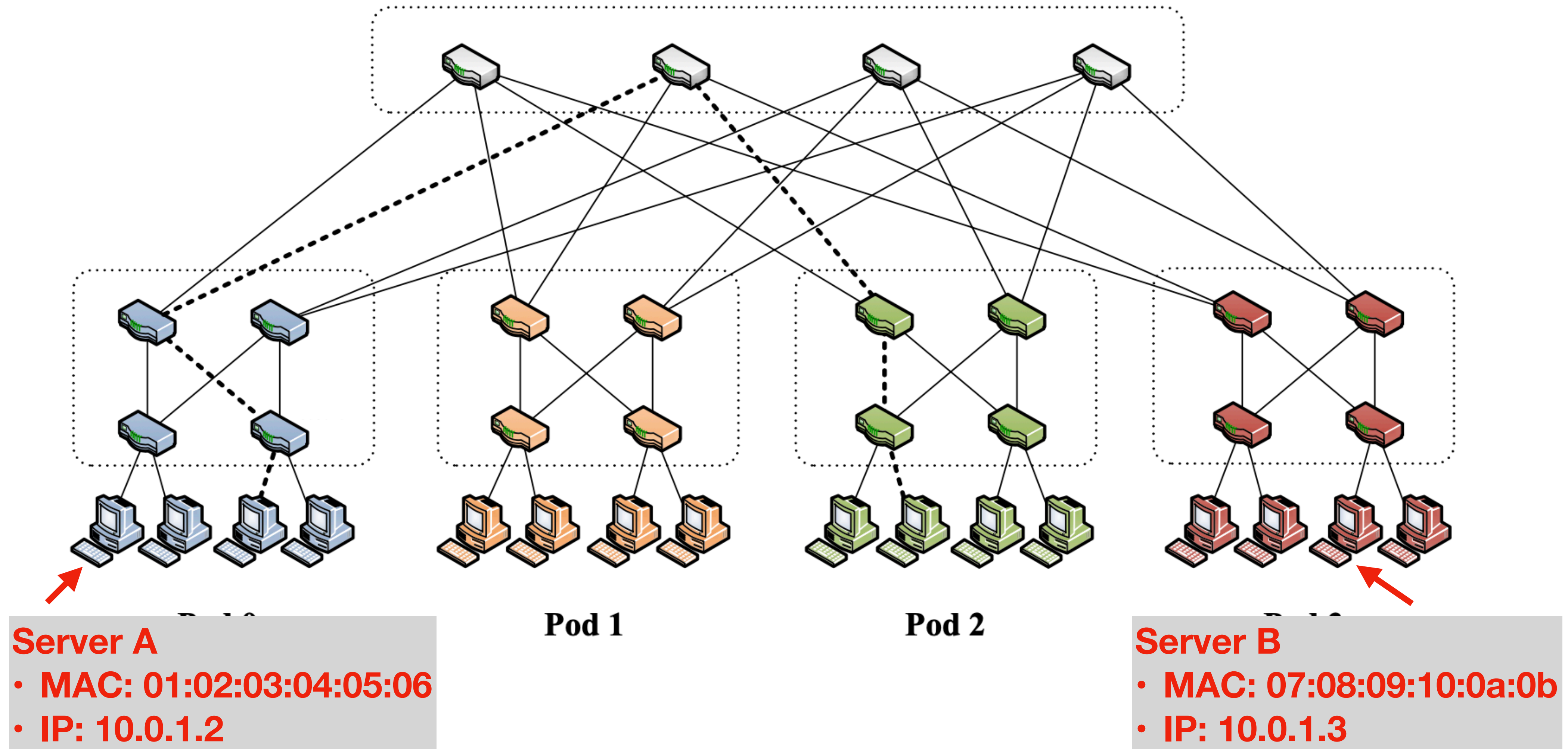
Type #3: Data Center Server $\longleftrightarrow$ Data Center Server

- Within an AS, but with a great number of servers



Type #3: Data Center Server $\longleftrightarrow$ Data Center Server

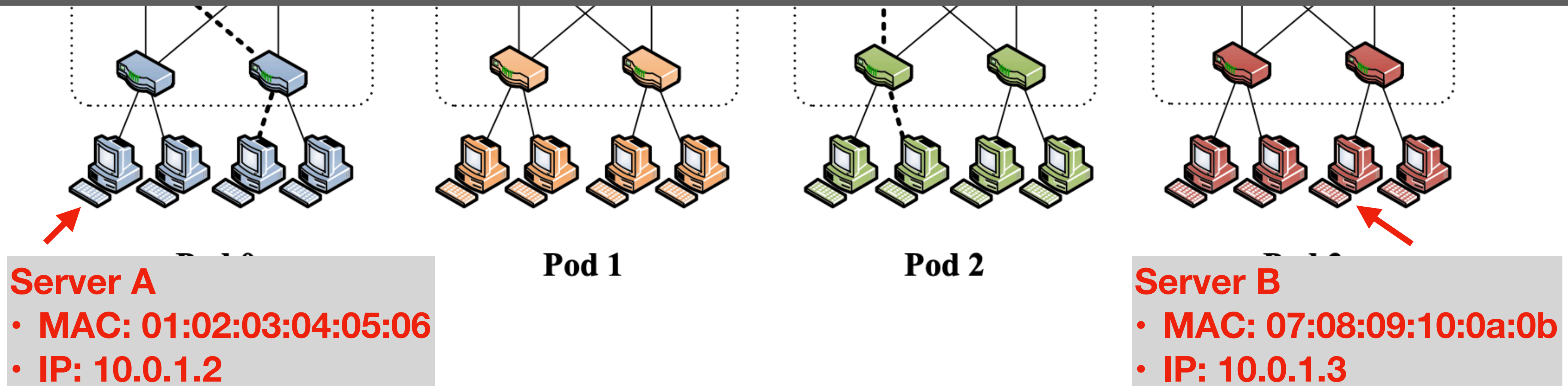
- Within an AS, but with a great number of servers



Type #3: Data Center Server $\longleftrightarrow$ Data Center Server

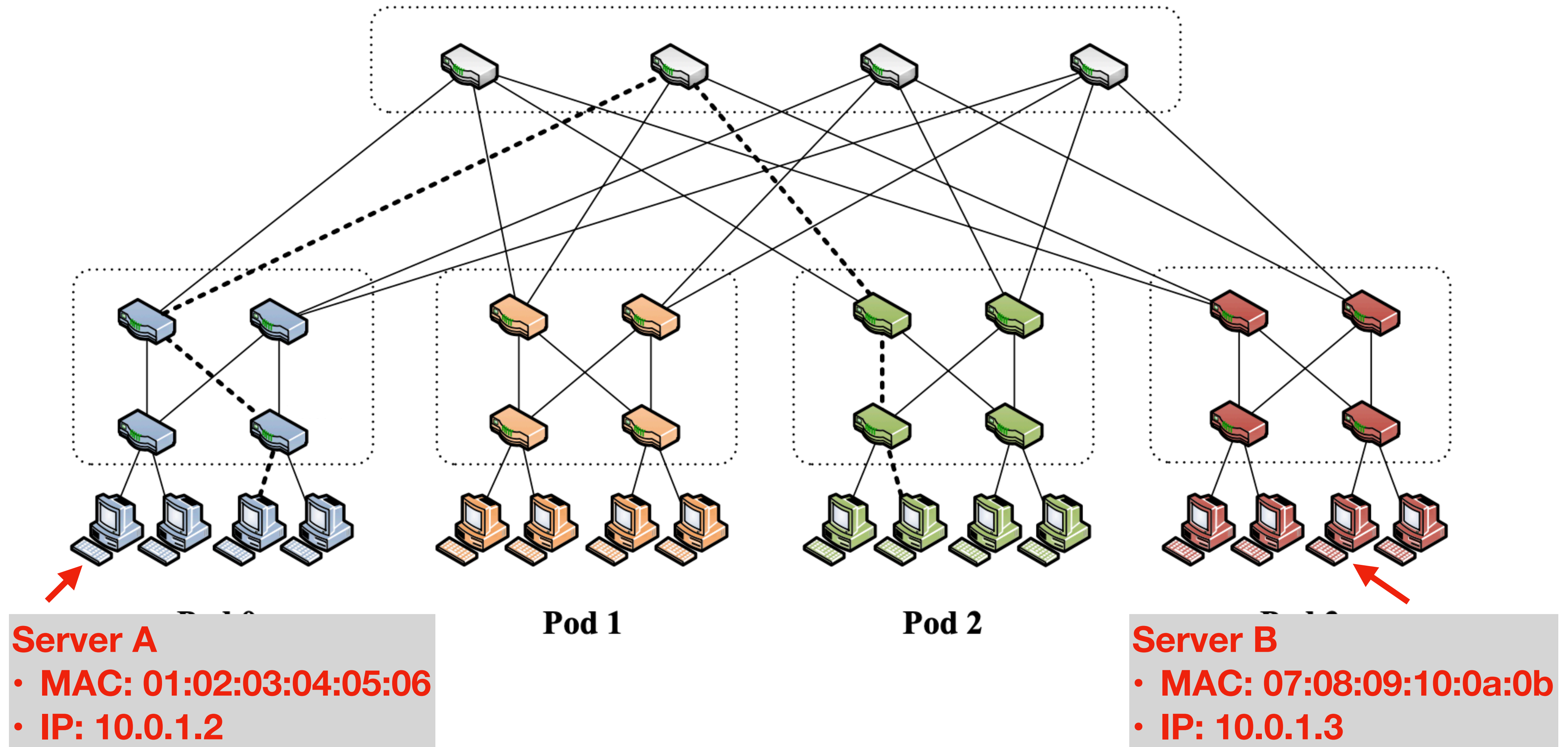
- Within an AS, but with a great number of servers

Which address should we use for routing?



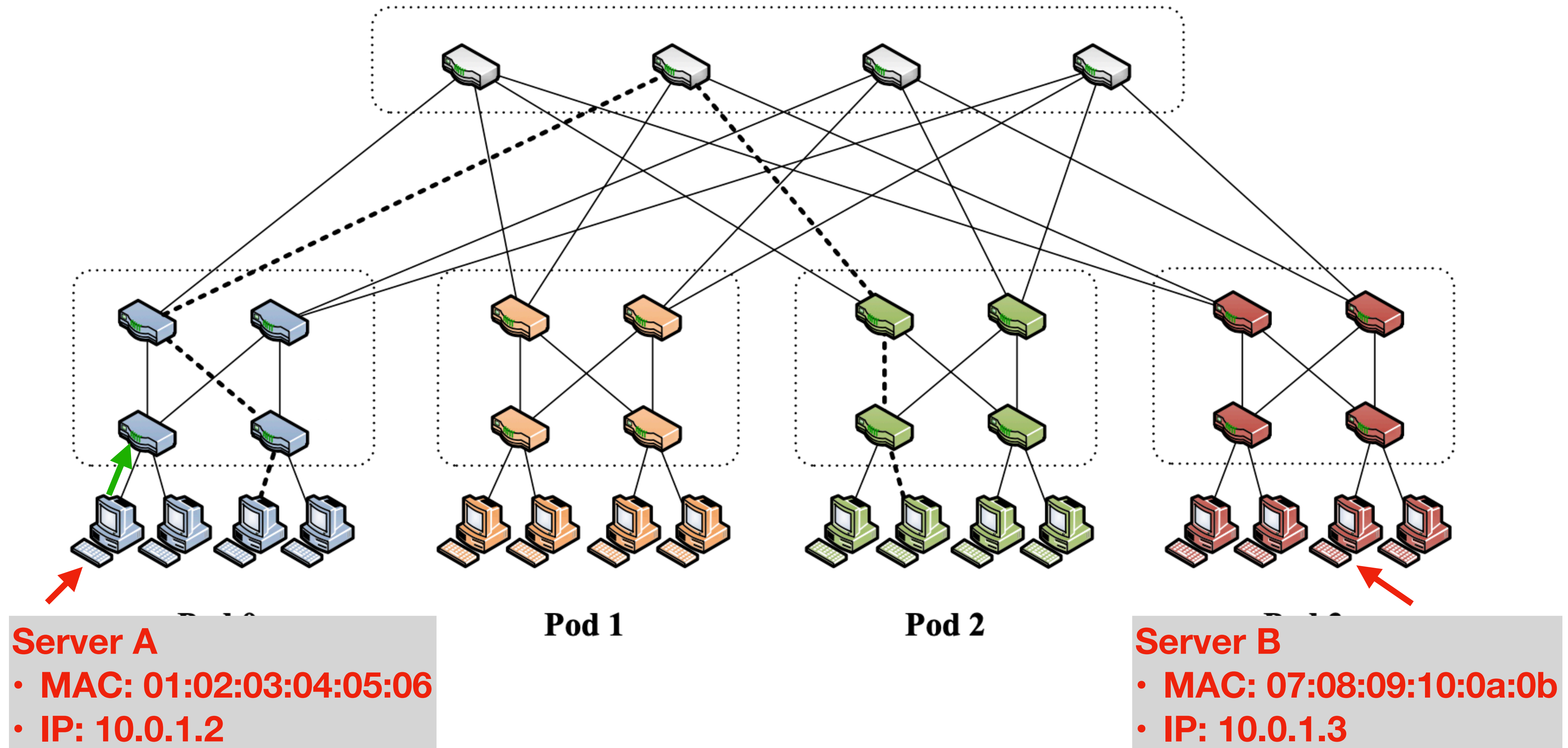
Proposal #1: Routing based on the MAC address

- Address Resolution Protocol (ARP) + MAC learning



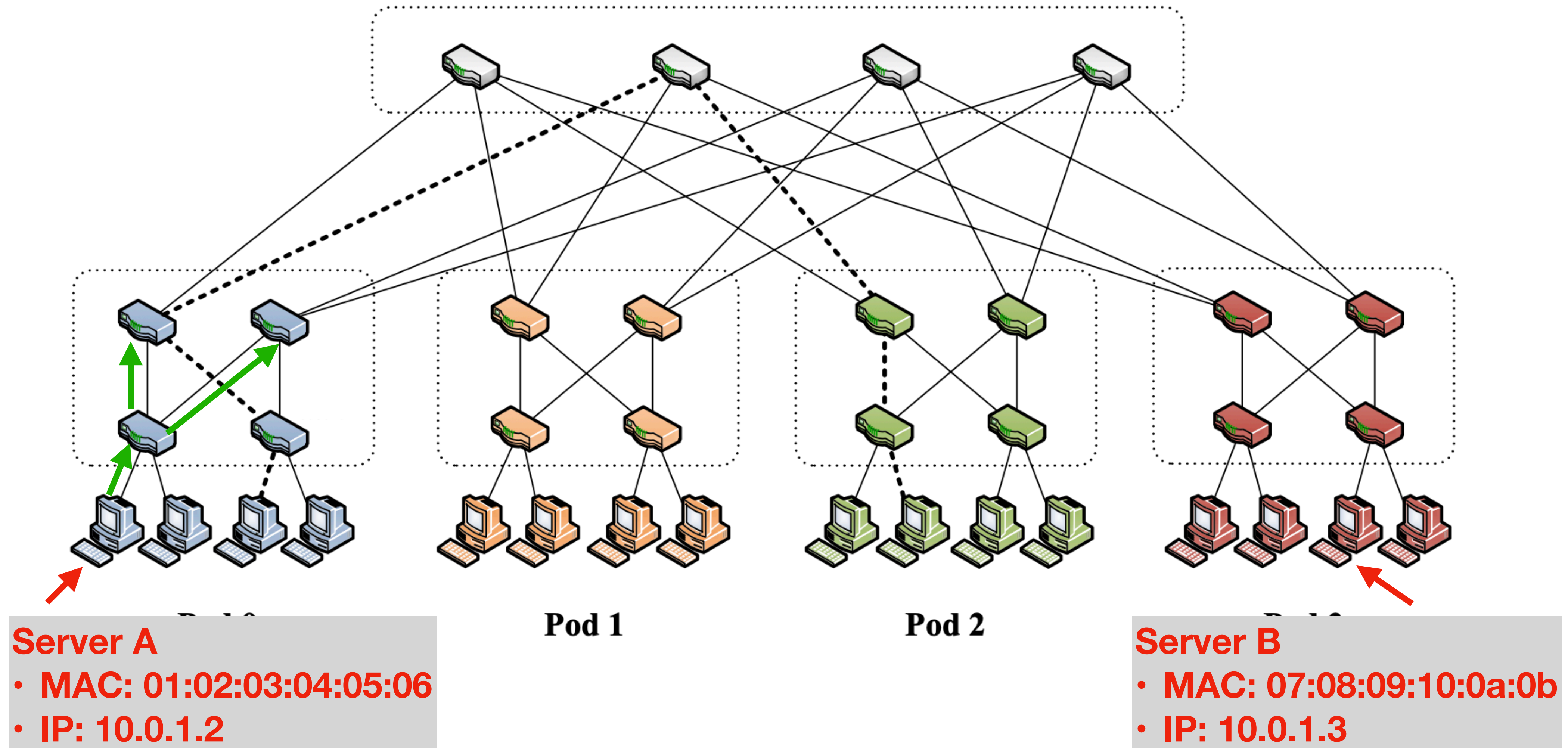
Proposal #1: Routing based on the MAC address

