KHOA CÔNG NGHỆ THÔNG TIN-ĐHKHTN
CSC11004 - MẠNG MÁY TÍNH NÂNG CAO

Software Defined Network

Lê Ngọc Sơn

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TP. HO CHI MINH
KHTN
cdio
4.0

fit@hcmus

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Outline

- What is SDN?
- OpenFlow basics
- Why is SDN happening now? (a brief history)

What is SDN?

Software Defined Network

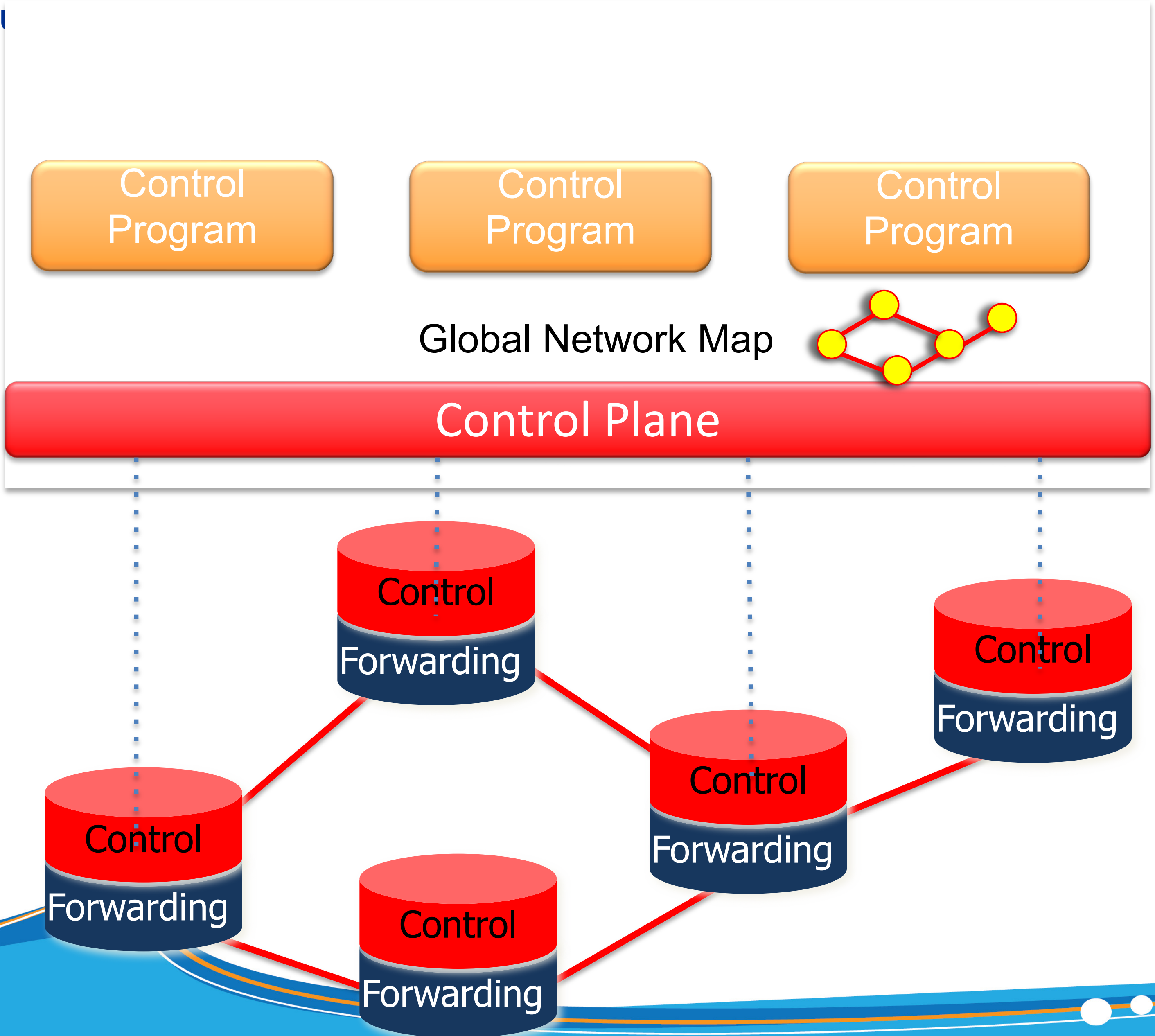
A network in which the control plane is physically separate from the data plane.

and

A single (logically centralized) control plane controls several forwarding devices.

Software Defined Network (SDN)

fit@hcmus



What You Said

- “Overall, the idea of SDN feels a little bit unsettling to me because it is proposing to change one of the main reasons for the success of computer networks: fully decentralized control. Once we introduce a centralized entity to control the network we have to make sure that it doesn’t fail, which I think is very difficult.”

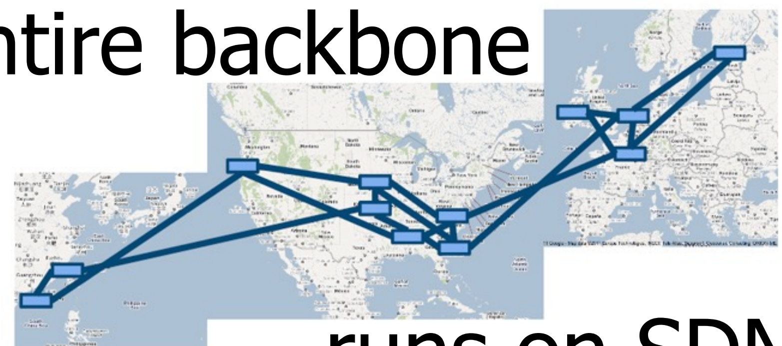
A Major Trend in Networking

OPEN NETWORKING FOUNDATION

Deutsche Telekom, facebook, Goldman Sachs, Google, Microsoft, NTT Communications, verizon, YAHOO!

Google

Entire backbone



runs on SDN

nicira

Bought for **$1.2 billion**
(mostly cash)

The Networking "Planes"

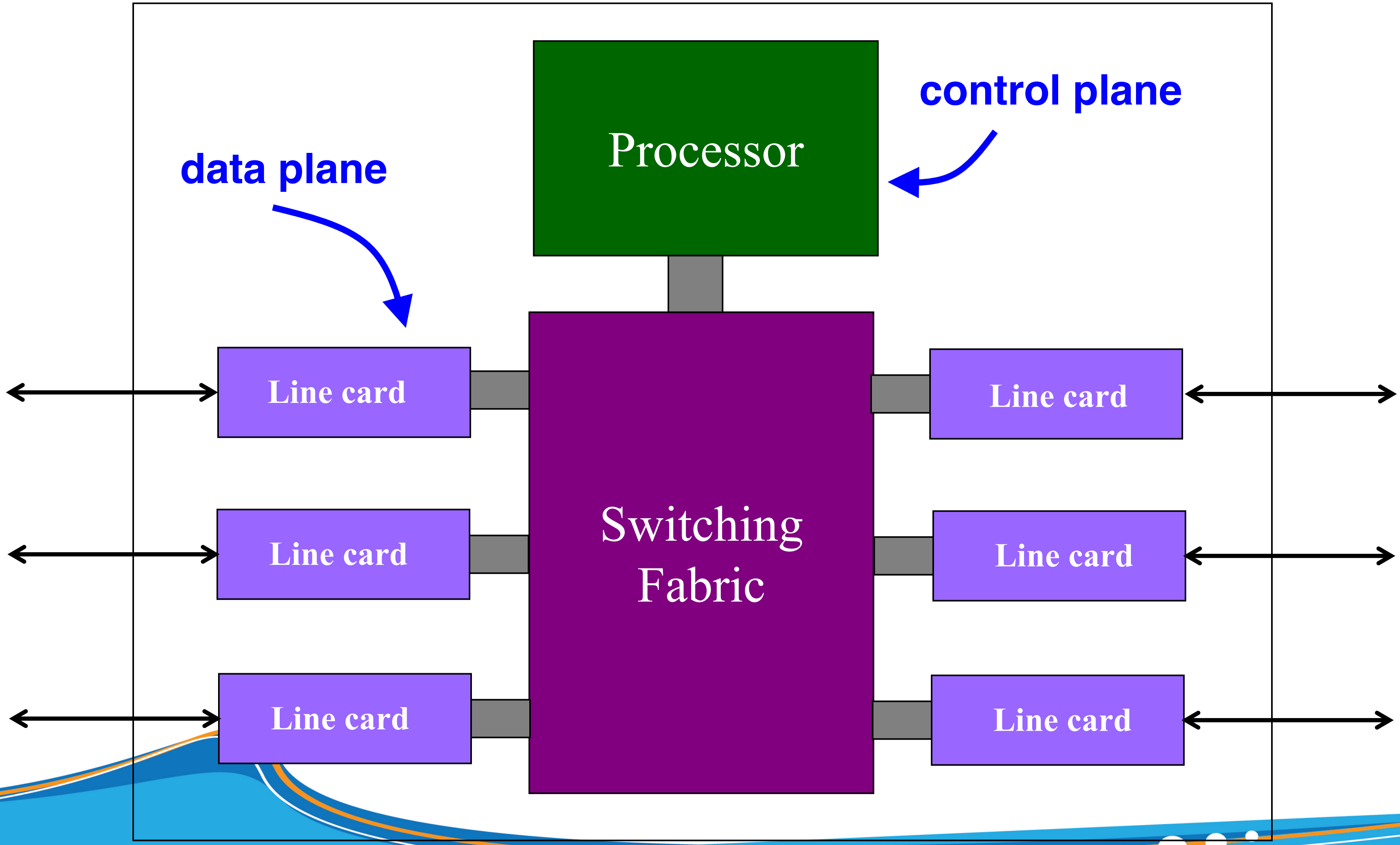
- **Data plane**: processing and delivery of packets with local forwarding state
  - Forwarding state + packet header → forwarding decision
  - Filtering, buffering, scheduling
- **Control plane**: computing the forwarding state in routers
  - Determines how and where packets are forwarded
  - Routing, traffic engineering, failure detection/recovery, …
- **Management plane**: configuring and tuning the network
  - Traffic engineering, ACL config, device provisioning, …