- Address Resolution Protocol (ARP) + MAC learning



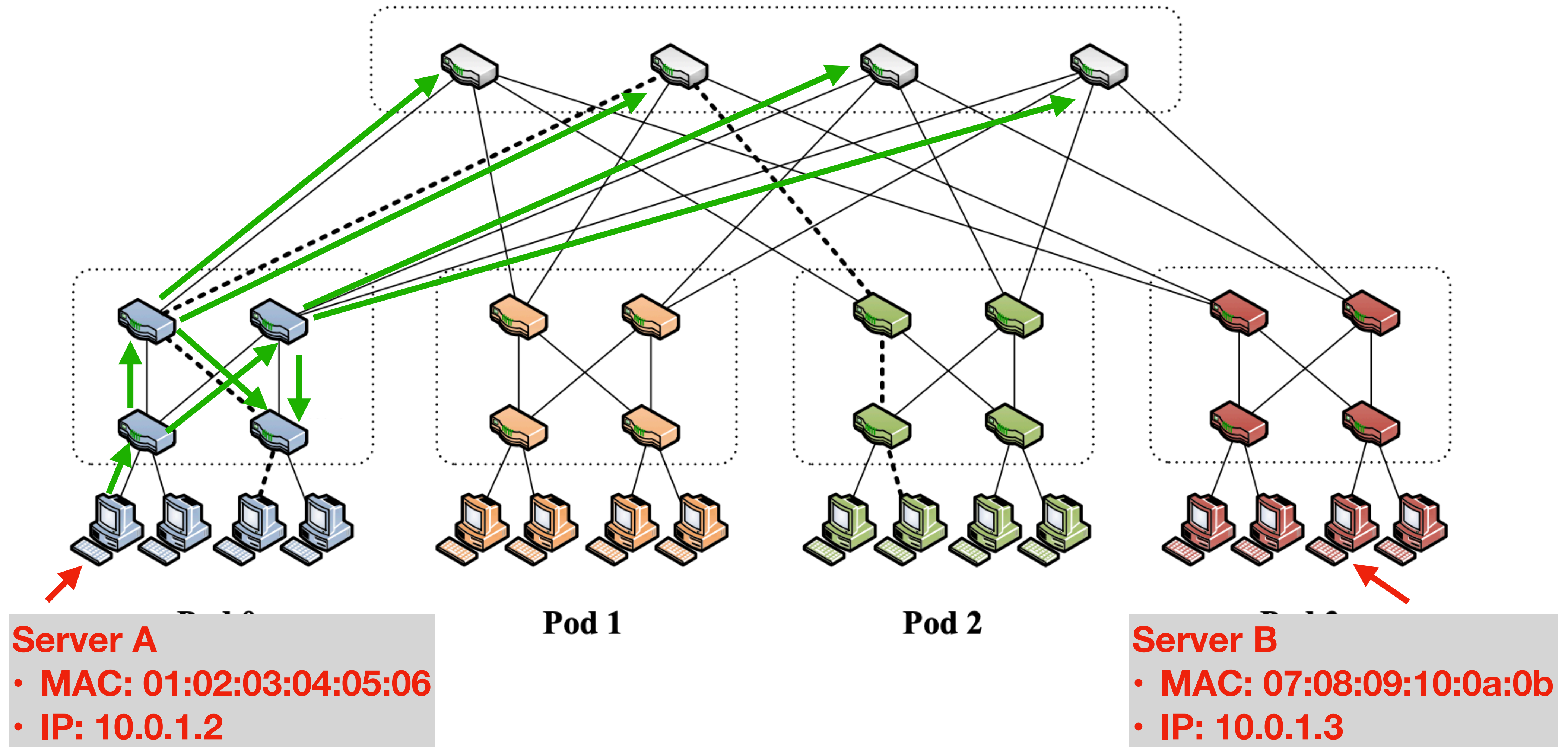
Proposal #1: Routing based on the MAC address

- Address Resolution Protocol (ARP) + MAC learning



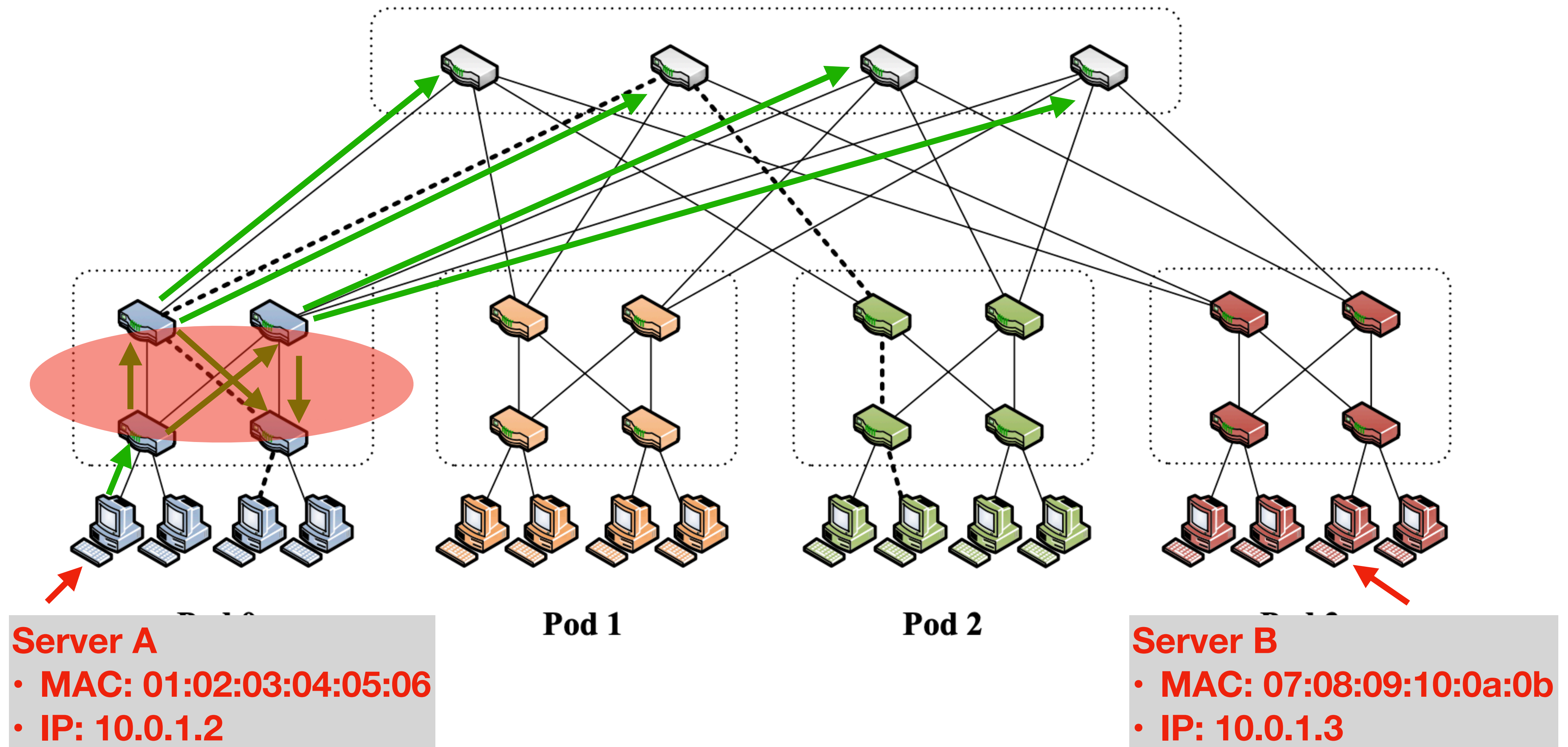
Proposal #1: Routing based on the MAC address

- Address Resolution Protocol (ARP) + MAC learning



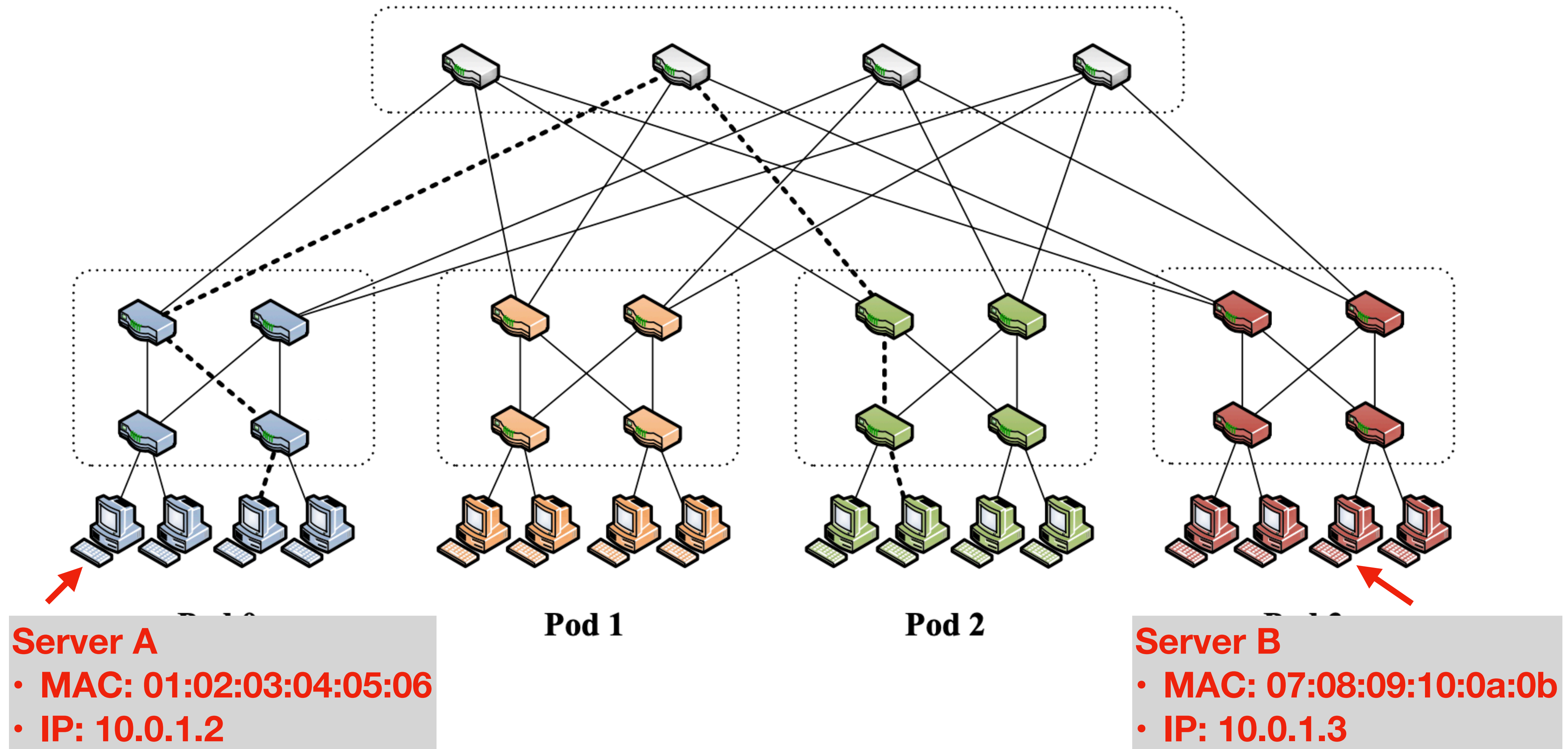
Proposal #1: Routing based on the MAC address

- Address Resolution Protocol (ARP) + MAC learning



Proposal #1: MAC + Spanning Tree

- Reduce a topological graph to a tree



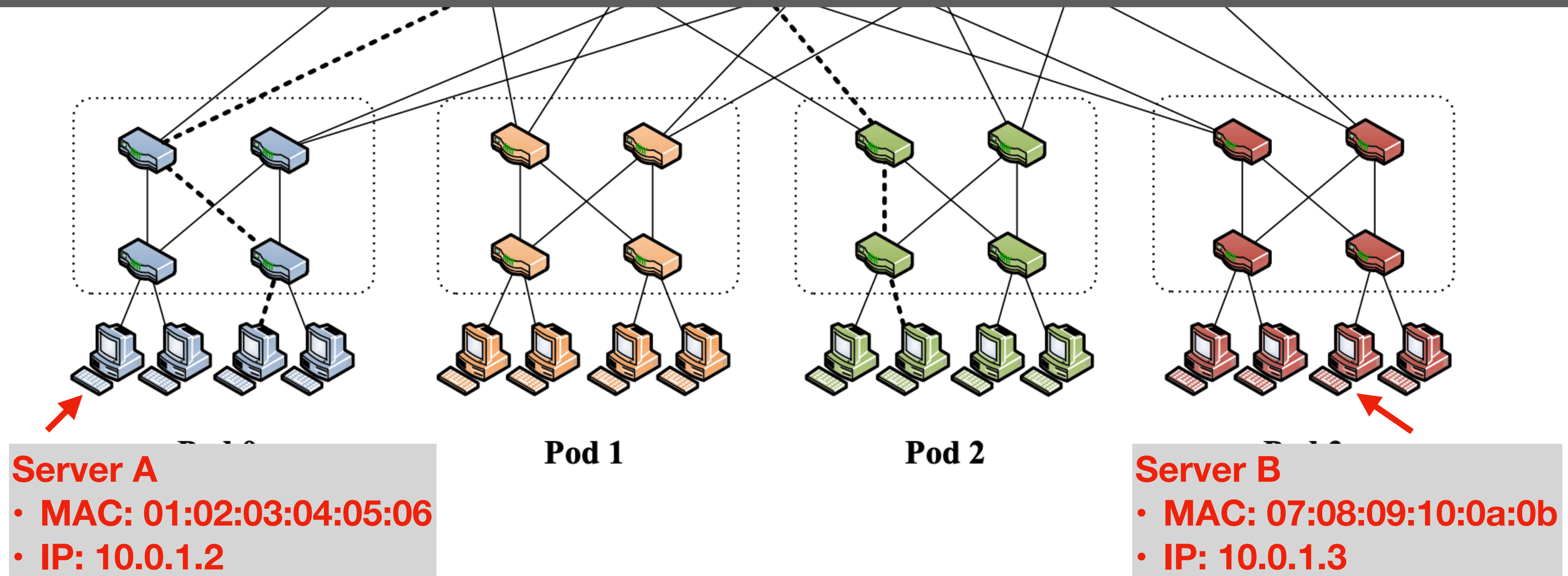
Recap: Spanning Tree

- Three basic steps:
 - #1: Select a root —> Use the switch ID
 - #2: Decide the shortest path to the root —> Use switch ID to break a tie
 - #3: Configure a designated port for forwarding —> No blindly broadcast
- Each switch maintains the following info:
 - The ID of the local and root switch
 - The distance to the root switch and per-port action table
- Configuration message: (Y, d, X)
 - Y: root switch ID in my view
 - d: the distance to the root
 - X: my local switch ID

Proposal #1: MAC + Spanning Tree

- Reduce a topological graph to a tree

Does it work?