fit@hcmus

Timescales

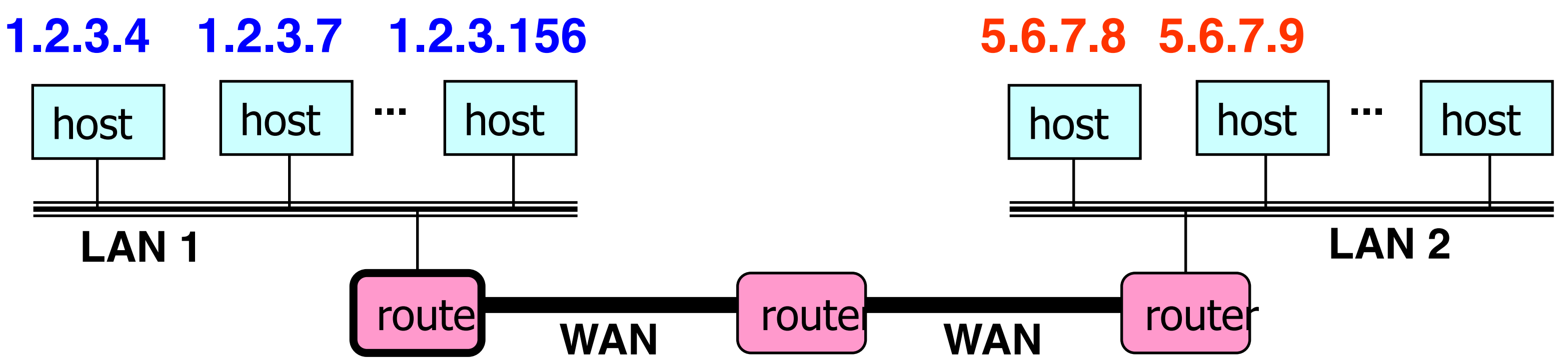
	Data	Control	Management
Time-scale	Packet (nsec)	Event (10 msec to sec)	Human (min to hours)
Location	Linecard hardware	Router software	Humans or scripts

Data and Control Planes



Data Plane

- Streaming algorithms on packets
  - Matching on some header bits
  - Perform some actions
- Example: IP Forwarding



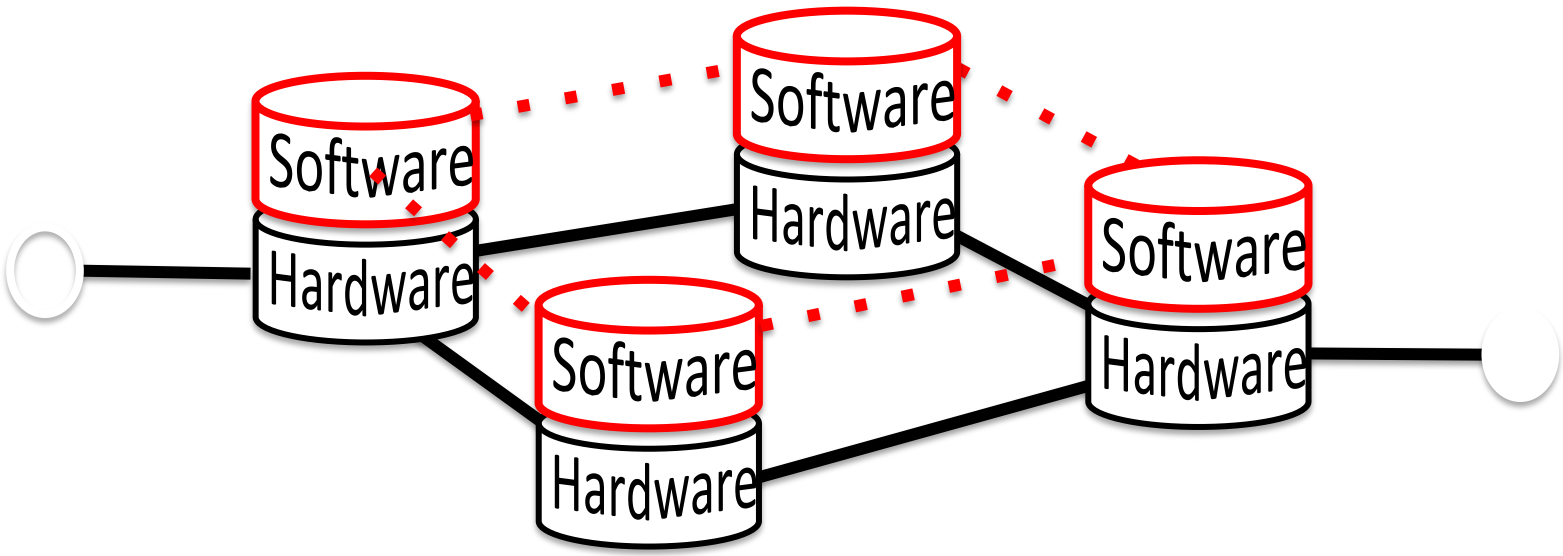
1.2.3.0/24	←
5.6.7.0/24	→

forwarding table

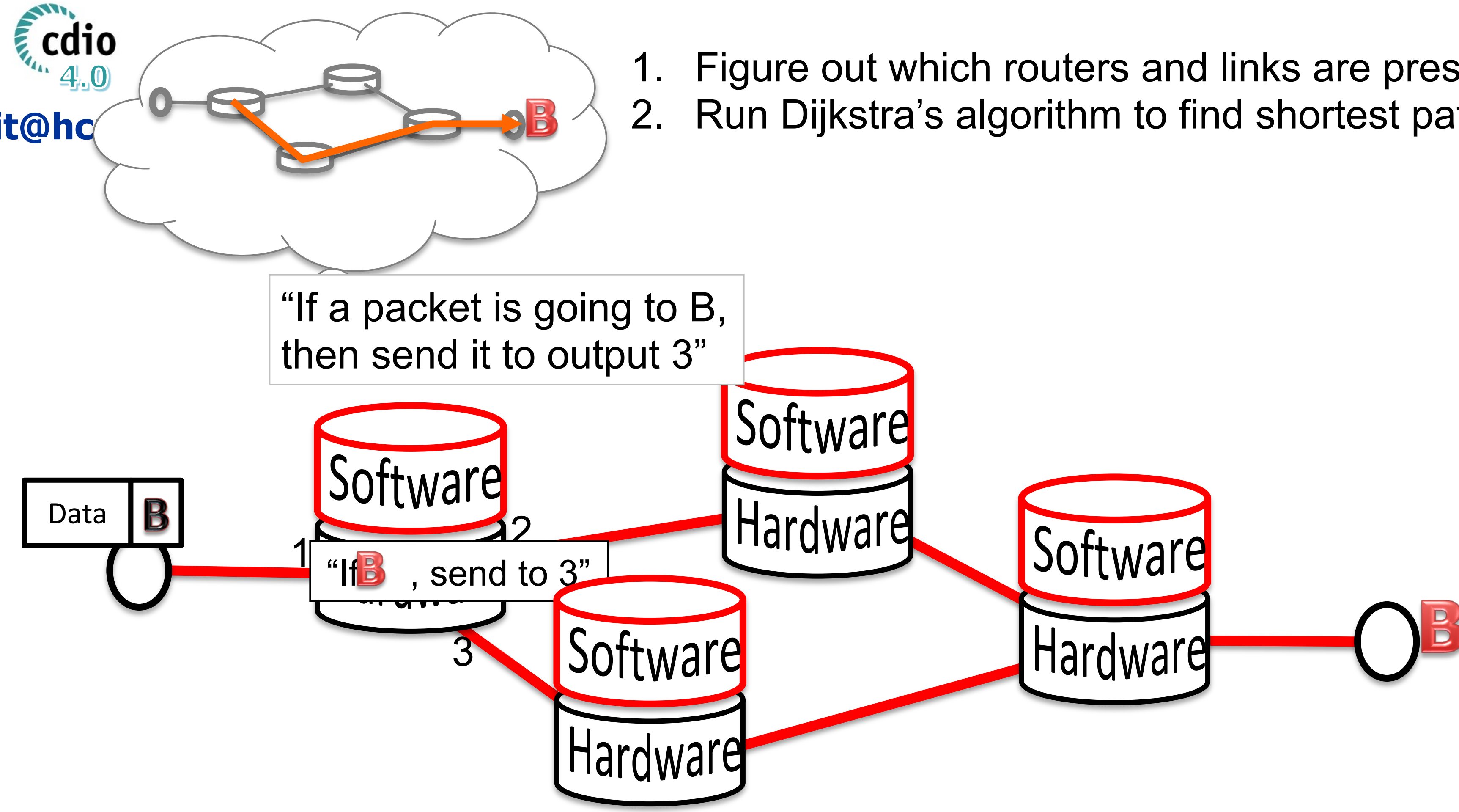
Control Plane

- Compute paths the packets will follow
  - Populate forwarding tables
  - Traditionally, a distributed protocol

- Example: Link-state routing (OSPF, IS-IS)
  - Flood the entire topology to all nodes
  - Each node computes shortest paths
  - Dijkstra's algorithm

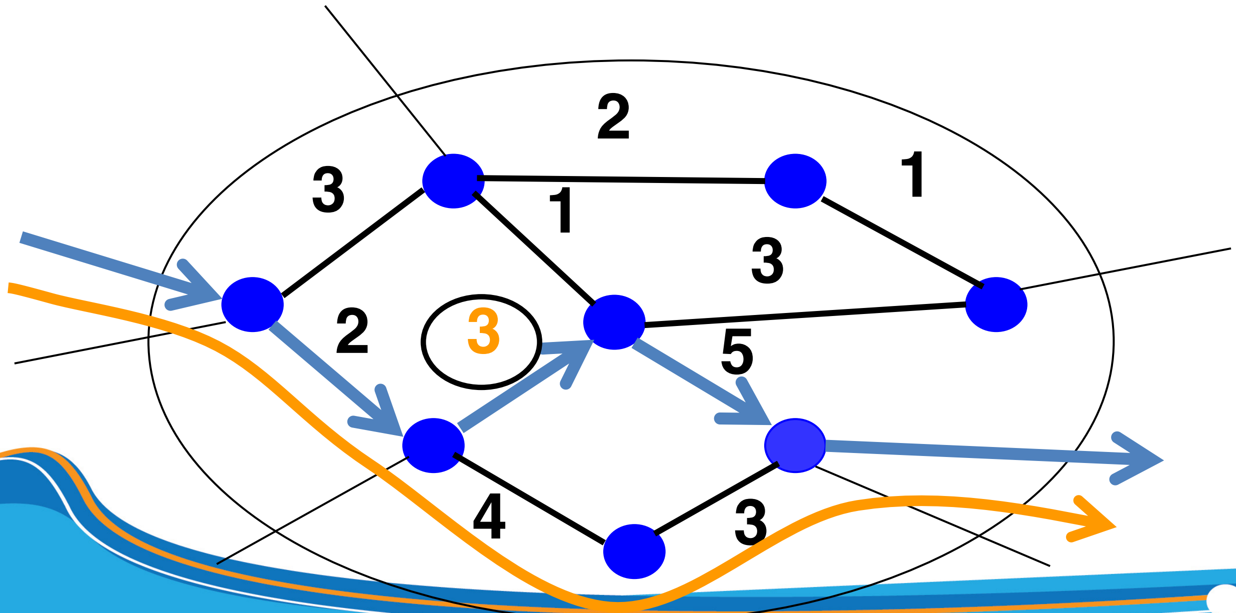

Software
Hardware
Software
Hardware
Software
Hardware
Software
Hardware

1. Figure out which routers and links are present.
2. Run Dijkstra's algorithm to find shortest paths.



Management Plane

- Traffic Engineering: setting the weights
  - Inversely proportional to link capacity?
  - Proportional to propagation delay?
  - Network-wide optimization based on traffic?



fit@hcmus

Challenges

(Too) many task-specific control mechanisms
- No modularity, limited functionality

Indirect
- Mu ou want
- Ex.

Uncoord
- Ca

The network is
- Hard to reason about
- Hard to evolve
- Expensive

Interacting protocols and mechanisms
- Routing, addressing, access control, QoS

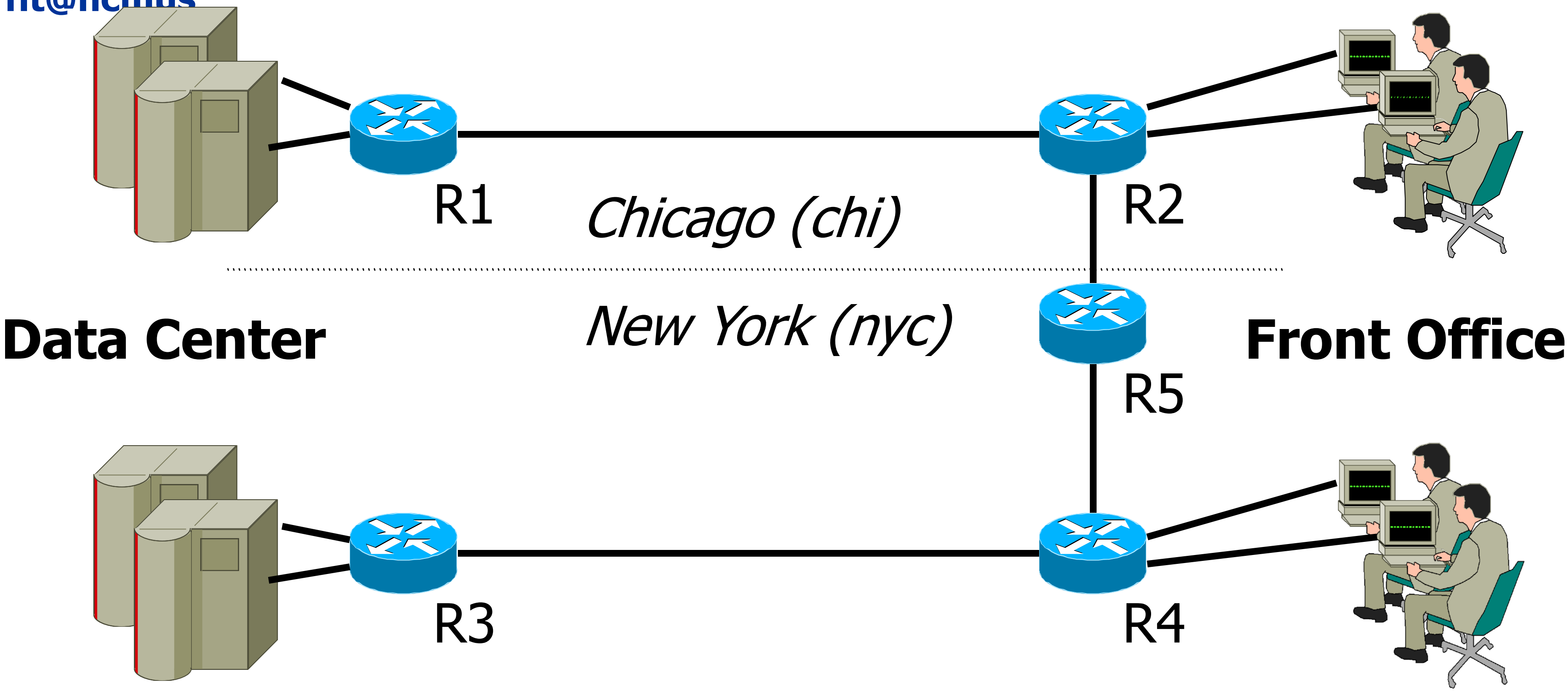
Example 1: Inter-domain Routing

- Today's inter-domain routing protocol, BGP, artificially constrains routes
  - Routing only on **destination IP address blocks**
  - Can only influence **immediate neighbors**
  - Very difficult to incorporate other information

- Application-specific peering
  - Route video traffic one way, and non-video another
- Blocking denial-of-service traffic
  - Dropping unwanted traffic further upstream
- Inbound traffic engineering
  - Splitting incoming traffic over multiple peering links