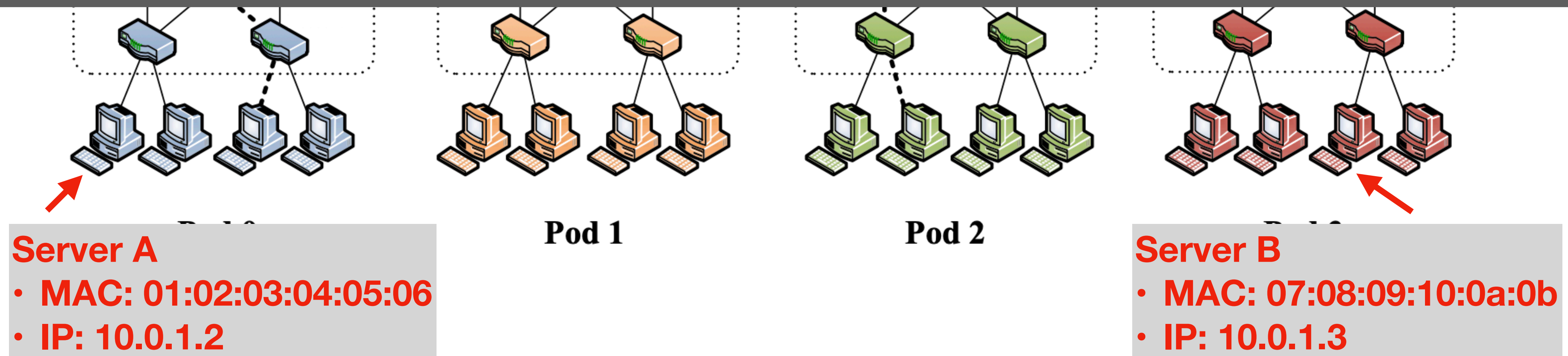


Proposal #1: MAC + Spanning Tree

- Reduce a topological graph to a tree

Does it work?

Does it work well?



Proposal #1: MAC + Spanning Tree

- Reduce a topological graph to a tree

Does it work?

Does it work well?

What are the issues?

Server A

- MAC: 01:02:03:04:05:06
- IP: 10.0.1.2

Pod 1

Pod 2

Server B

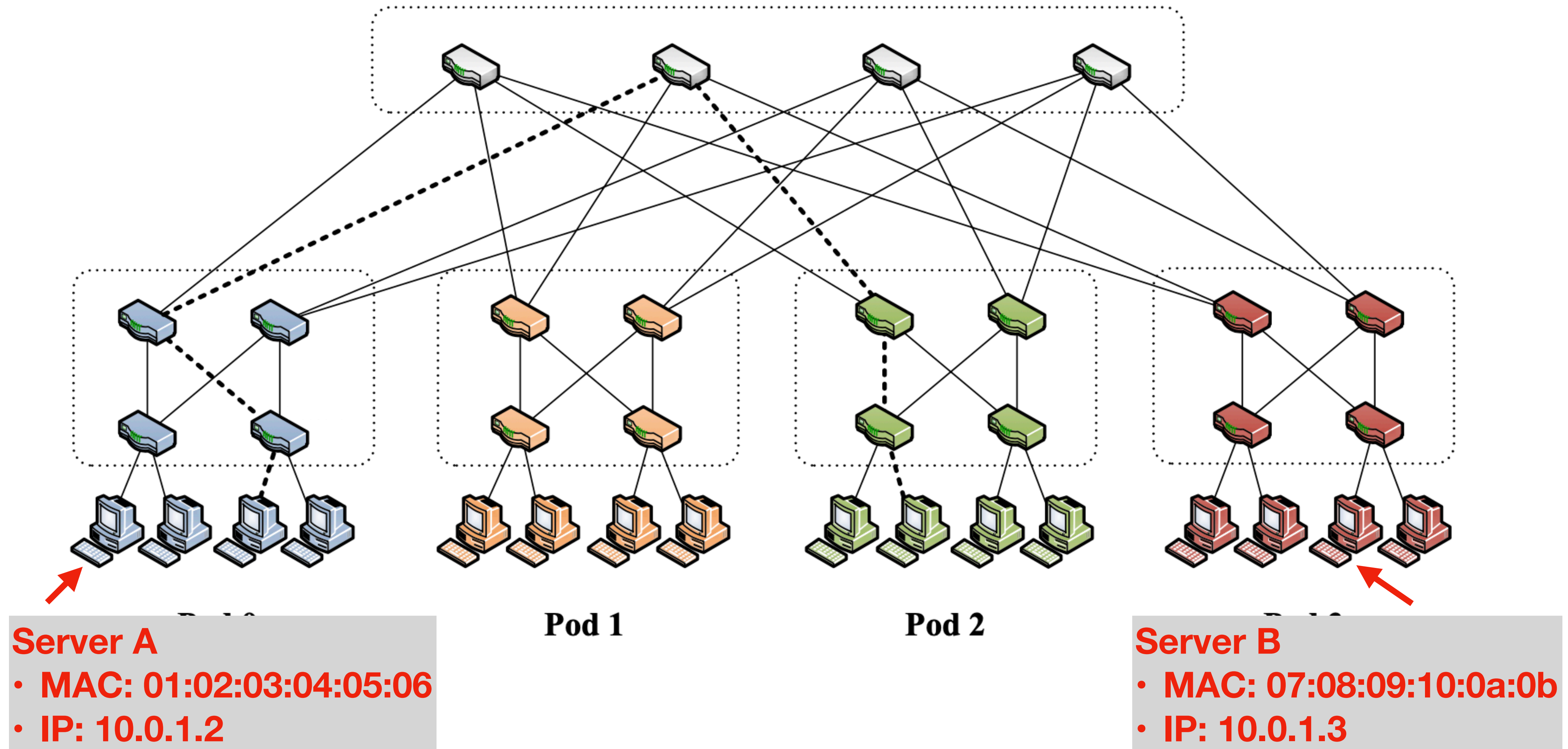
- MAC: 07:08:09:10:0a:0b
- IP: 10.0.1.3

Proposal #1: Pros and Cons

- Pros:
 - Simple
 - Low operational efforts
- Cons:
 - Low performance (high latency, reduced bandwidth, contention,...)
 - Reduced availability

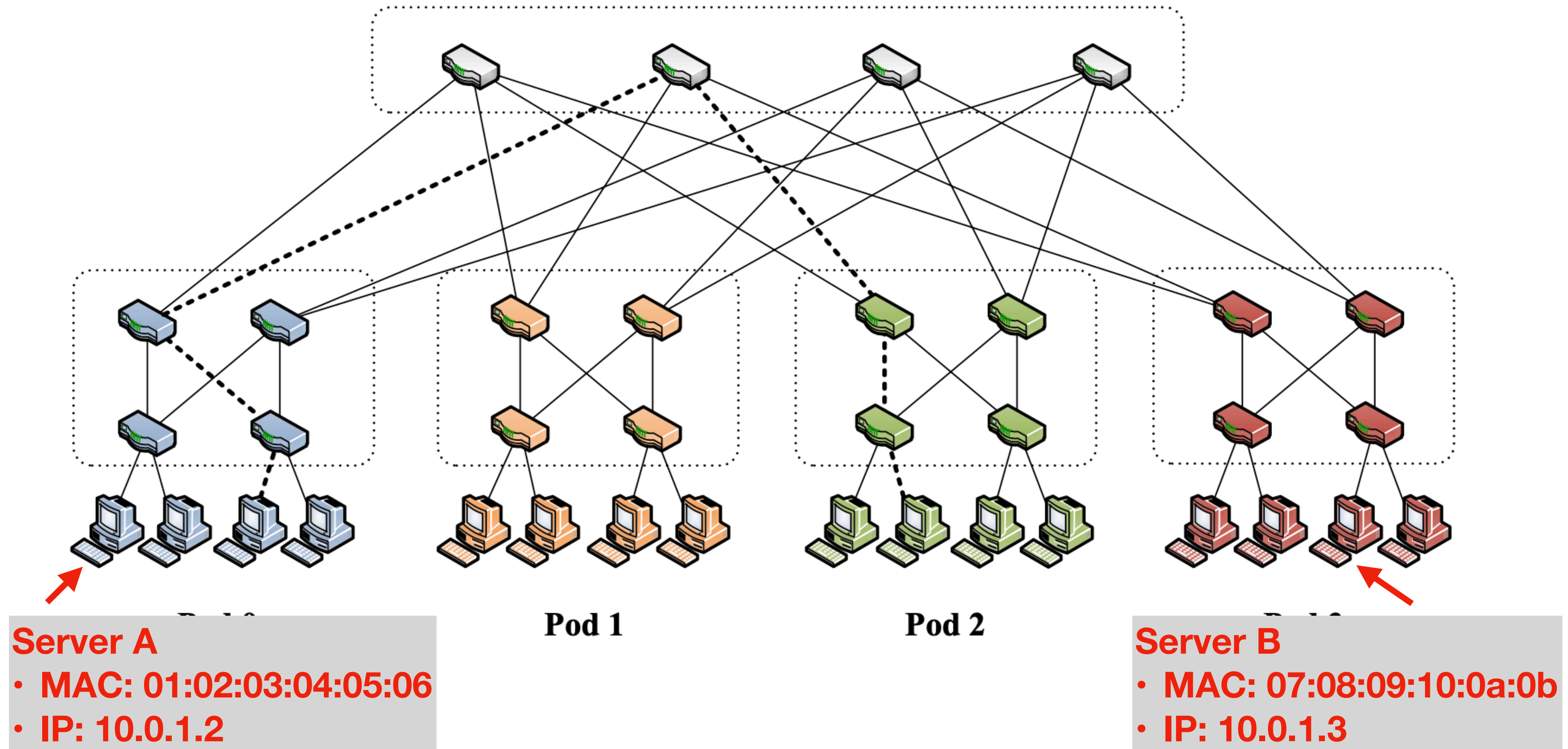
Proposal #2: Intra-domain Routing based on IP

- Just another enterprise networks



Proposal #2: Intra-domain Routing based on IP

- Apply the OSPF protocol



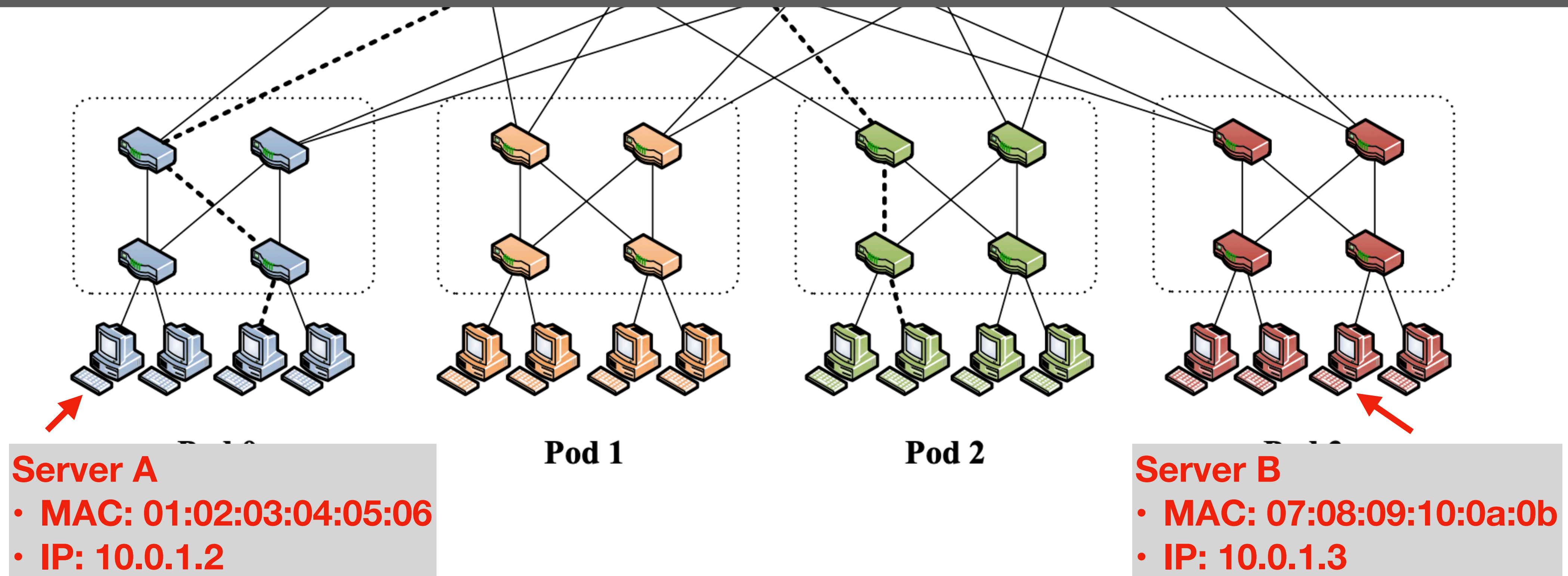
Recap: Open Shortest Path First (OSPF)

- Key idea: send the information of directly connected links to all nodes (in the network)
- Two mechanisms
 - Reliable flooding
 - Route calculation: Dijkstra algorithm
- Link state packet
 - ID of the node
 - Cost of link to each directly connected neighbor
 - Sequence number
 - Time-to-live (TTL)

Proposal #2: Intra-domain Routing based on IP

- Apply the OSPF protocol

Does it work?



Proposal #2: Intra-domain Routing based on IP

- Apply the OSPF protocol

Does it work?

Does it work well?

What are the issues?

Server A

- MAC: 01:02:03:04:05:06
- IP: 10.0.1.2

Pod 1

Pod 2

Server B

- MAC: 07:08:09:10:0a:0b
- IP: 10.0.1.3

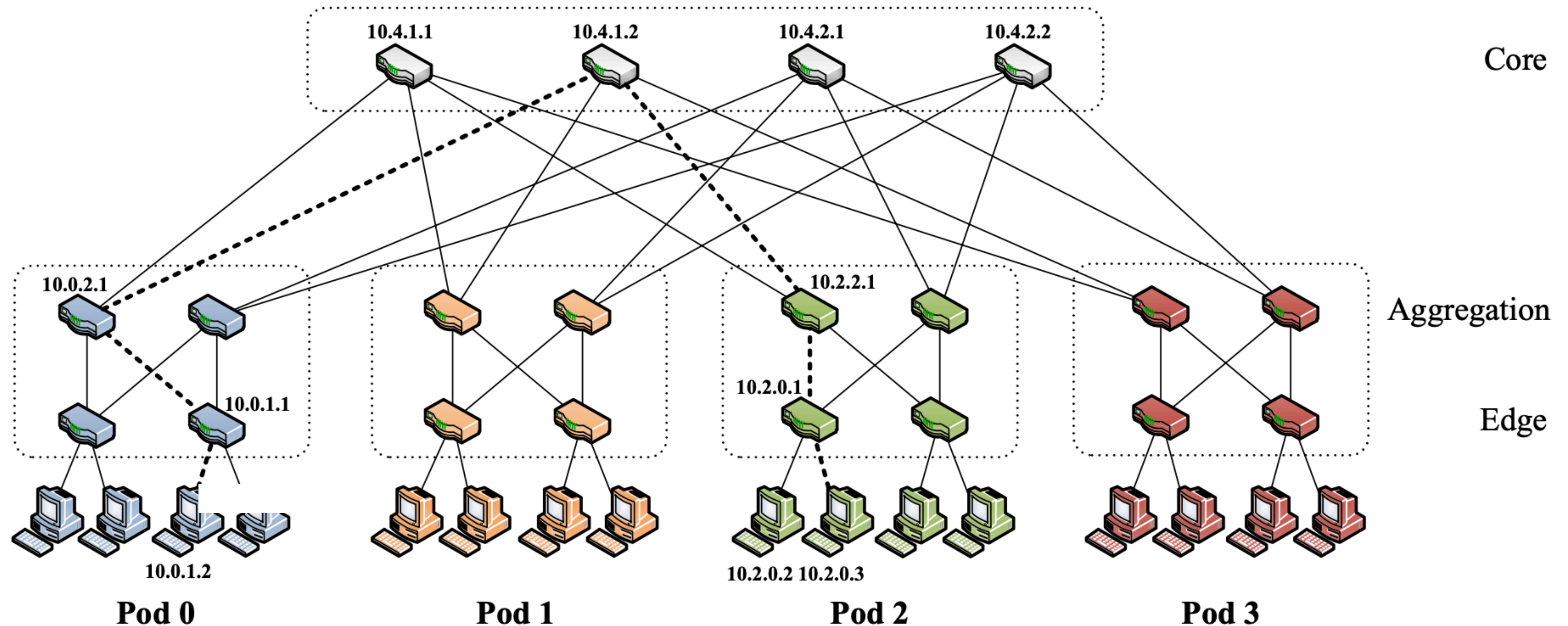
Proposal #2: Pros and Cons

- Pros:
 - Apply established techniques
 - Modest operational efforts
- Cons:
 - Low performance (high latency, reduced bandwidth, contention,...)
 - Reduced availability

**Key: The mechanism should inherently
encode the multiple-path routing capability!**

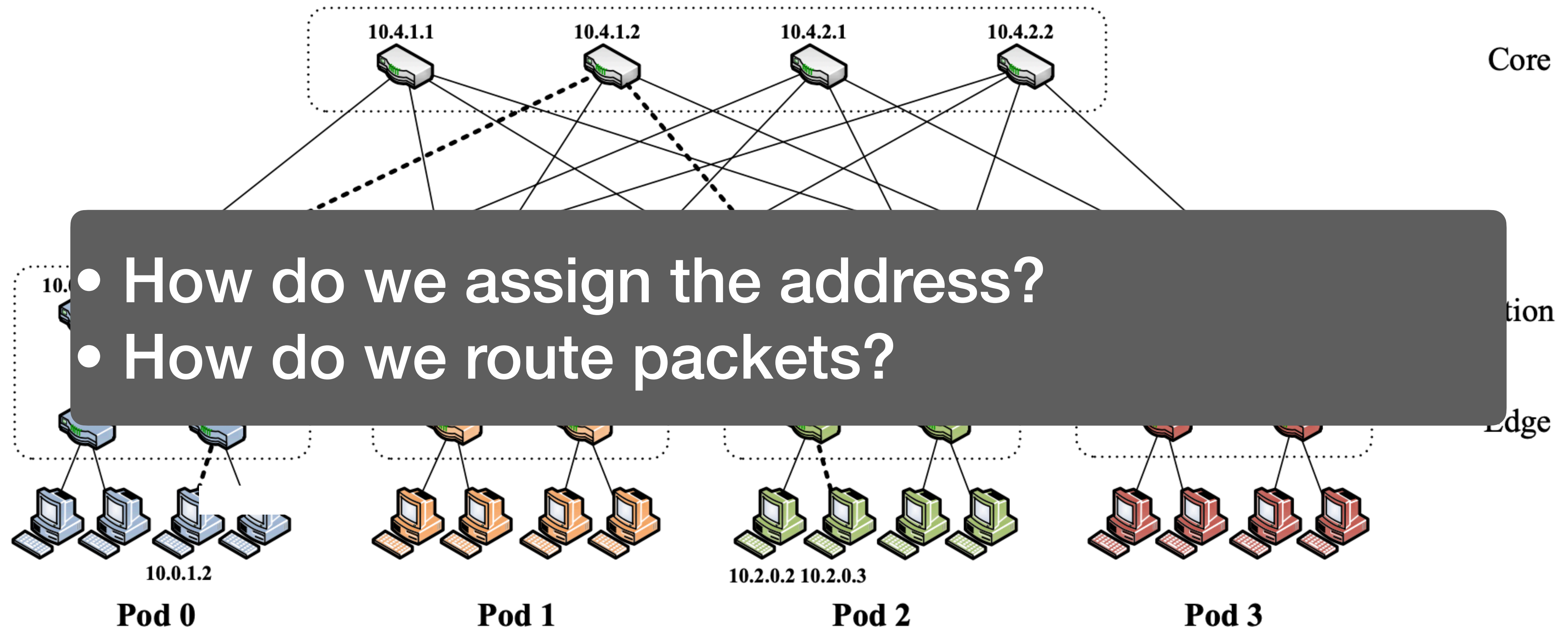
Proposal #3: Scalable DCNet (SIGCOMM'08)

- Key: co-design addressing and routing
- Target the k-array fat-tree topology, e.g., k=4



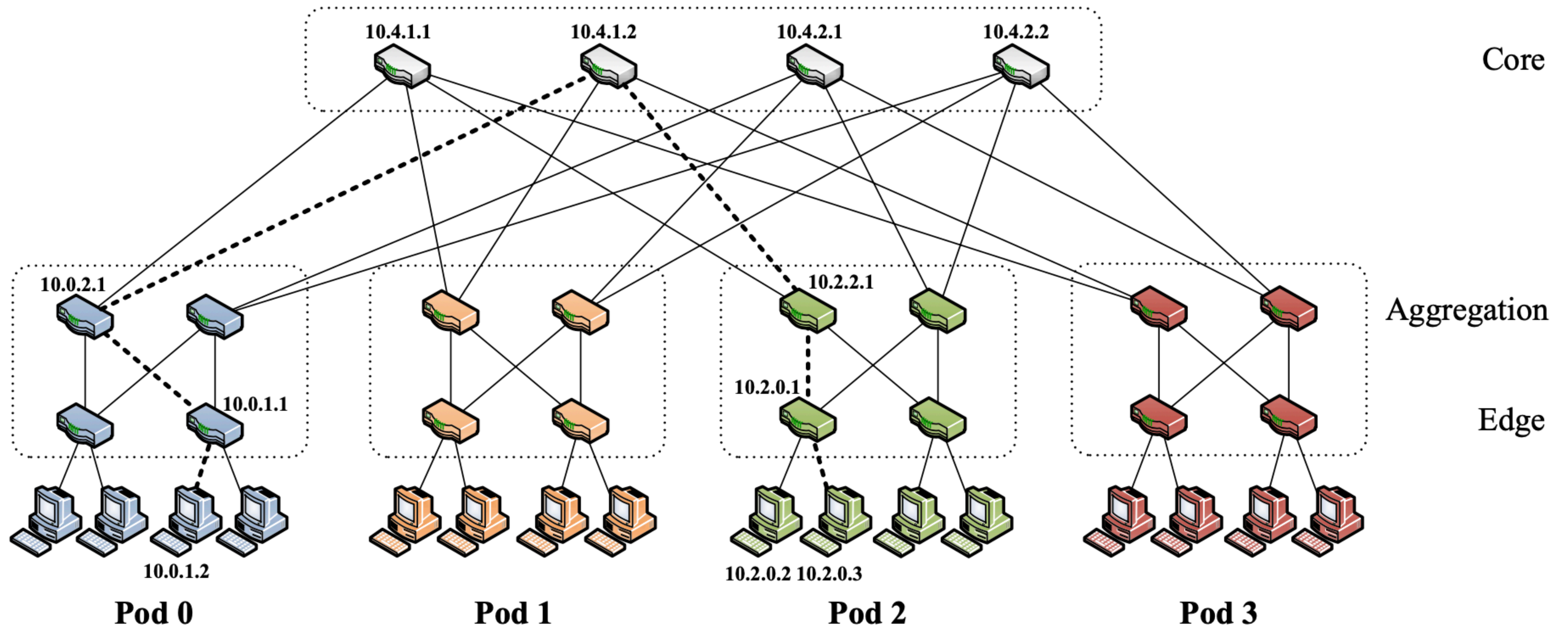
Proposal #3: Scalable DCNet (SIGCOMM'08)

- Key: co-design addressing and routing
- Target the k-array fat-tree topology, e.g., k=4



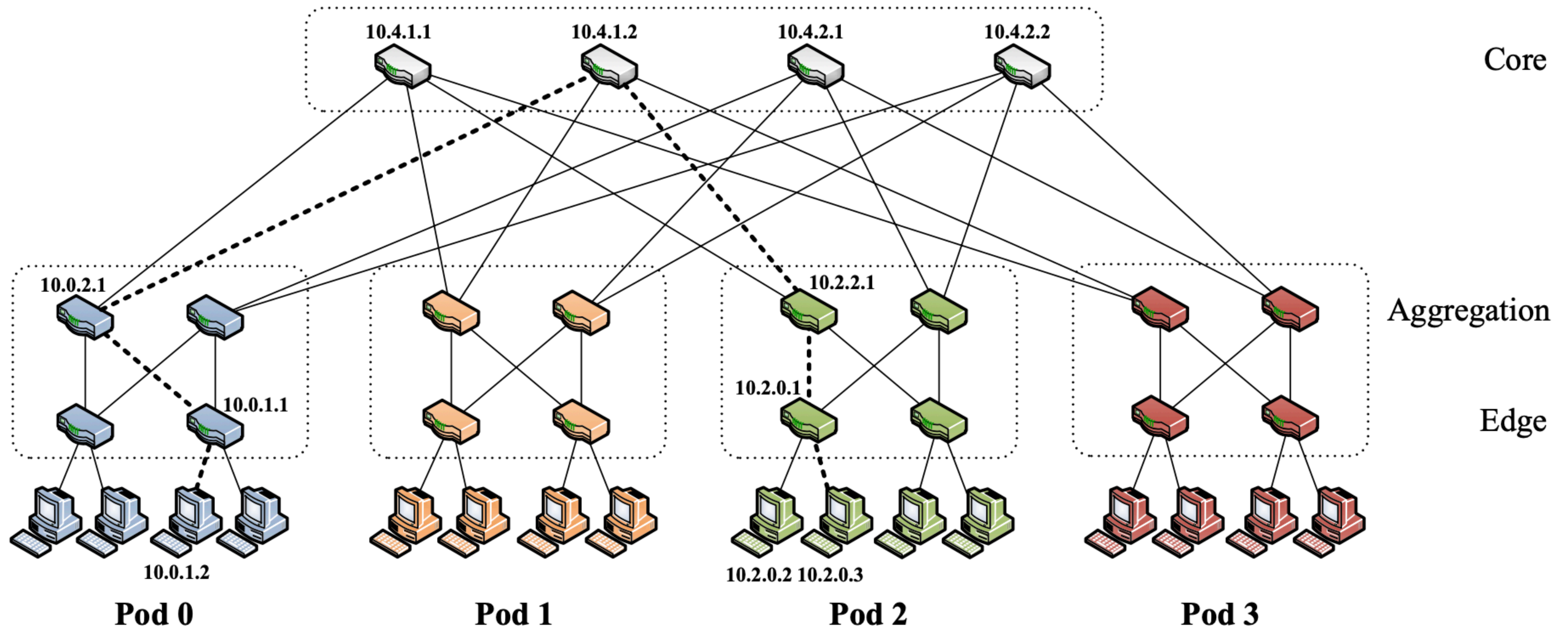
Addressing: pod switch

- 10.*pod*.*switch*.1
- pod:[0, k-1]



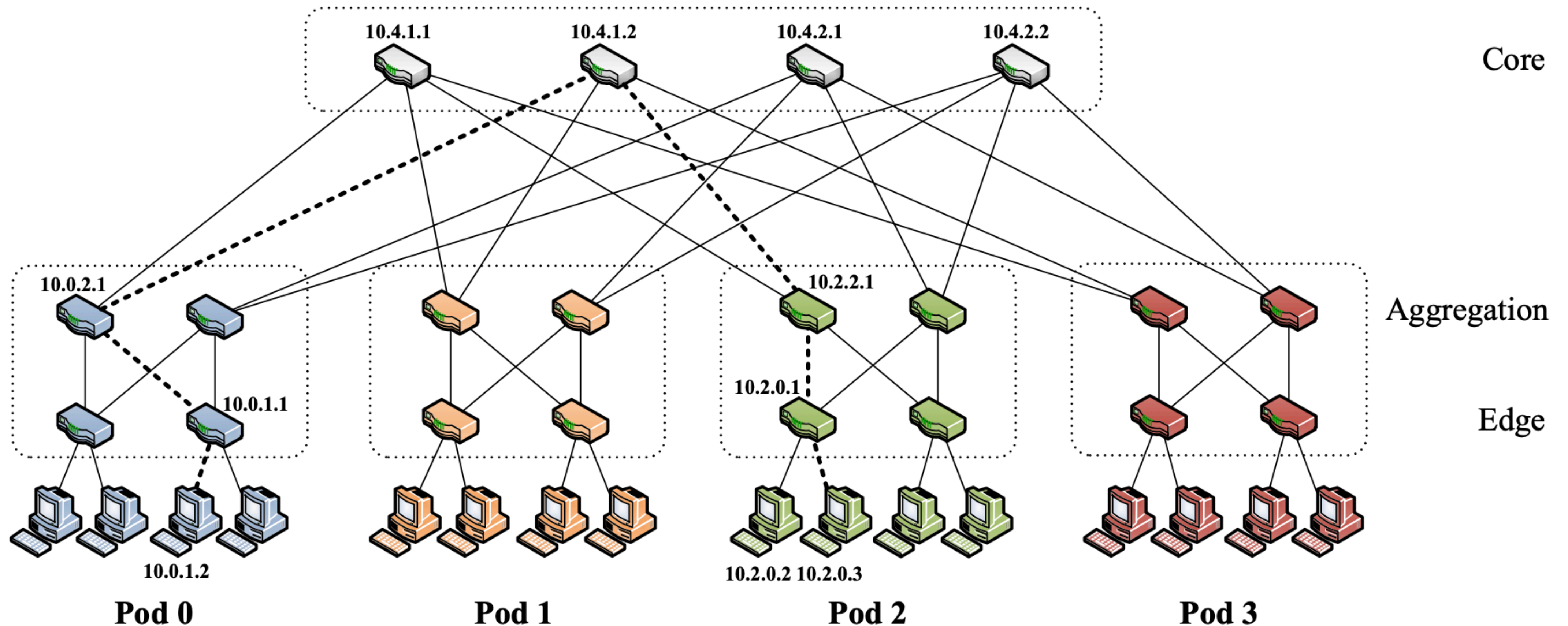
Addressing: pod switch

- 10.*pod*.*switch*.1
- switch:[0,k-1], starting from left to right, bottom to up



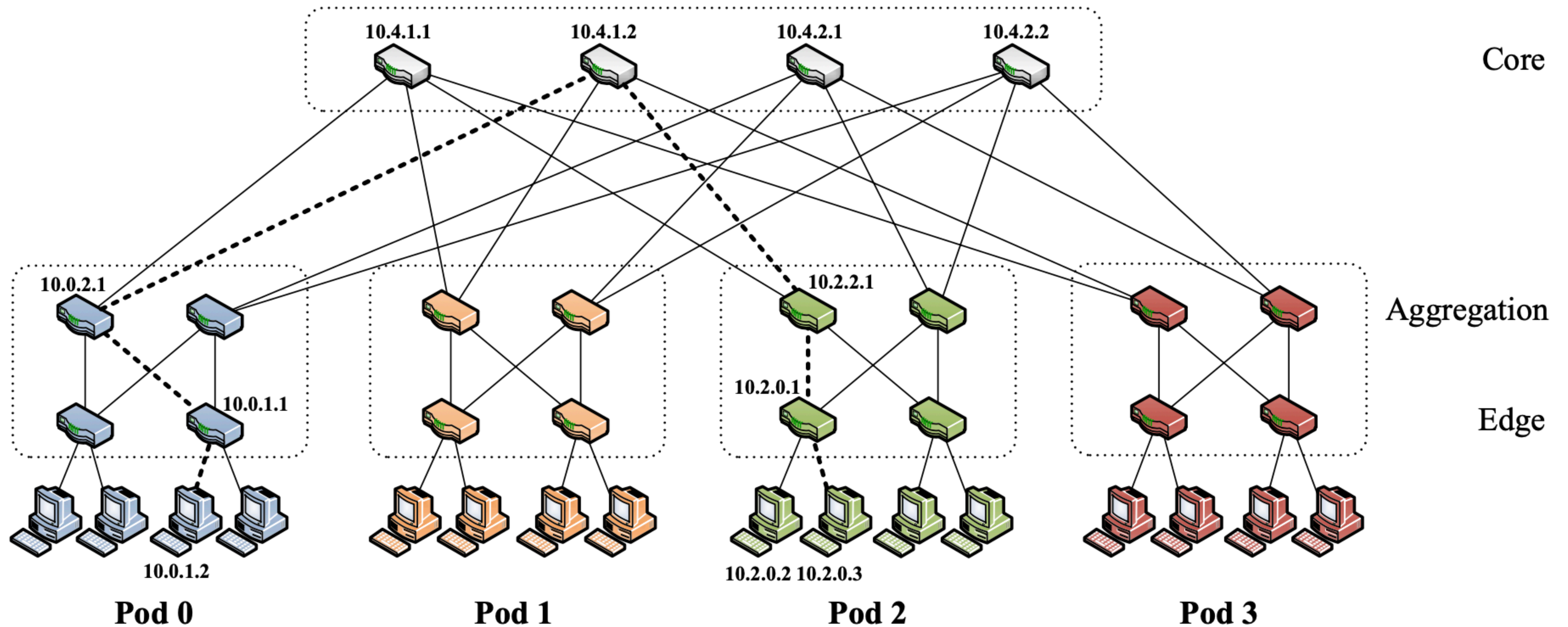
Addressing: core switch

- $10.k.j.i$
- j and i is the switch's coordinates in the $(k/2)^2$ core switch grid