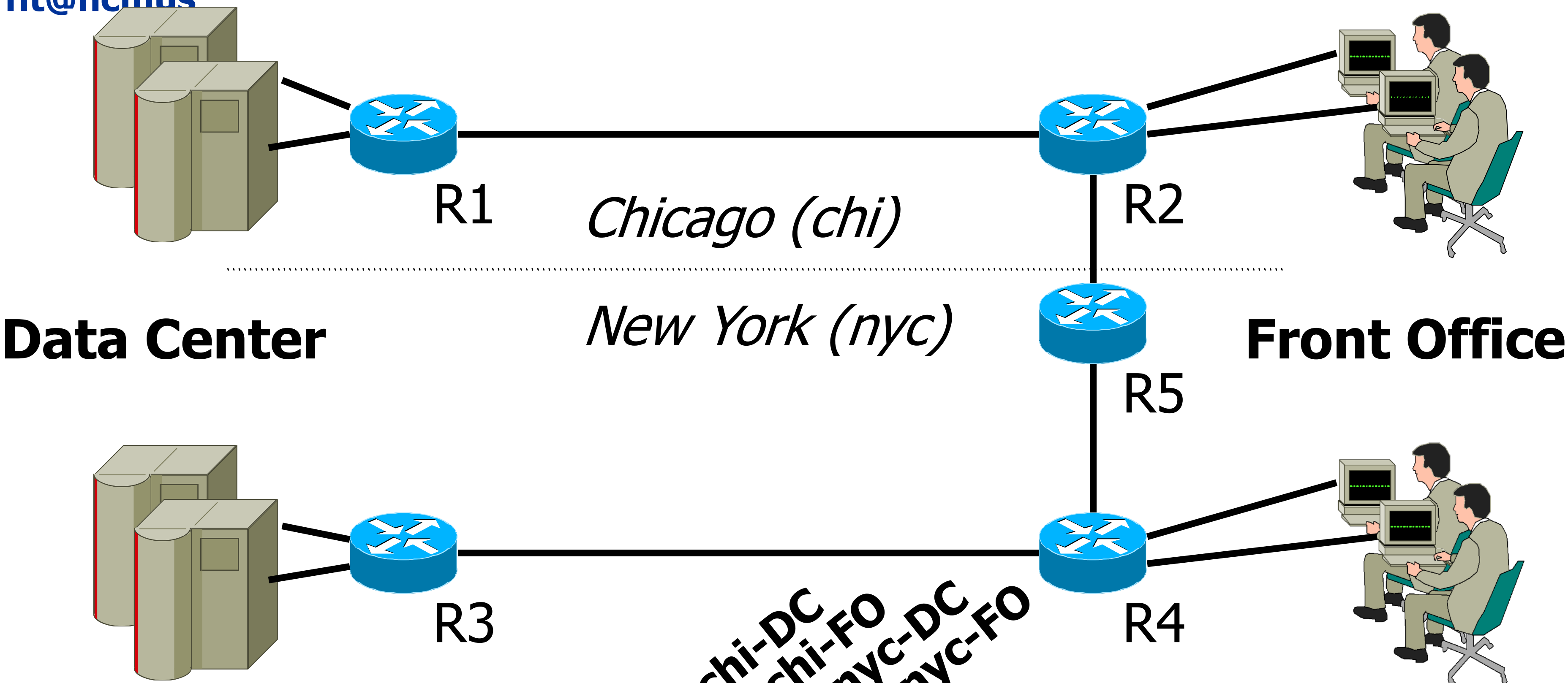
cdio 4.0
fit@hcmus

Example 2: Access Control



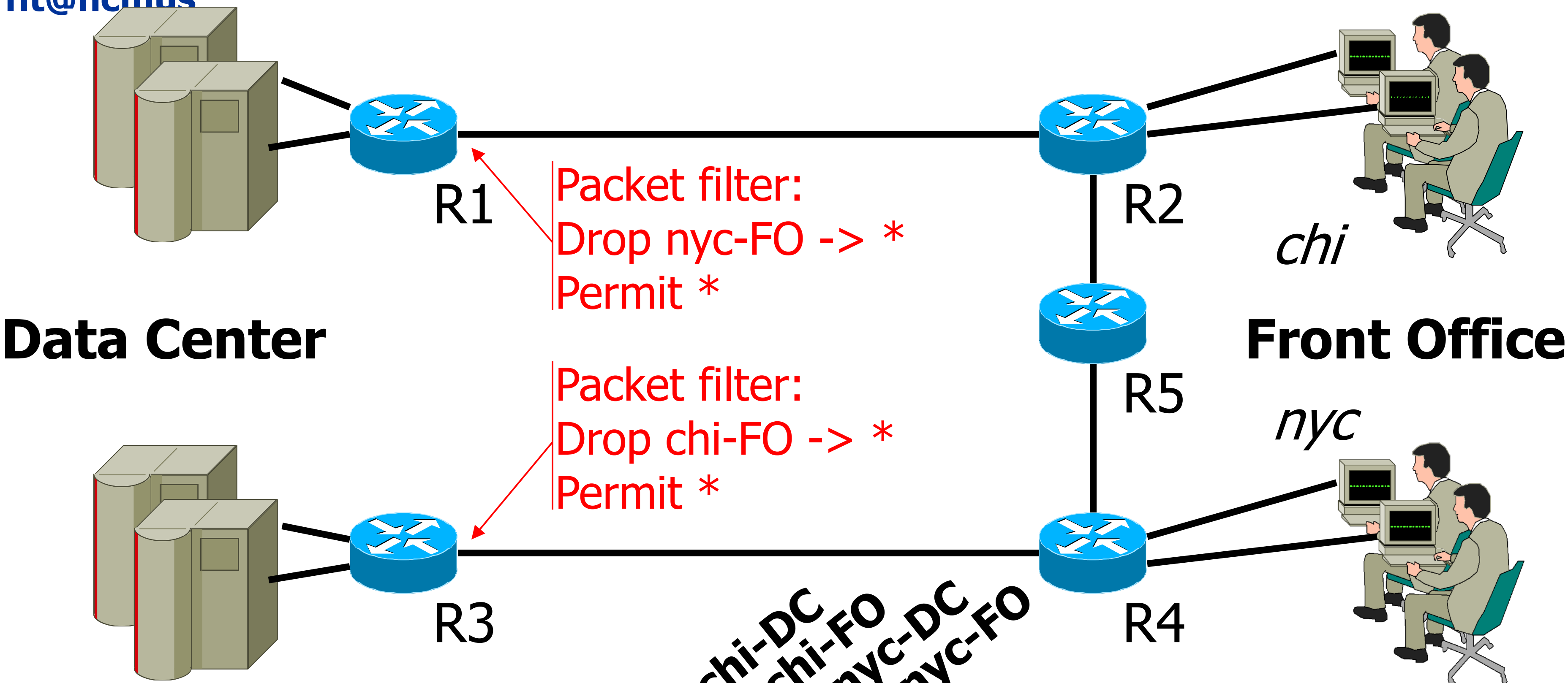
- Two locations, each with data center & front office
- All routers exchange routes over all links

Example 2: Access Control



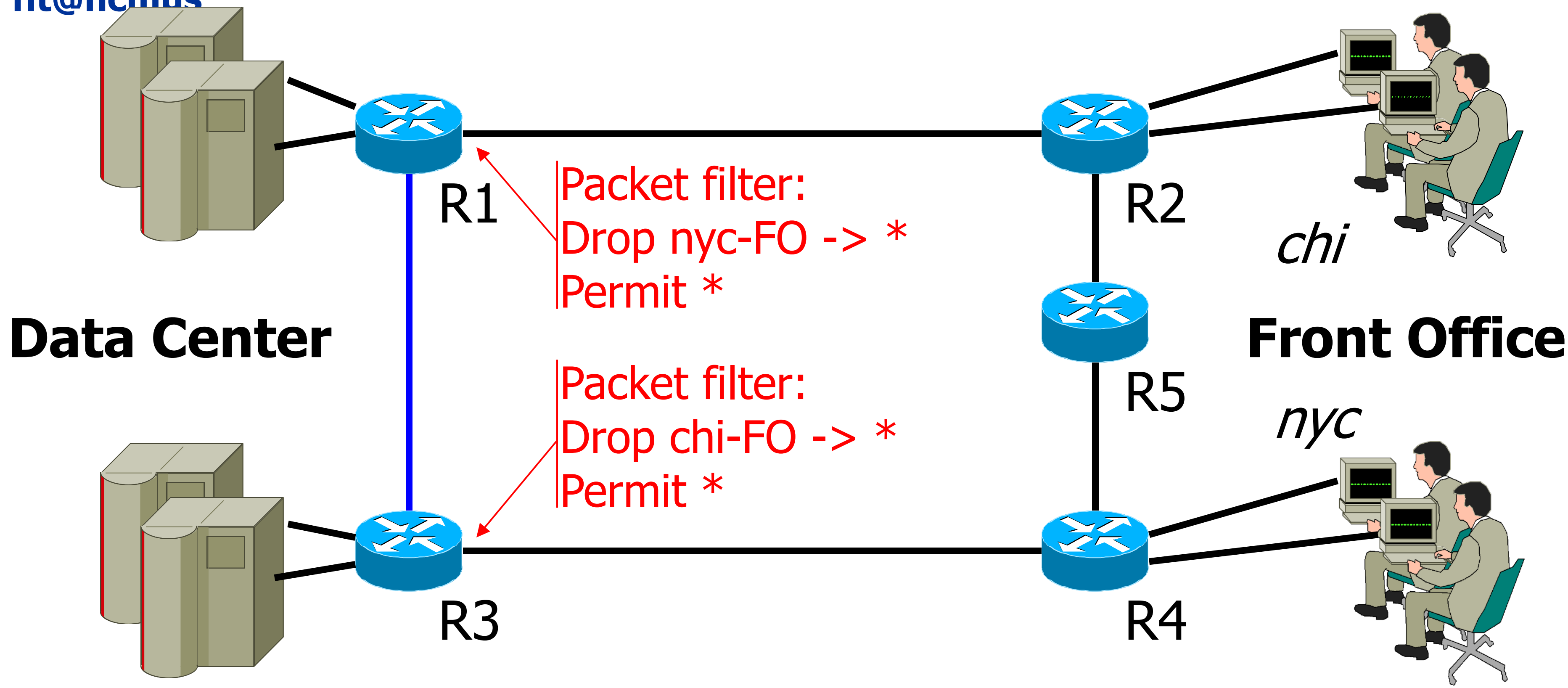
	chi-DC	chi-FO	nyc-DC	nyc-FO
chi-DC		●	●	⊘
chi-FO	●		⊘	●
nyc-DC	●	⊘		●
nyc-FO	⊘	●	●	