Addressing: host

- 10.*pod*.*switch*.*ID*
- ID:[2,k/2+1], starting from left to right



Routing Overview

- Two-Level Routing Table
 - Primary routing table: (prefix, port)
 - Secondary routing table: (suffix, port)
- Primary Table
 - Left-handed, i.e., /m prefix masks
- Secondary Table
 - Right-handed, i.e., /m suffix masks

Prefix	Output port
10.2.0.0/24	0
10.2.1.0/24	1
0.0.0.0/0	

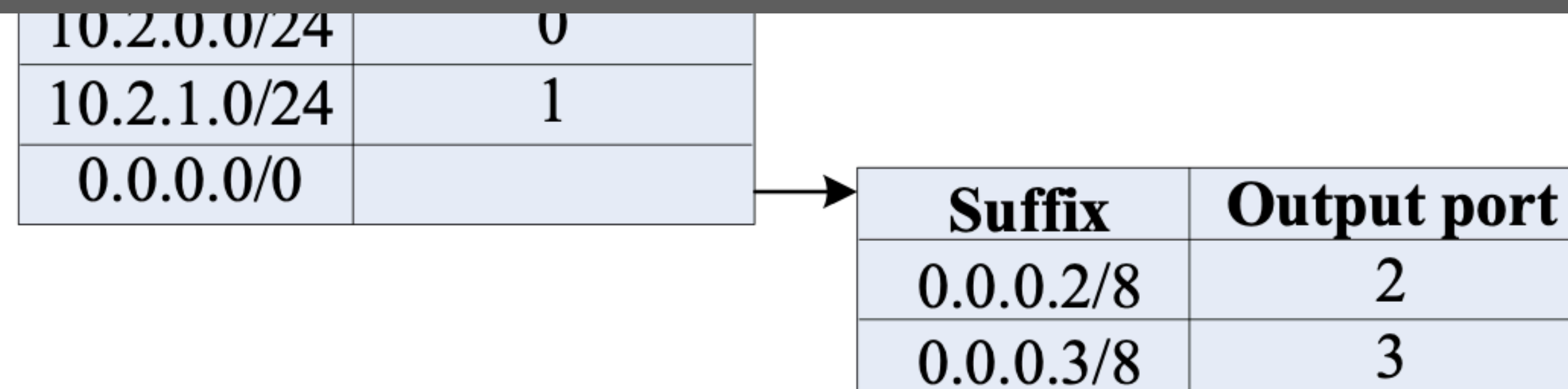


Suffix	Output port
0.0.0.2/8	2
0.0.0.3/8	3

Routing Overview

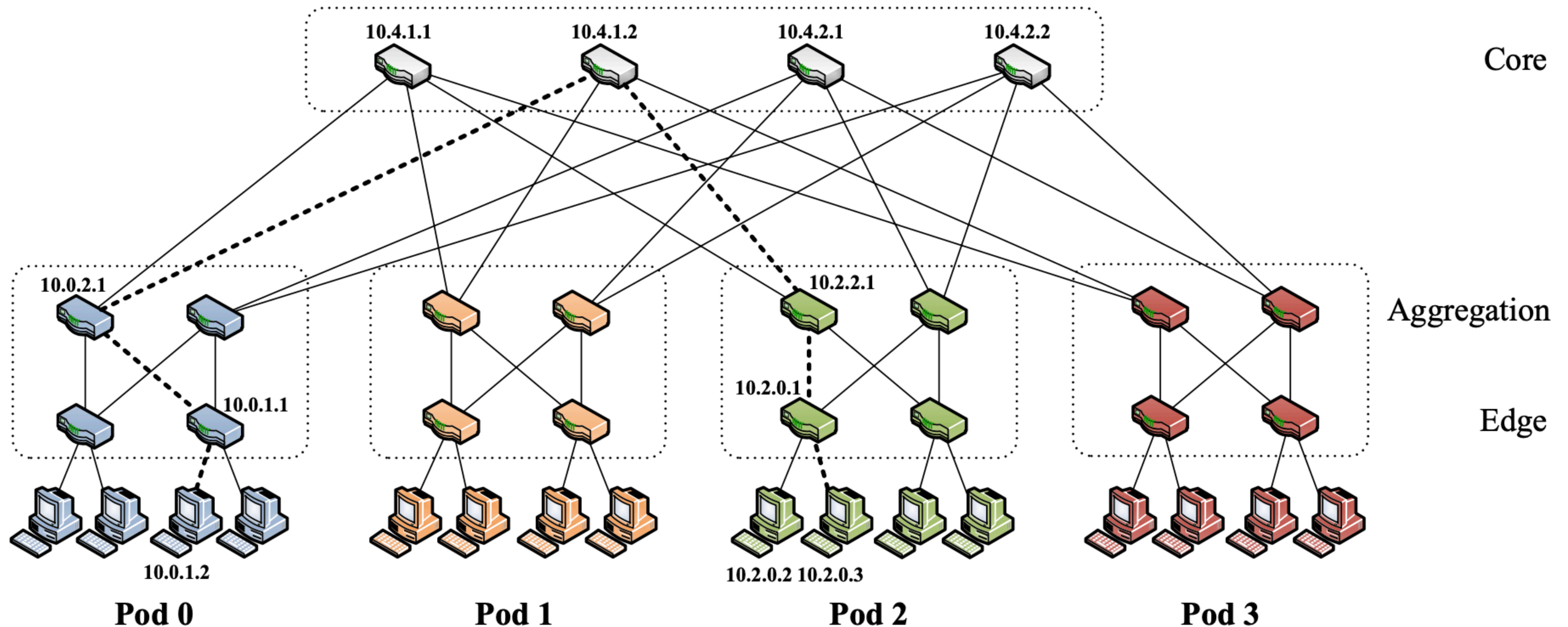
- Two-Level Routing Table
 - Primary routing table: (prefix, port)
 - Secondary routing table: (suffix, port)

How do we build the routing table?



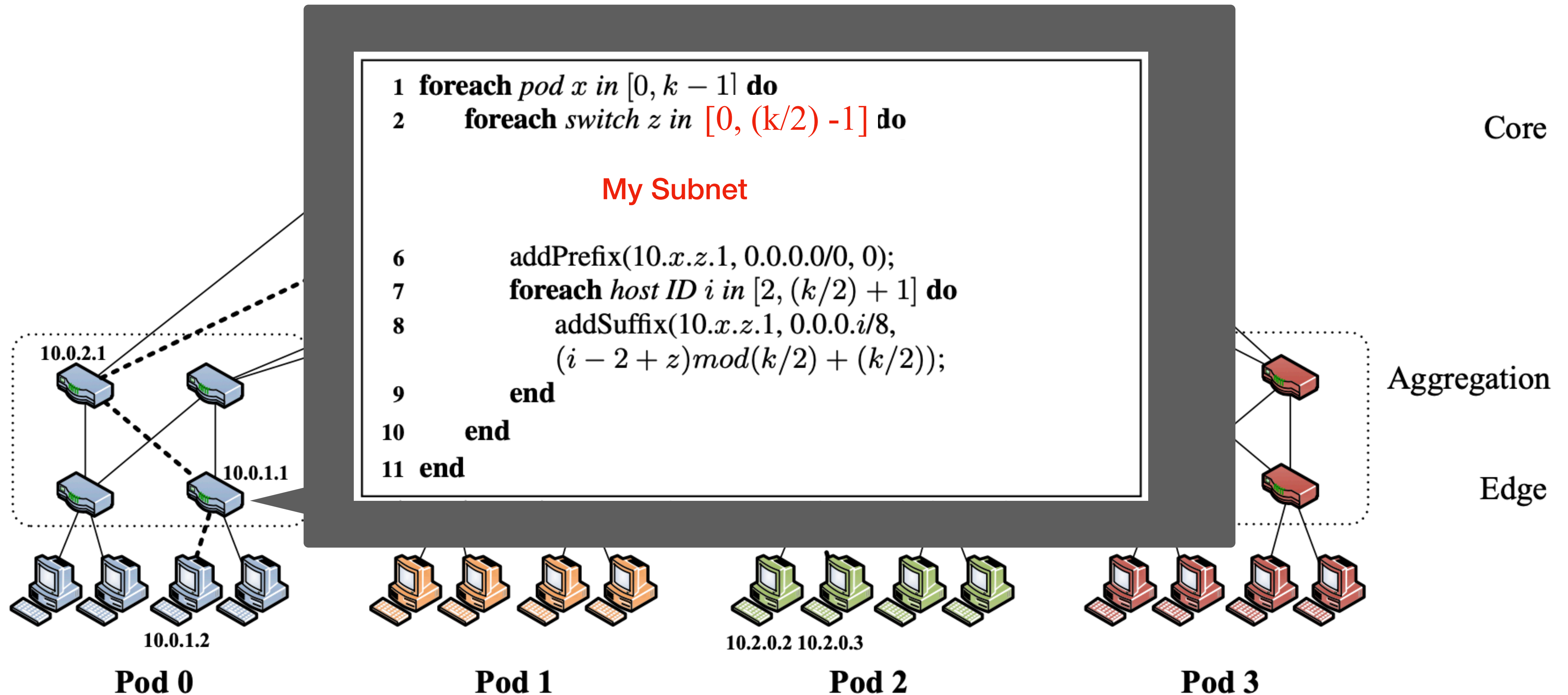
Routing @Edge Switch

- Two types of traffic: to hosts and to aggregation switches



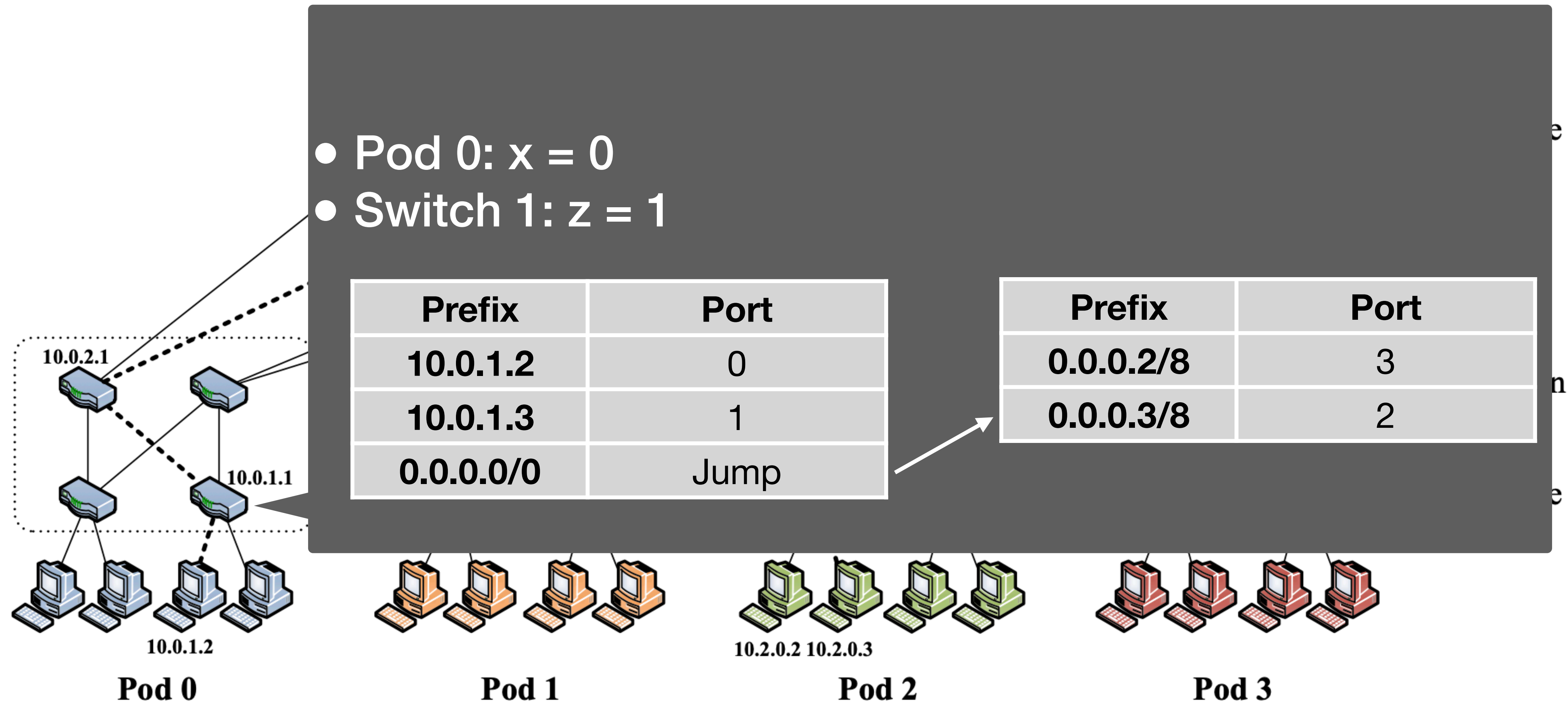
Routing @Edge Switch

- Two types of traffic: to hosts and to aggregation switches



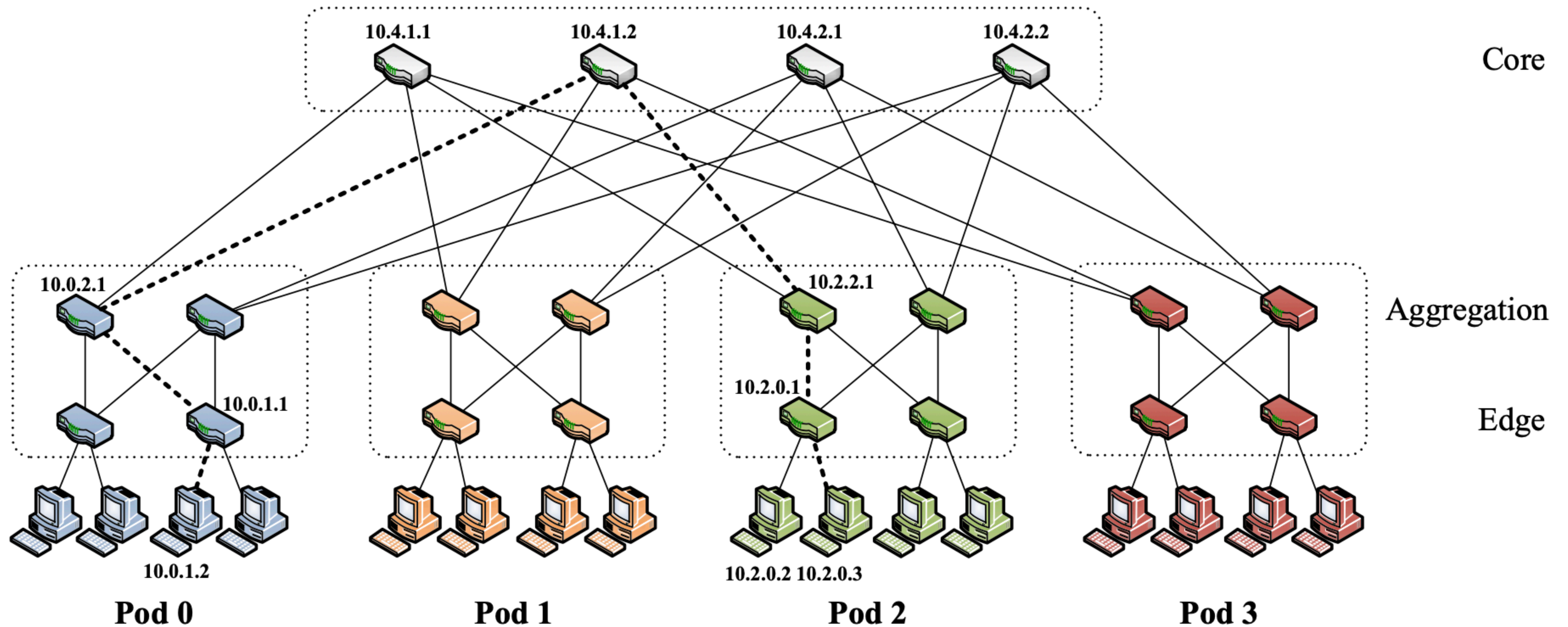
Routing @Edge Switch

- Two types of traffic: to hosts and to aggregation switches



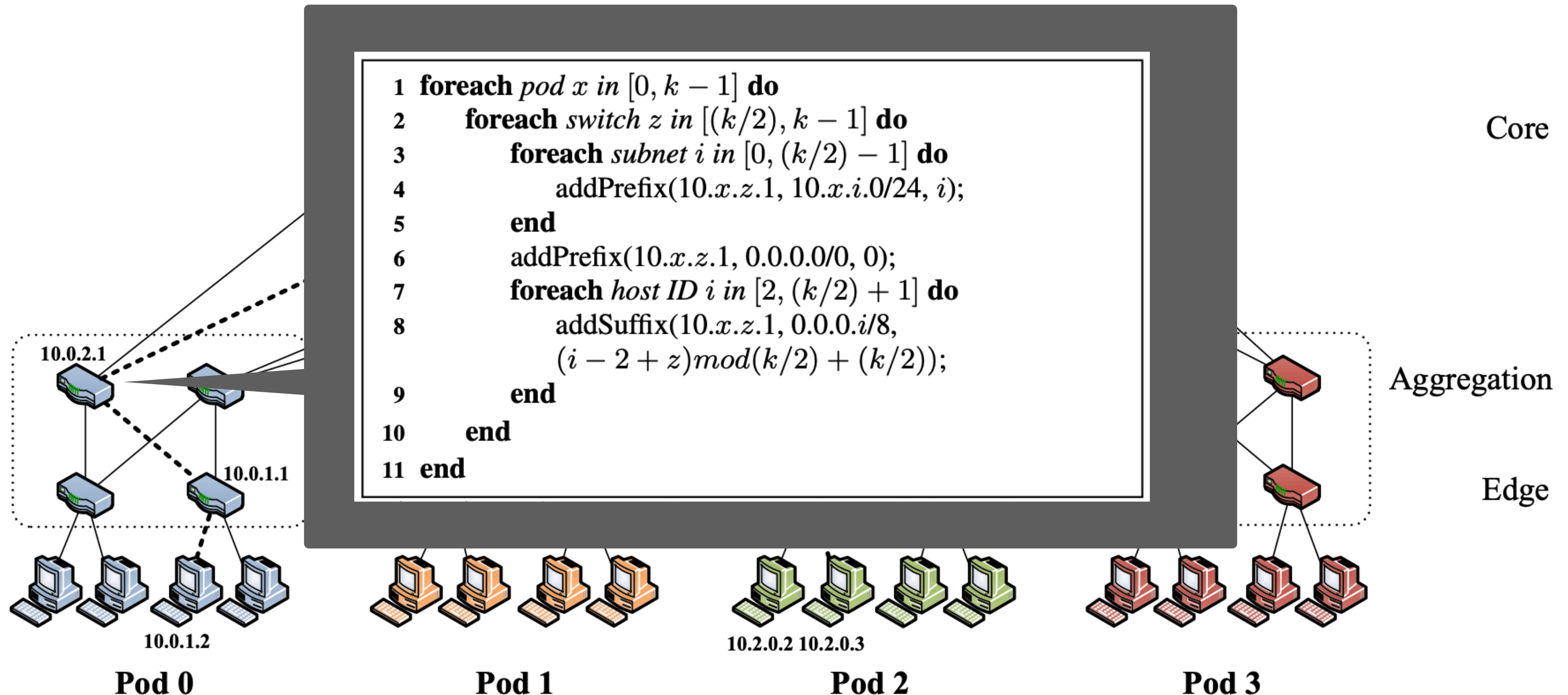
Routing @Aggregation Switch

- Two types of traffic: to core switches and to edge switches



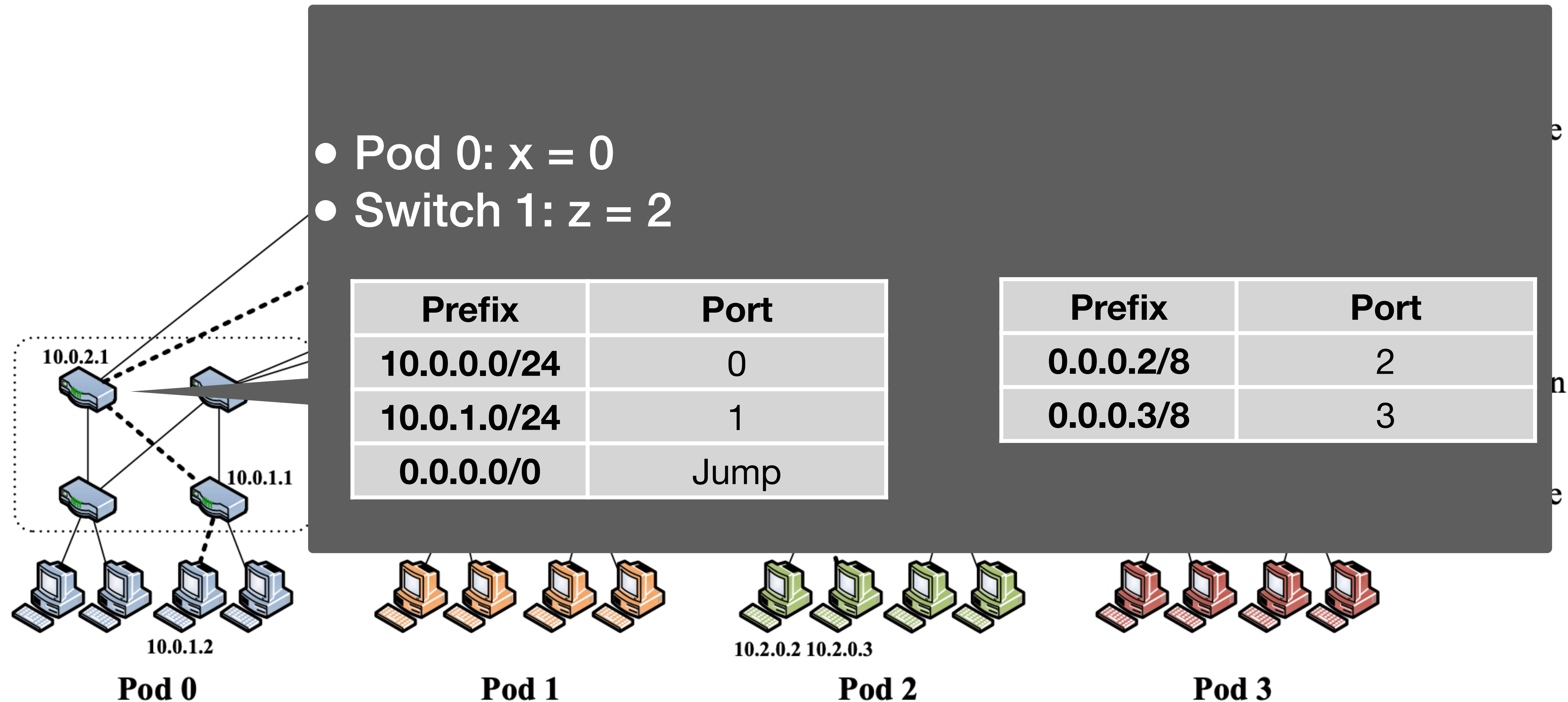
Routing @Aggregation Switch

- Two types of traffic: to core switches and to edge switches



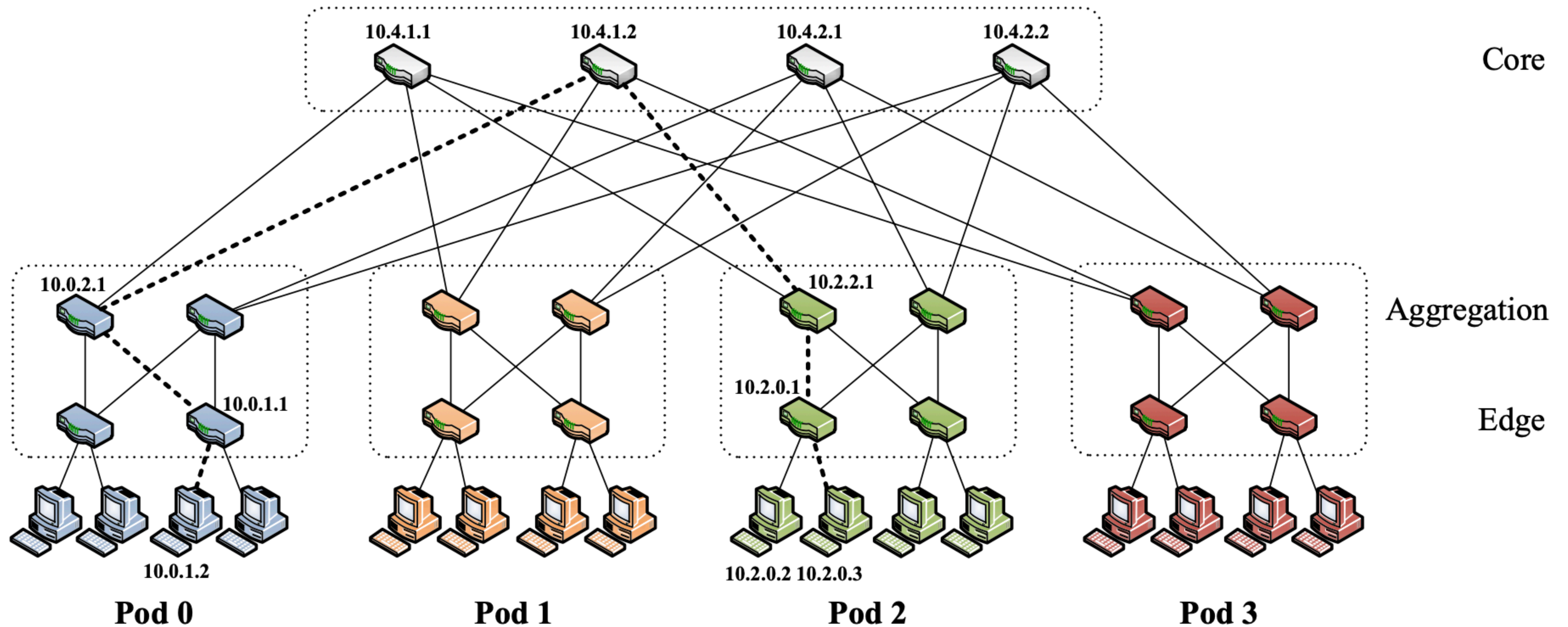
Routing @Aggregation Switch

- Two types of traffic: to core switches and to edge switches