Example 2: Access Control



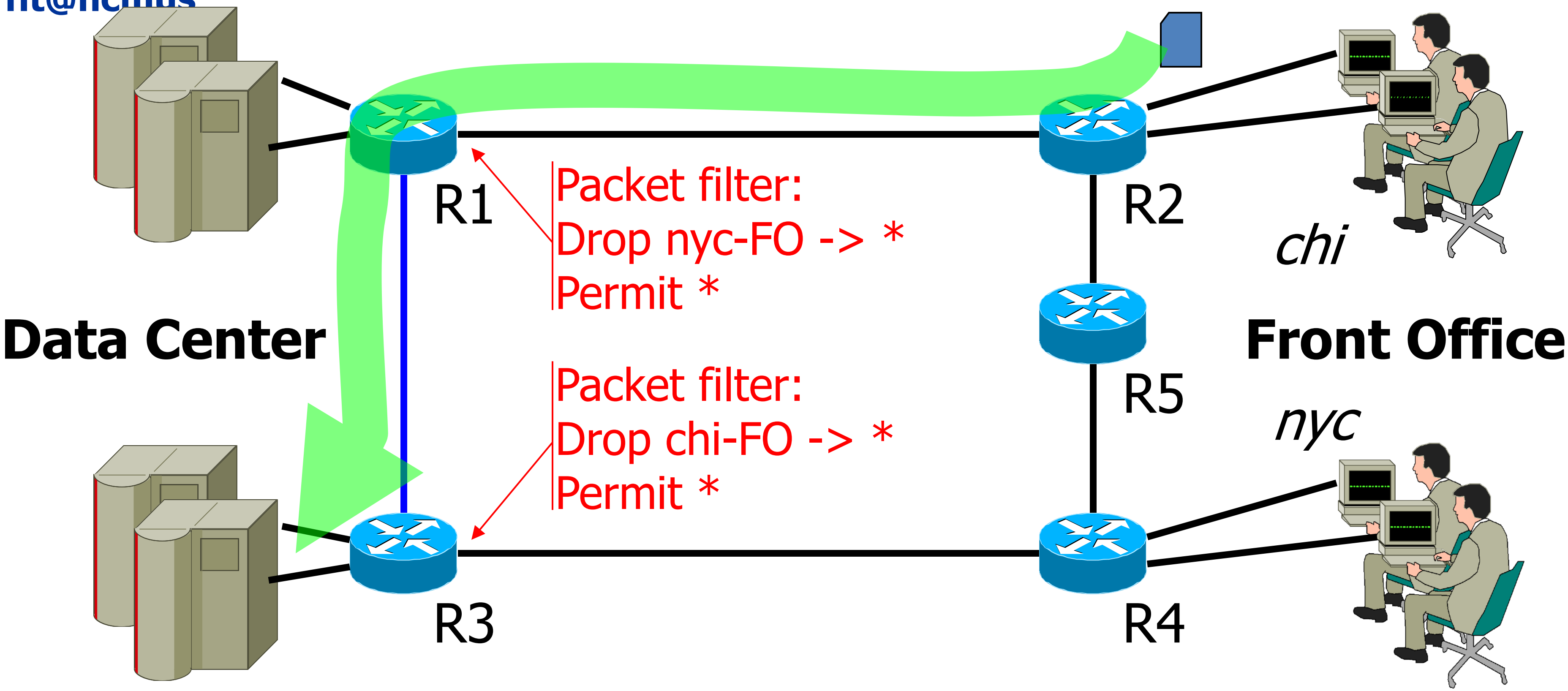
	chi-DC	chi-FO	nyc-DC	nyc-FO
chi-DC		●	●	⊘
chi-FO	●		⊘	●
nyc-DC	●	⊘		●
nyc-FO	⊘	●	●	

Example 2: Access Control



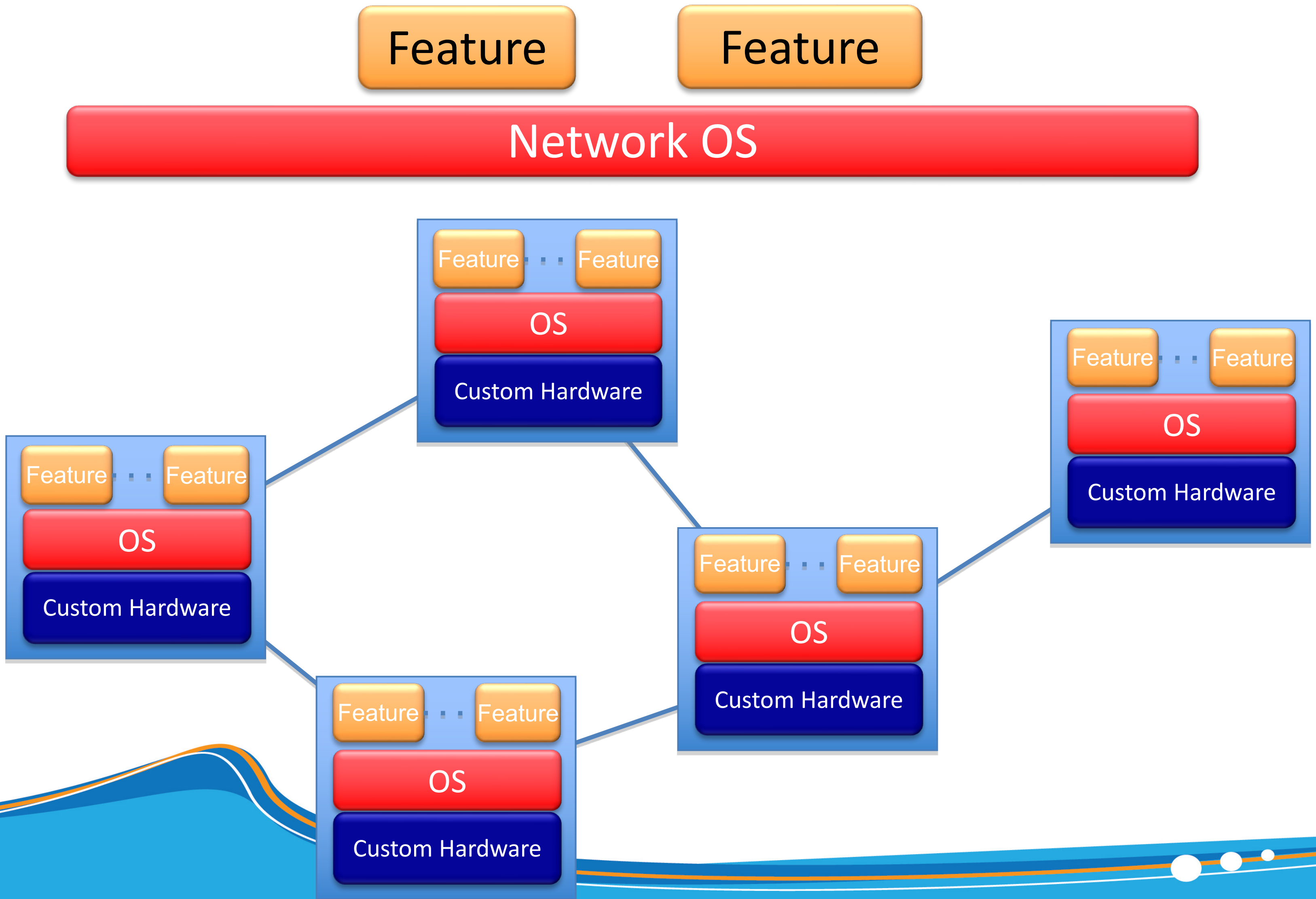
- A new short-cut link added between data centers
- Intended for backup traffic between centers

Example 2: Access Control



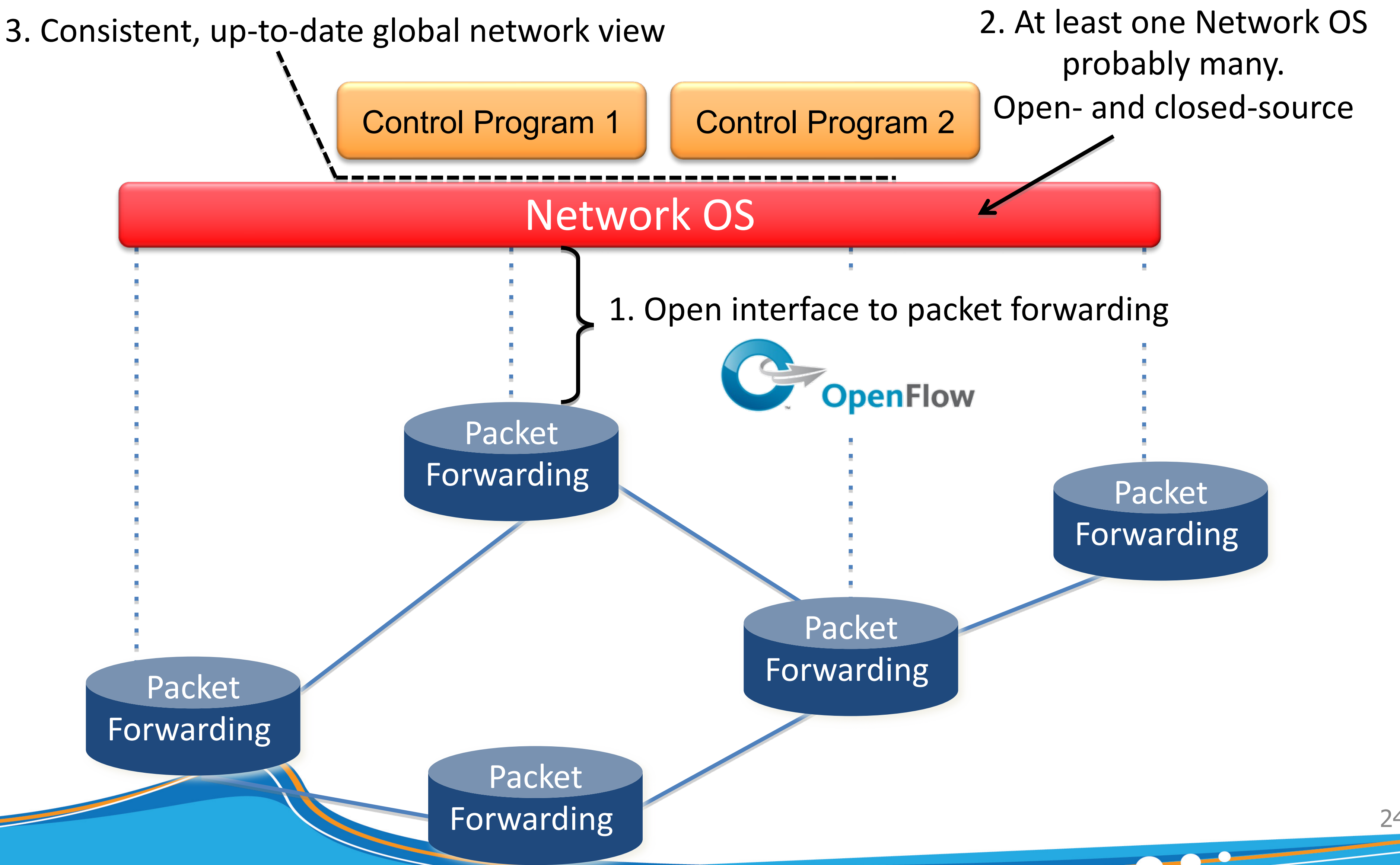
- Oops – new link lets packets violate ***access control policy***!
- Routing changed, but
- Packet filters don't update automatically

How SDN Changes the Network



fit@hcmus

Software Defined Network (SDN)



Network OS

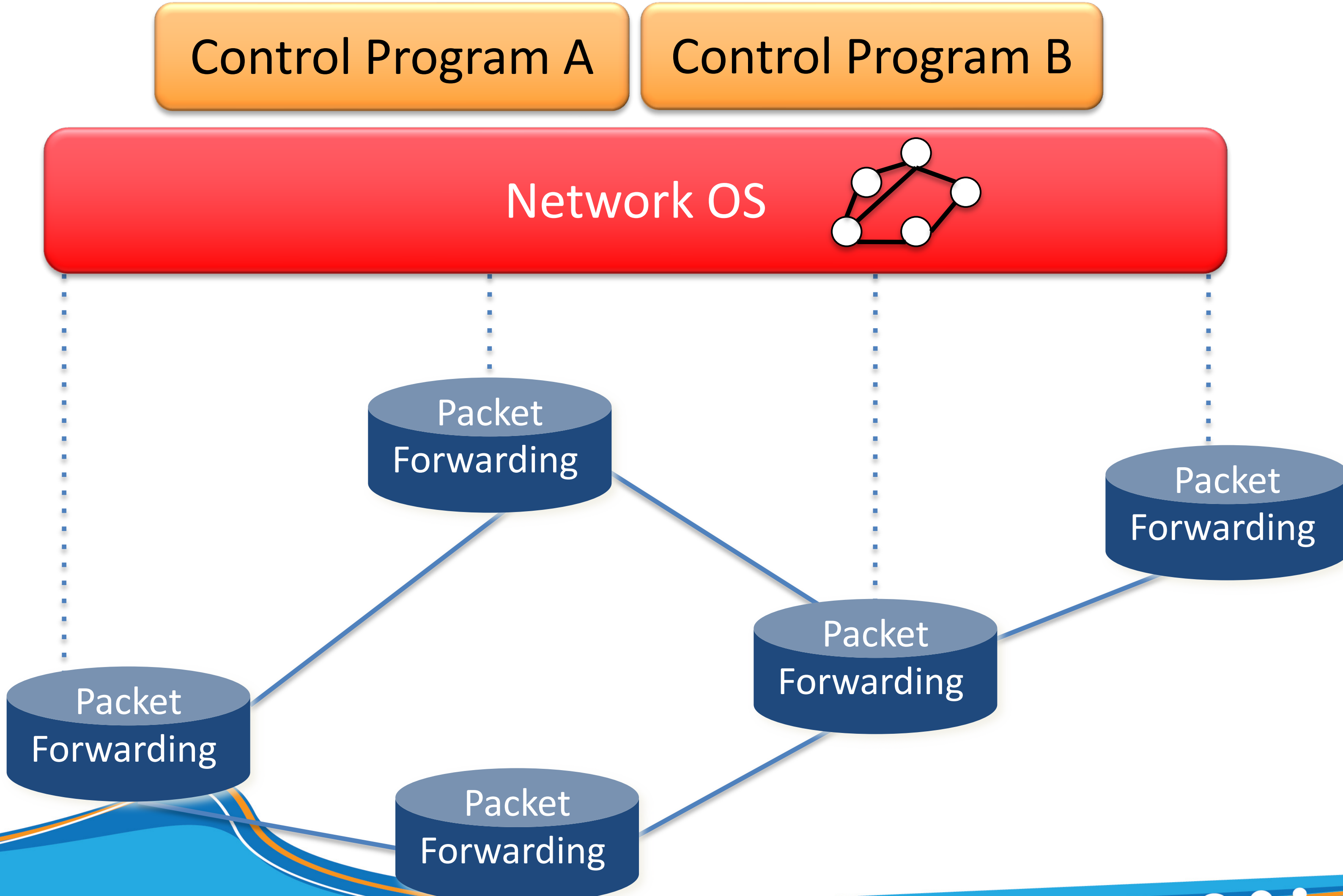
Network OS: distributed system that creates a consistent, up-to-date network view

- Runs on servers (controllers) in the network
- NOX, ONIX, Floodlight, Trema, OpenDaylight, HyperFlow, Kandoo, Beehive, Beacon, Maestro, … + more

Uses **forwarding abstraction** to:

- Get state information **from** forwarding elements
- Give control directives **to** forwarding elements

Software Defined Network (SDN)



Control Program

Control program operates on view of network

- **Input**: global network view (graph/database)
- **Output**: configuration of each network device