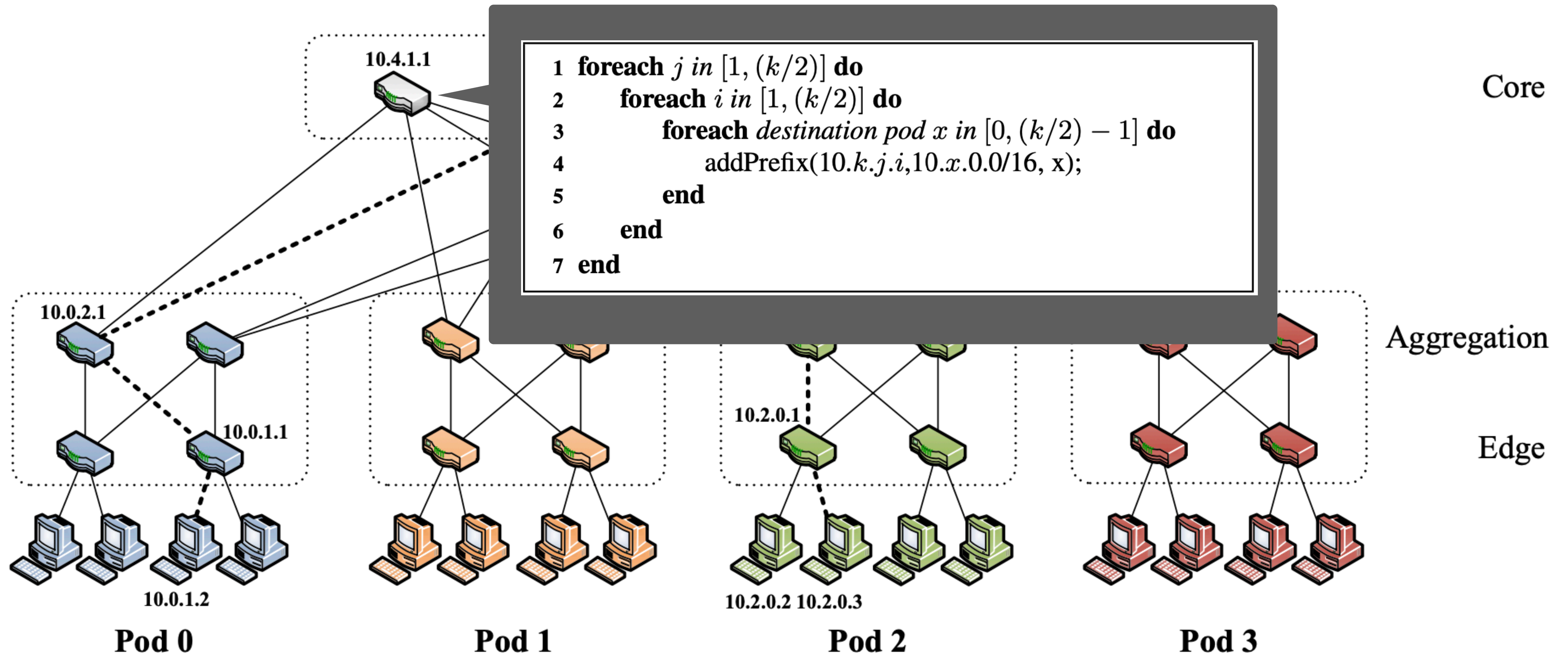
Routing @Core Switch

- One type of traffic: to aggregation switches



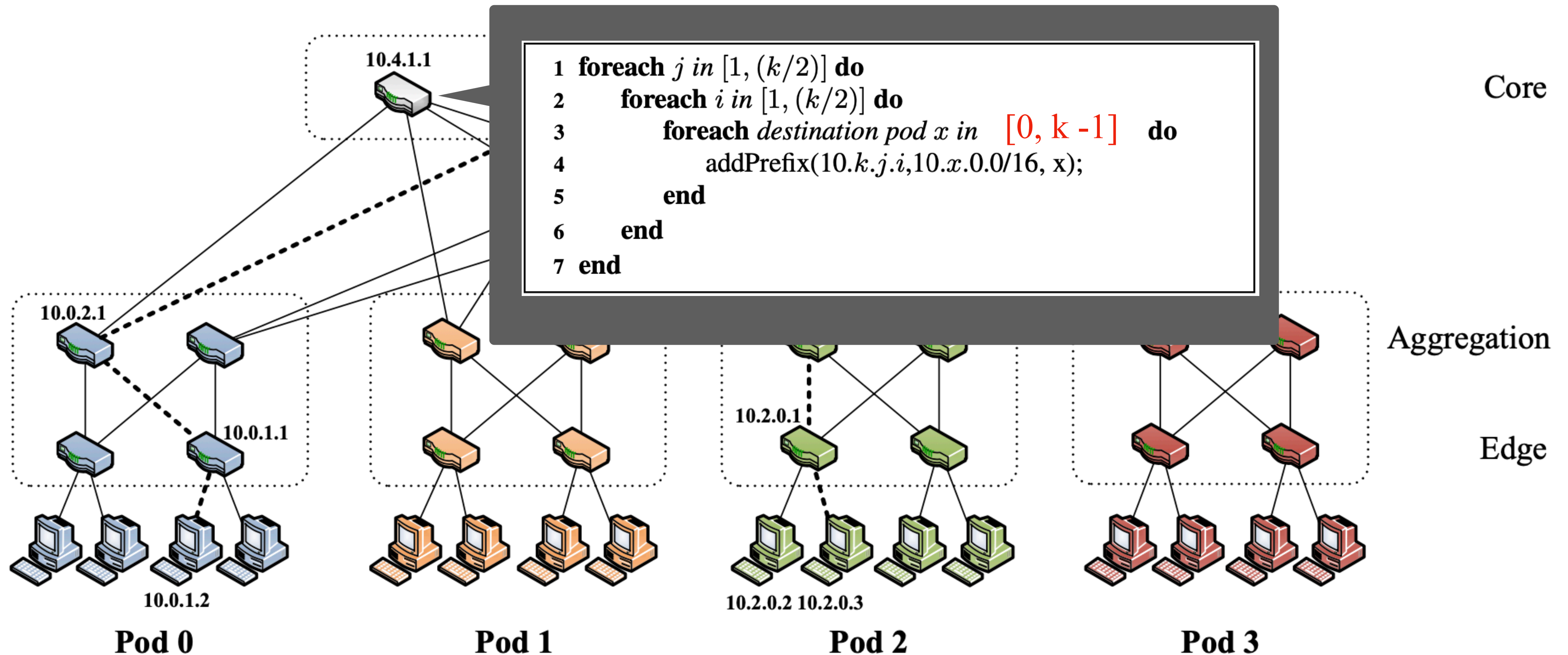
Routing @Core Switch

- One type of traffic: to aggregation switches



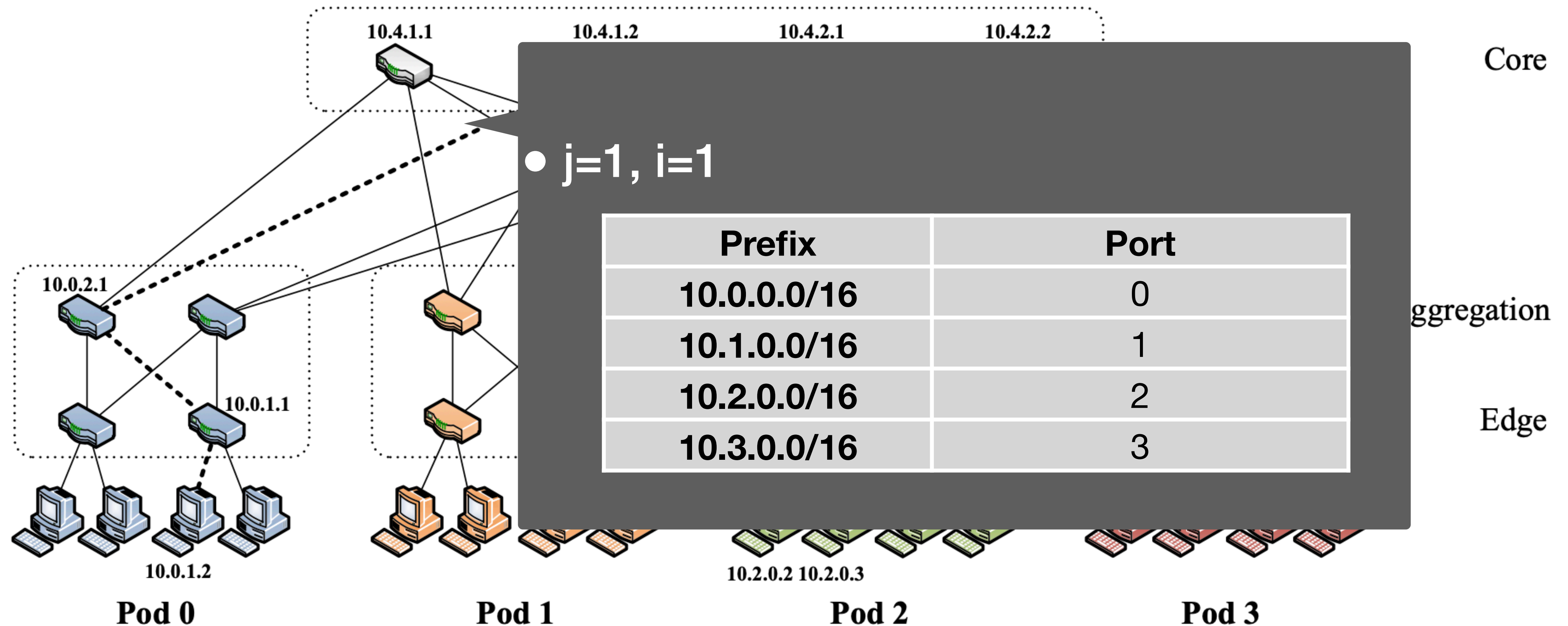
Routing @Core Switch

- One type of traffic: to aggregation switches



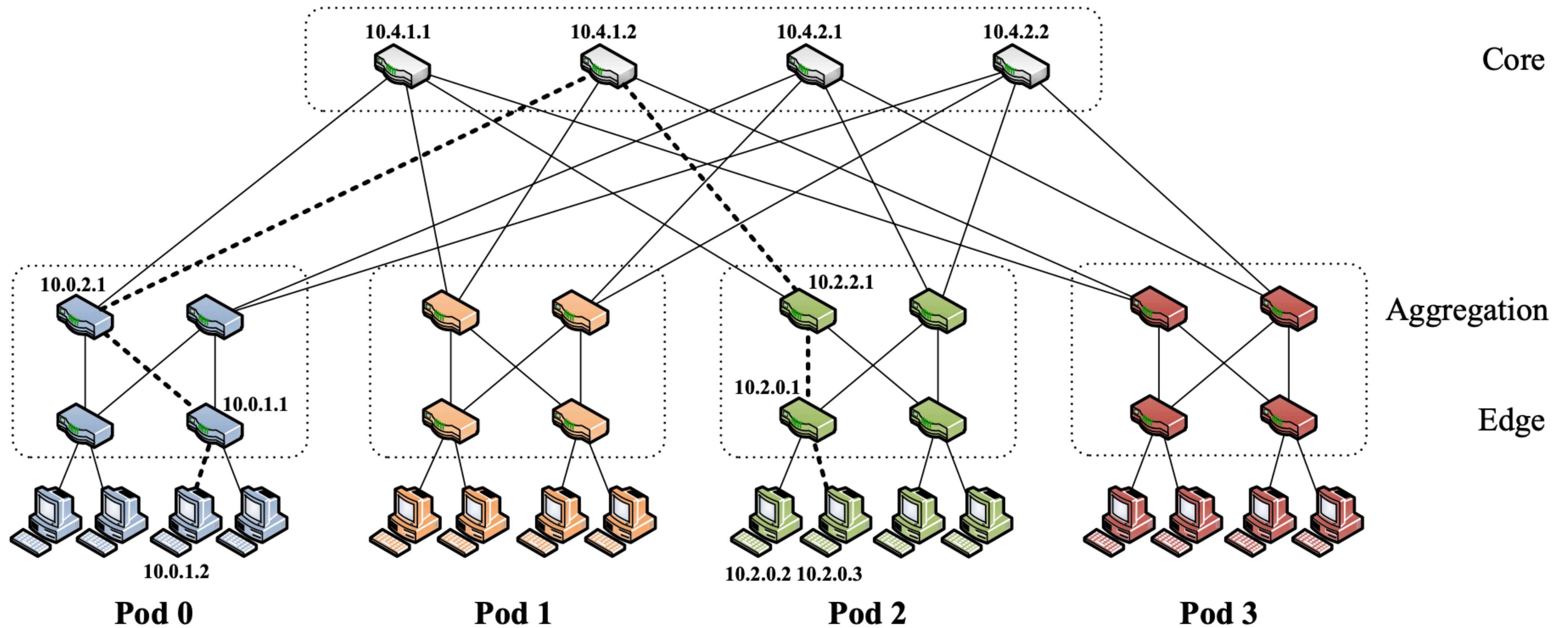
Routing @Core Switch

- One type of traffic: to aggregation switches



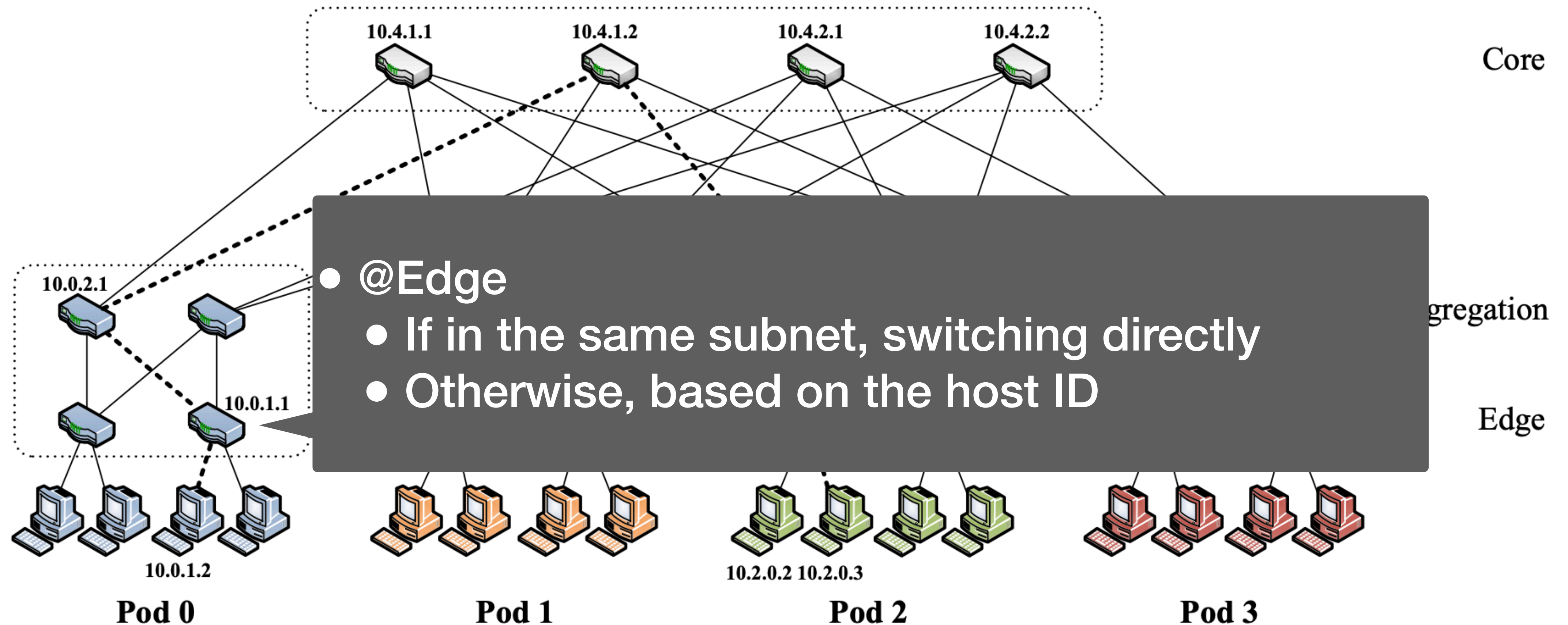
An Example

- 10.0.1.2 $\rightarrow$ 10.2.0.3



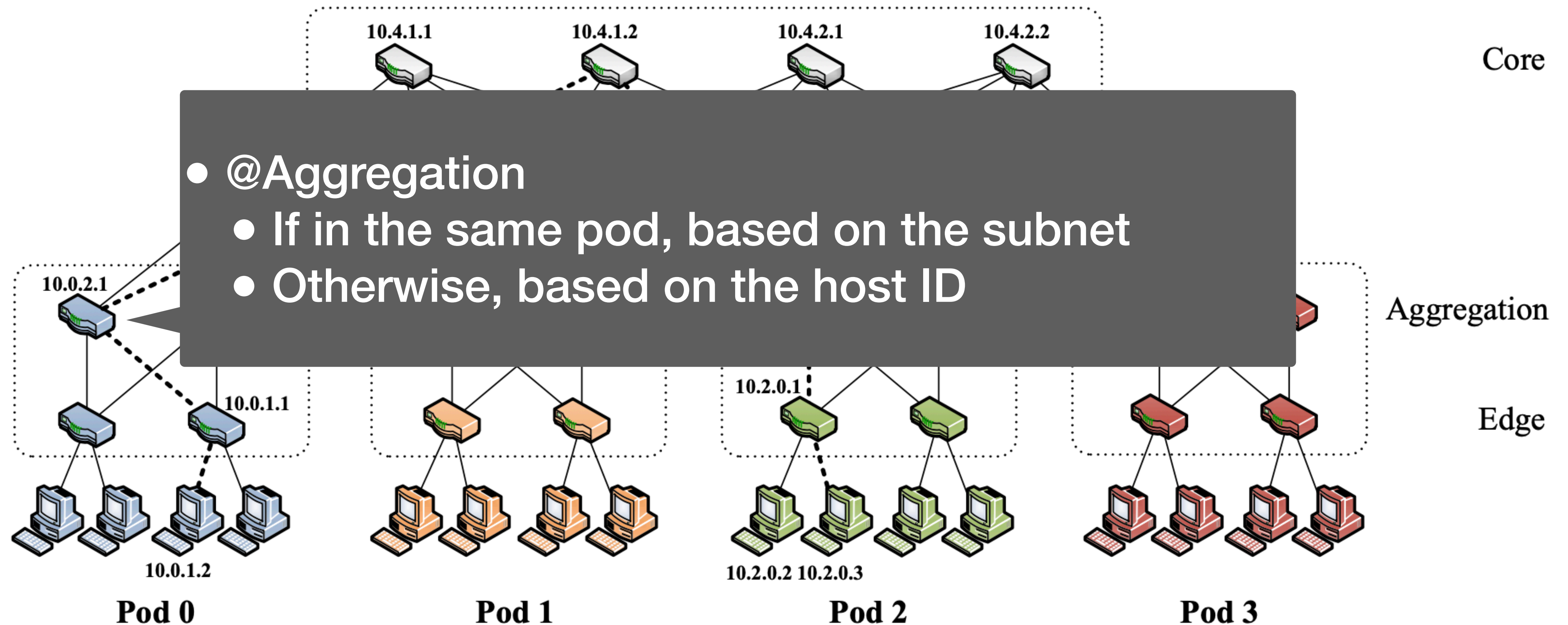
An Example

- 10.0.1.2 → 10.2.0.3



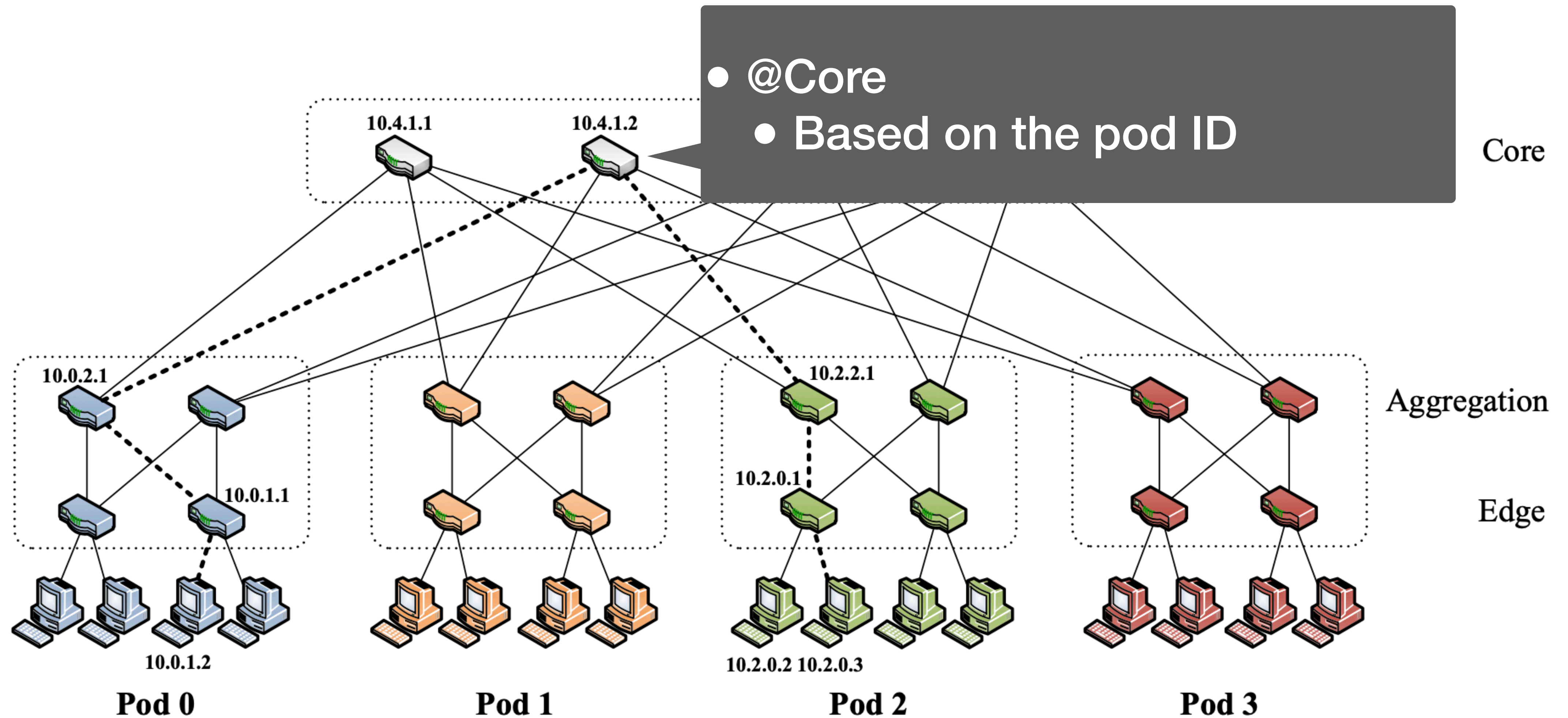
An Example

- 10.0.1.2 → 10.2.0.3



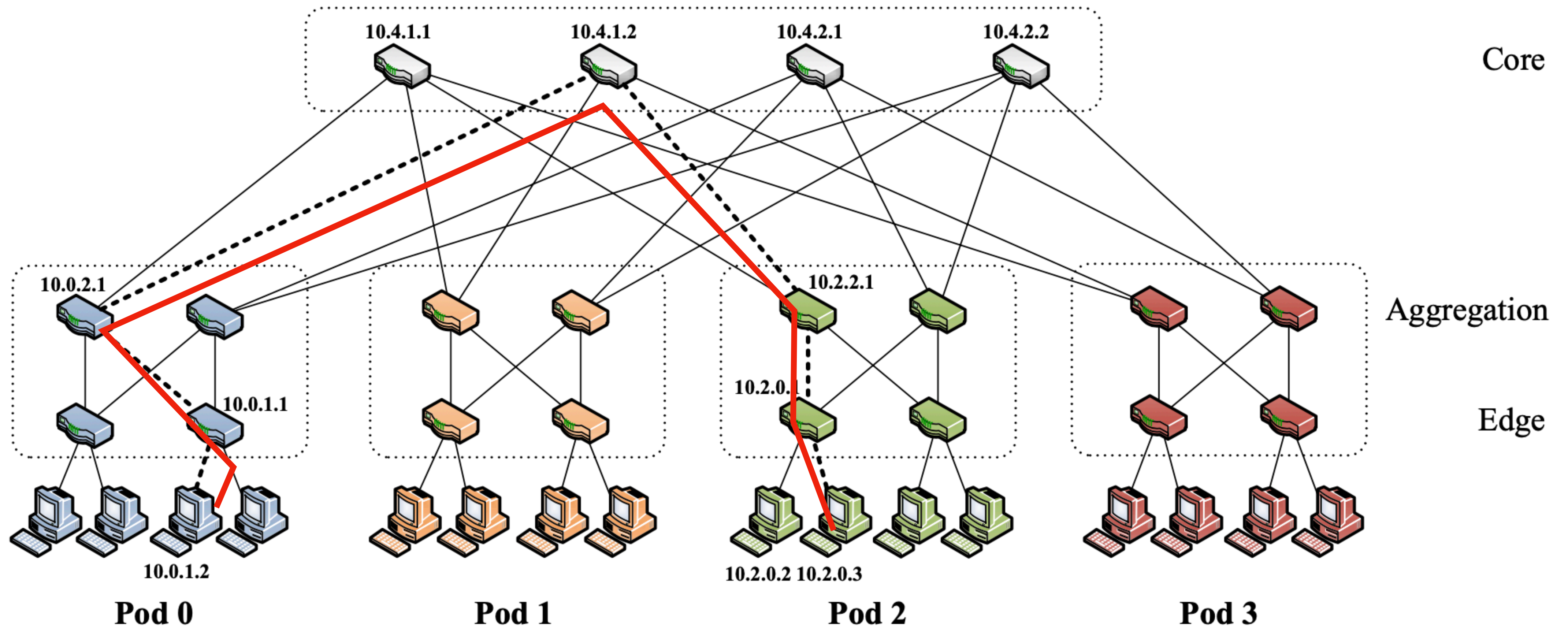
An Example

- 10.0.1.2 $\rightarrow$ 10.2.0.3



An Example

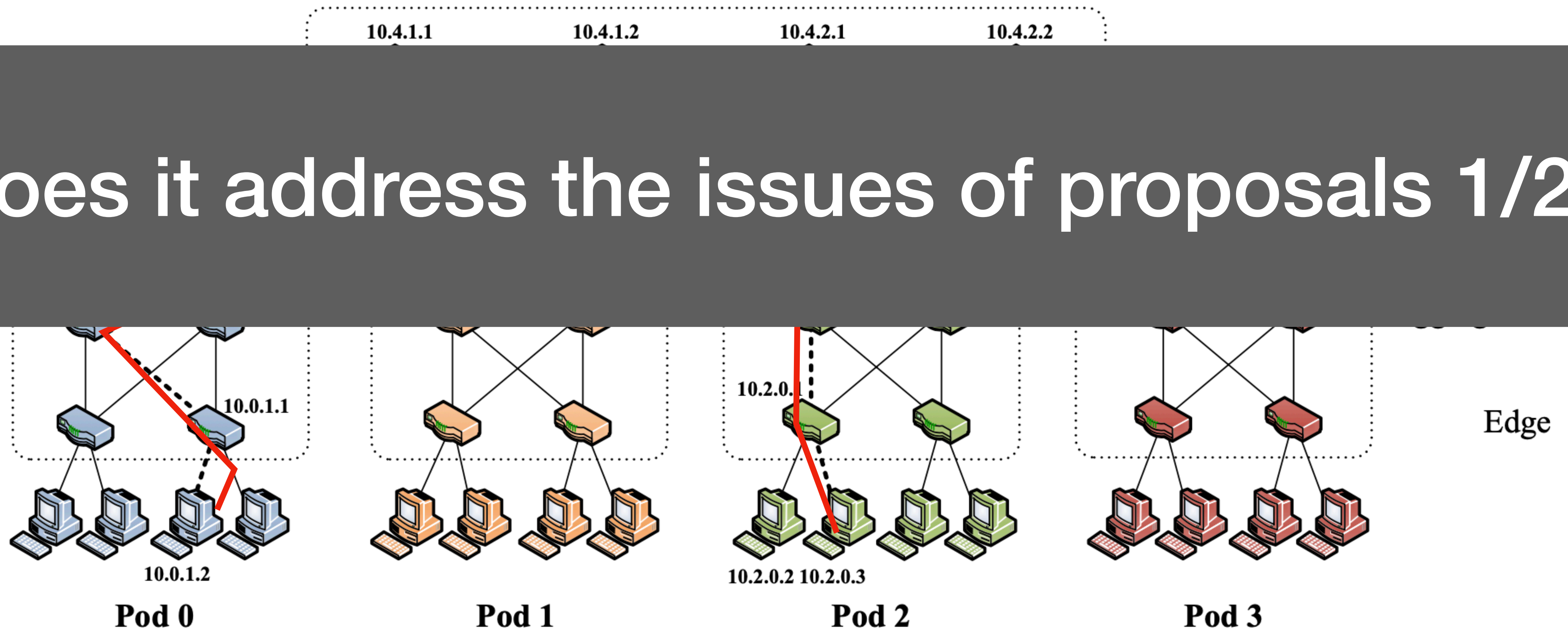
- 10.0.1.2 $\rightarrow$ 10.2.0.3



An Example

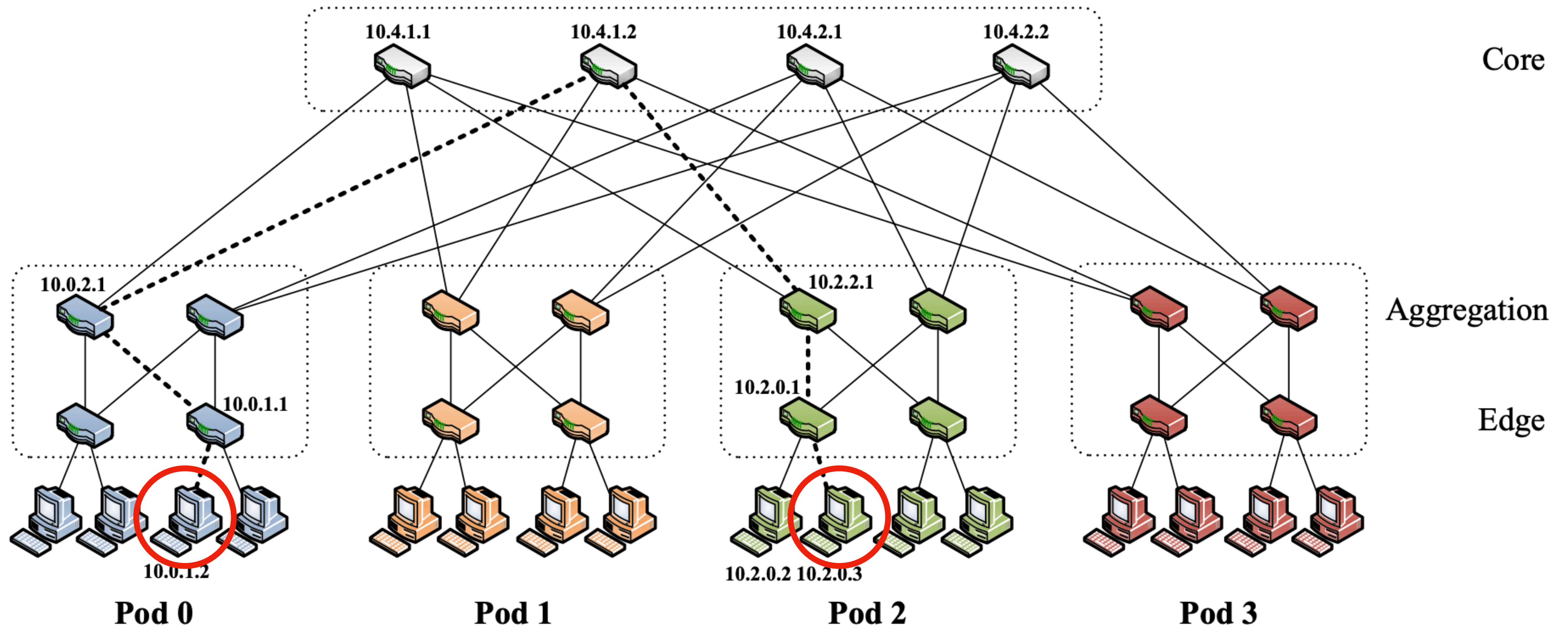
- 10.0.1.2 $\rightarrow$ 10.2.0.3

Does it address the issues of proposals 1/2?



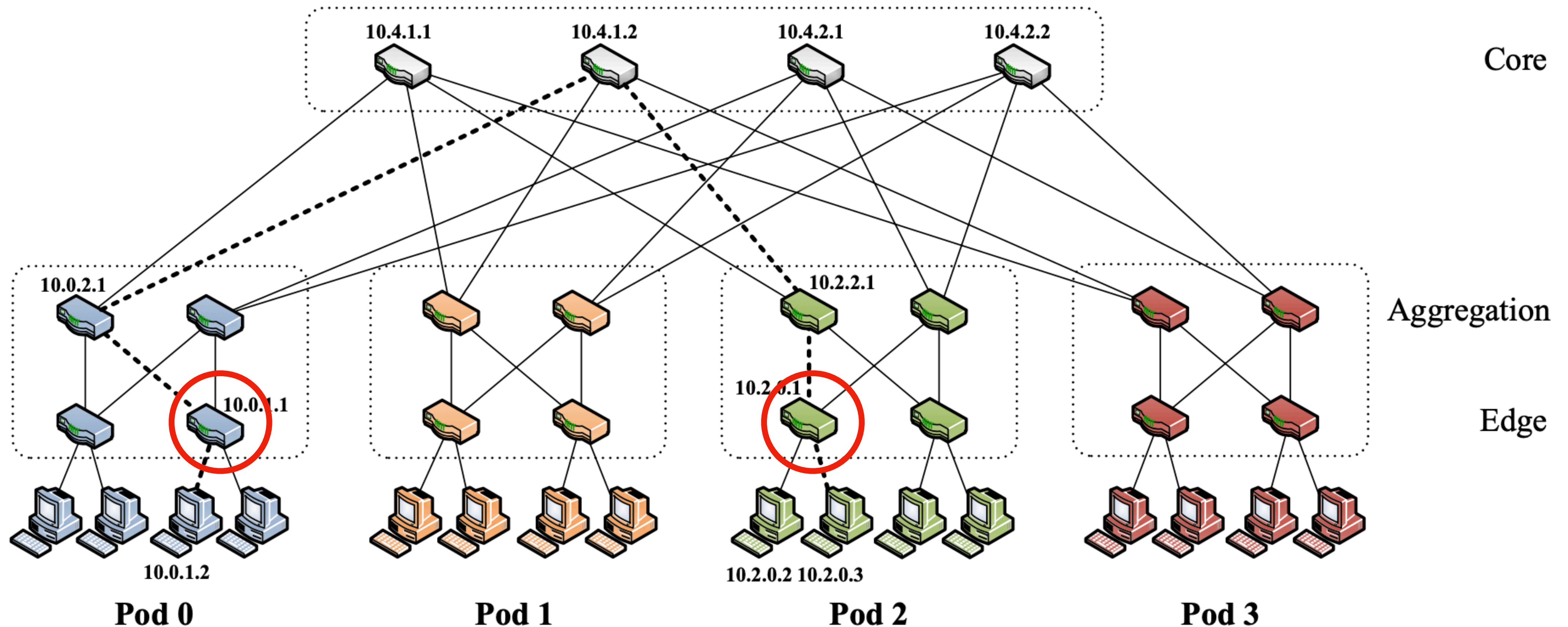
Proposal #3 Discussion

- How many communication paths do we use between two hosts?



Proposal #3 Discussion

- How many communication paths do we use between two edges?



**Bring multi-path communication between
two edge switches!**

Proposal #3: Pros and Cons

- Pros:
 - Better performance
 - Better availability
- Cons:
 - Design and implementation complexity (e.g., routable table maintenance)

Summary

- Today
 - Addressing and routing in data center networks (I)
- Next lecture
 - Addressing and routing in data center networks (II)
 - VL2 (Sigcomm'09)