Control program is not a distributed system

- Abstraction hides details of distributed state

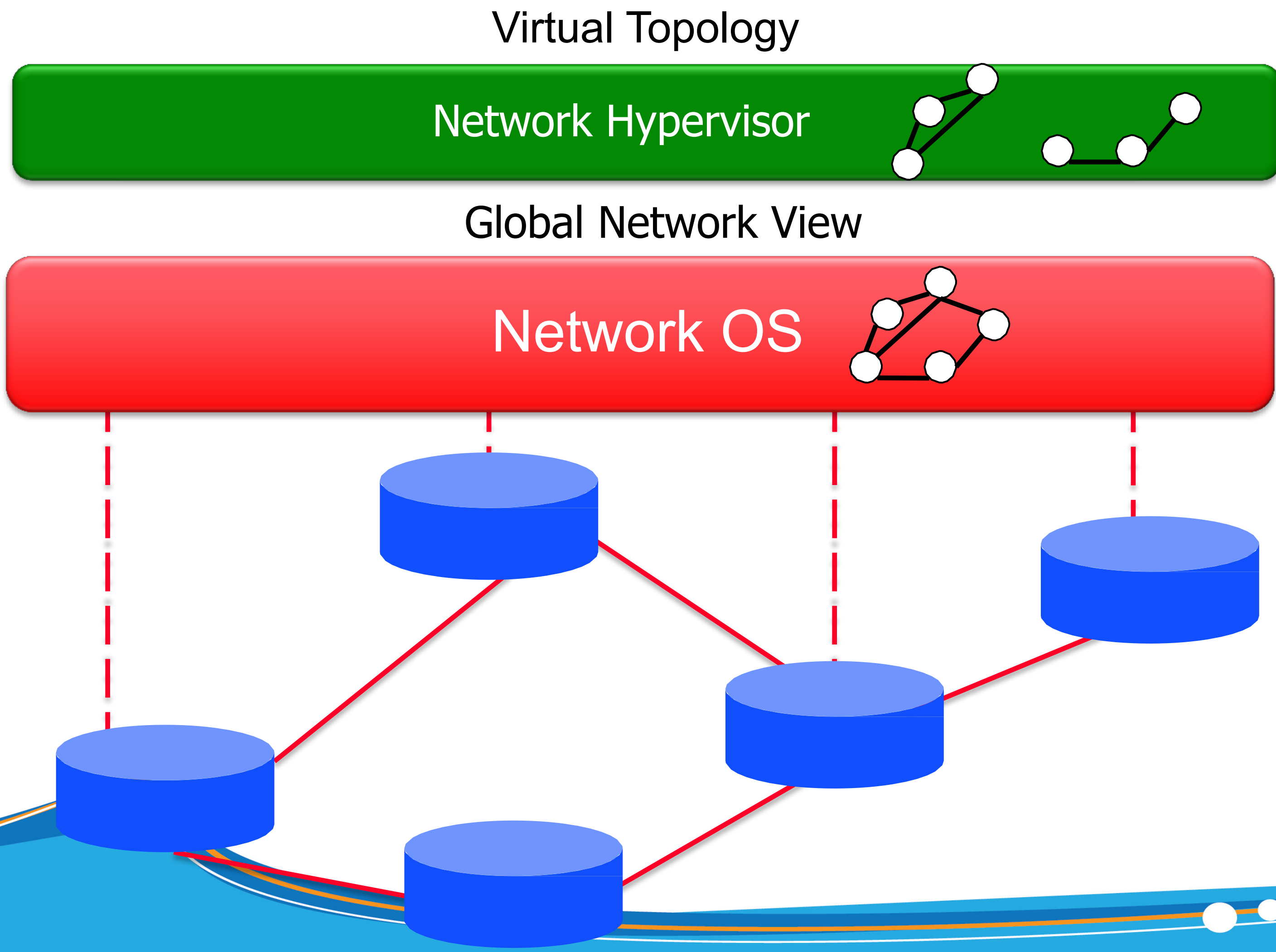
Forwarding Abstraction

Purpose: Standard way of defining forwarding state

- Flexible
  - Behavior specified by control plane
  - Built from basic set of forwarding primitives
- Minimal
  - Streamlined for speed and low-power
  - Control program not vendor-specific

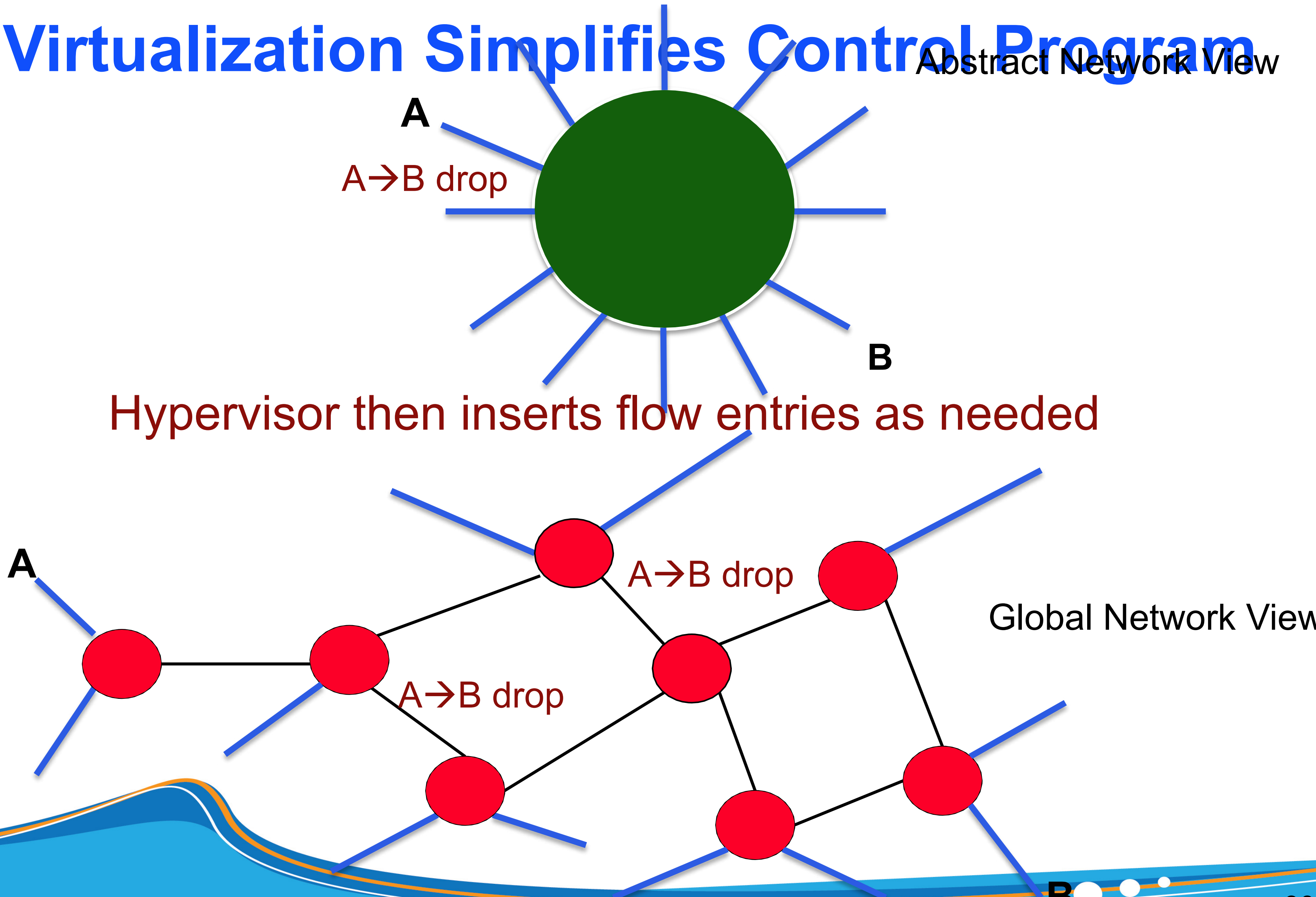
- OpenFlow is an example of such an abstraction

Software Defined Network



fit@hcmus

Virtualization Simplifies Control Program



Does SDN Simplify the Network?

What You Said

- "However, I remain skeptical that such an approach will actually simplify much in the long run. That is, the basic paradigm in networks (layers) is in fact a simple model. However, the ever-changing performance and functionality goals have forced more complexity into network design. I'm not sure if SDN will be able to maintain its simplified model as goals continue to evolve."

Does SDN Simplify the Network?

Abstraction doesn't eliminate complexity

- NOS, Hypervisor are still complicated pieces of code

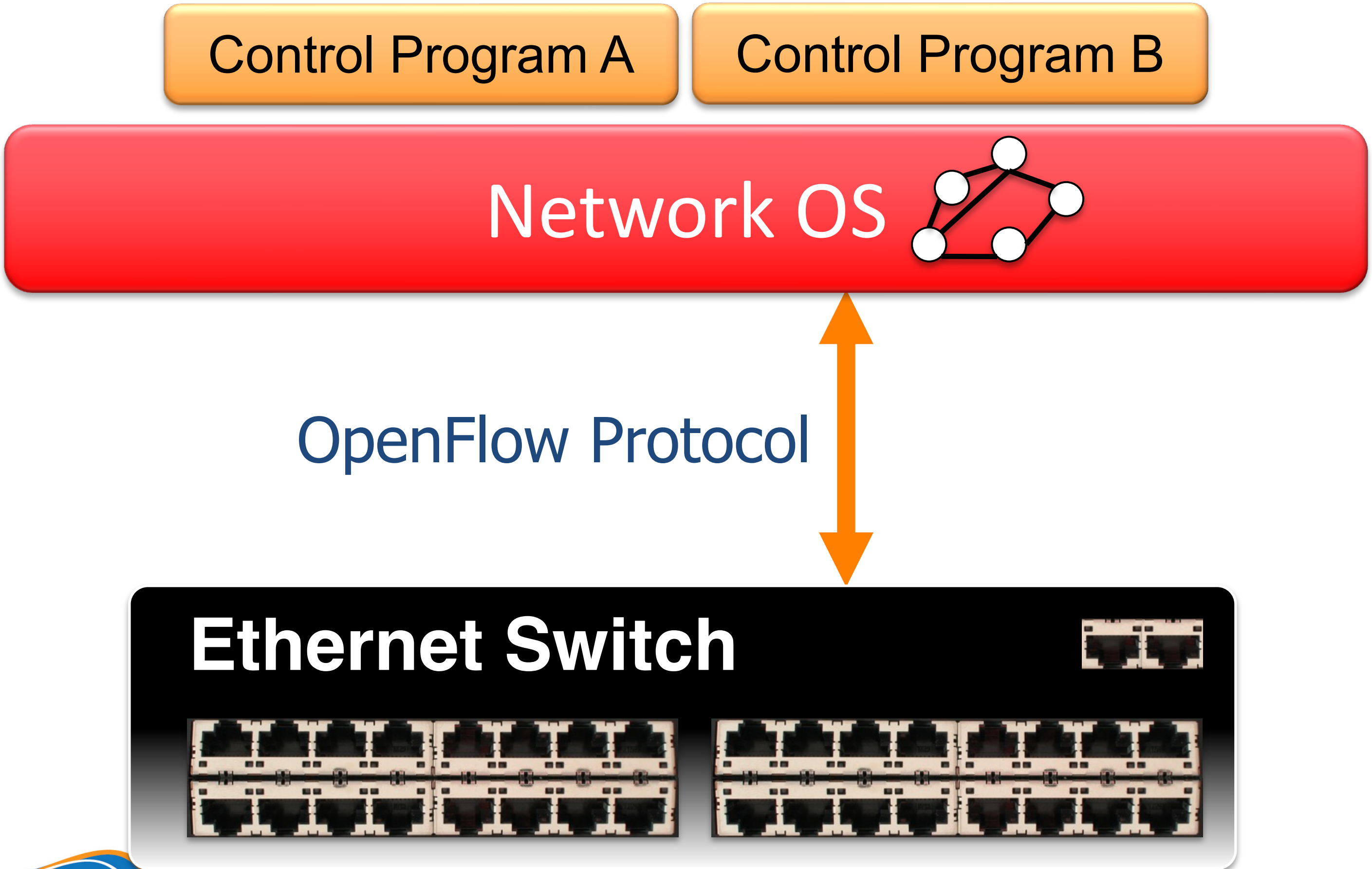
SDN main achievements

- Simplifies interface for control program (user-specific)
- Pushes complexity into reusable code (SDN platform)

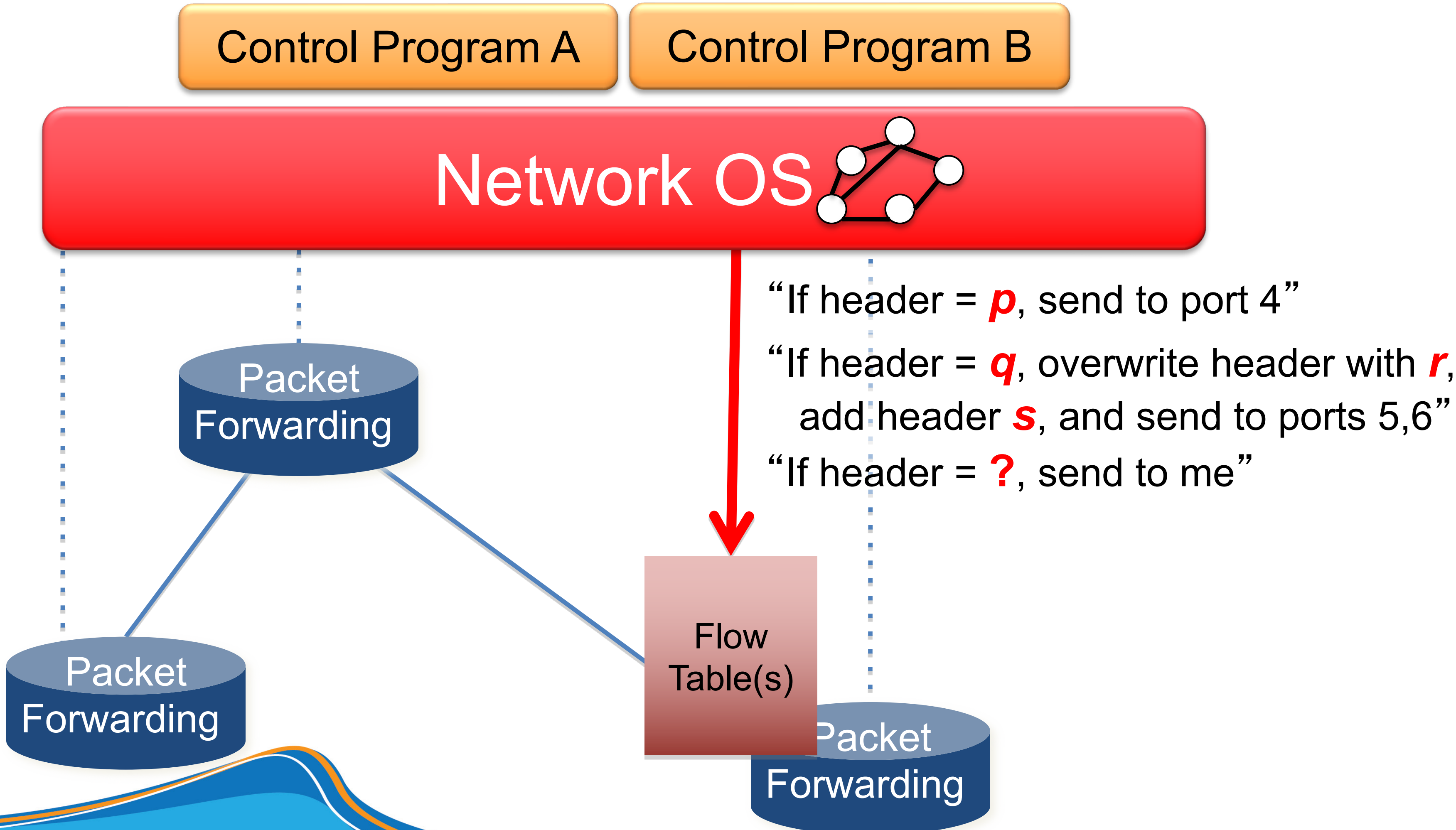
Just like compilers….

OpenFlow Basics

OpenFlow Basics



OpenFlow Basics



Primitives <Match, Action>

Match arbitrary bits in headers:

Header	Data

Match: 1000x01xx0101001x

- Match on any header, or new header
- Allows any flow granularity

Action

- Forward to port(s), drop, send to controller
- Overwrite header with mask, push or pop
- Forward at specific bit-rate

fit@hcmus

OpenFlow Rules

Exploit the flow table in switches, routers, and chipsets

Flow 1.	Rule (exact & wildcard)	Action	Statistics
Flow 2.	Rule (exact & wildcard)	Action	Statistics
Flow 3.	Rule (exact & wildcard)	Action	Statistics
		....	
Flow N.	Rule (exact & wildcard)	Default Action	Statistics

Why is SDN happening now?

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

fit@hcmus

The Road to SDN

- Active Networking: 1990s
  - First attempt make networks programmable
  - Demultiplexing packets to software programs, network virtualization, …
- Control/Dataplane Separation: 2003-2007
  - ForCes [IETF],
    RCP, 4D [Princeton, CMU],
    SANE/Ethane [Stanford/Berkeley]
  - Open interfaces between data and control plane, logically centralized control
- OpenFlow API & Network Oses: 2008
  - OpenFlow switch interface [Stanford]
  - NOX Network OS [Nicira]

N. Feamster et al., "The Road to SDN: An Intellectual History of Programmable Networks", ACM SIGCOMM CCR 2014.

SDN Drivers

- Rise of merchant switching silicon
  - Democratized switching
  - Vendors eager to unseat incumbents
- Cloud / Data centers
  - Operators face real network management problems
  - Extremely cost conscious; desire a lot of control
- The right balance between vision & pragmatism
  - OpenFlow compatible with existing hardware
- A “killer app”: Network virtualization

Virtualization is Killer App for SDN

Consider a multi-tenant datacenter

- Want to allow each tenant to specify virtual topology
- This defines their individual policies and requirements

Datacenter's network hypervisor compiles these virtual topologies into set of switch configurations

- Takes 1000s of individual tenant virtual topologies
- Computes configurations to implement all simultaneously

This is what people are paying money for….

- ***Enabled by SDN's ability to virtualize the network***

Q&A