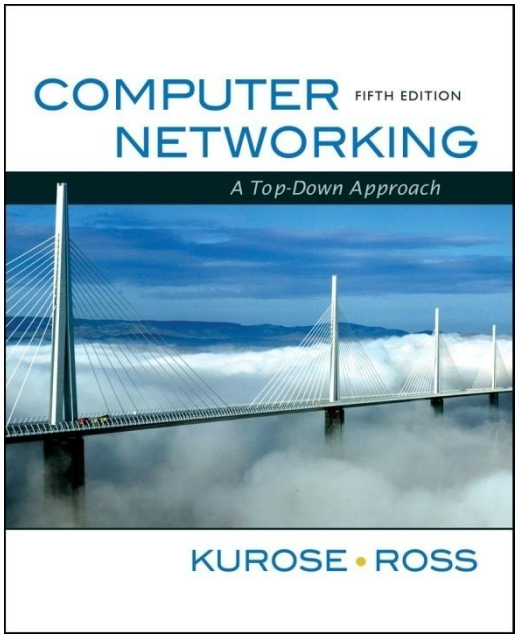
Camada de Redes

Prof Nelson Fonseca

Chapter 4
Network Layer



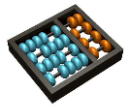
A note on the use of these ppt slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- ❑ If you use these slides (e.g., in a class) in substantially unaltered form, that you mention their source (after all, we'd like people to use our book!)
- ❑ If you post any slides in substantially unaltered form on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy! JFK/KWR

Computer Networking: A Top Down Approach 5th edition.
Jim Kurose, Keith Ross
Addison-Wesley, April 2009.



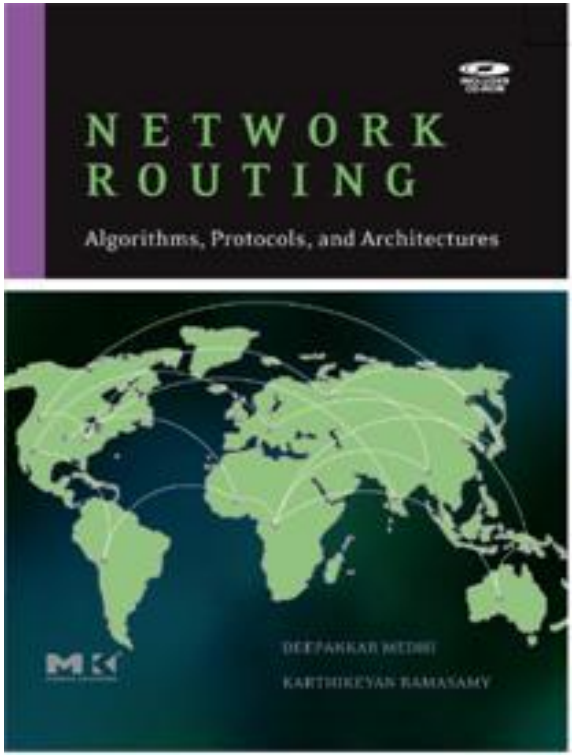
Outra fonte bibliográfica

- Alguns slides nesse arquivo foram gentilmente cedidos pelos autores do livro:
- Computer Networks: An Open Source Approach, Ying-Dar Lin, Ren-Hung Hwang, Fred Baker, published by McGraw Hill, Feb 2011

UNICAMP

Outra fonte bibliográfica

- Alguns slides nesse arquivo foram gentilmente cedidos pelos autores do livro:
- D. Medhi and K. Ramasamy, Network Routing: Algorithms, Protocols and Architectures, Morgan Kaufmann Publishers



Camada de Redes

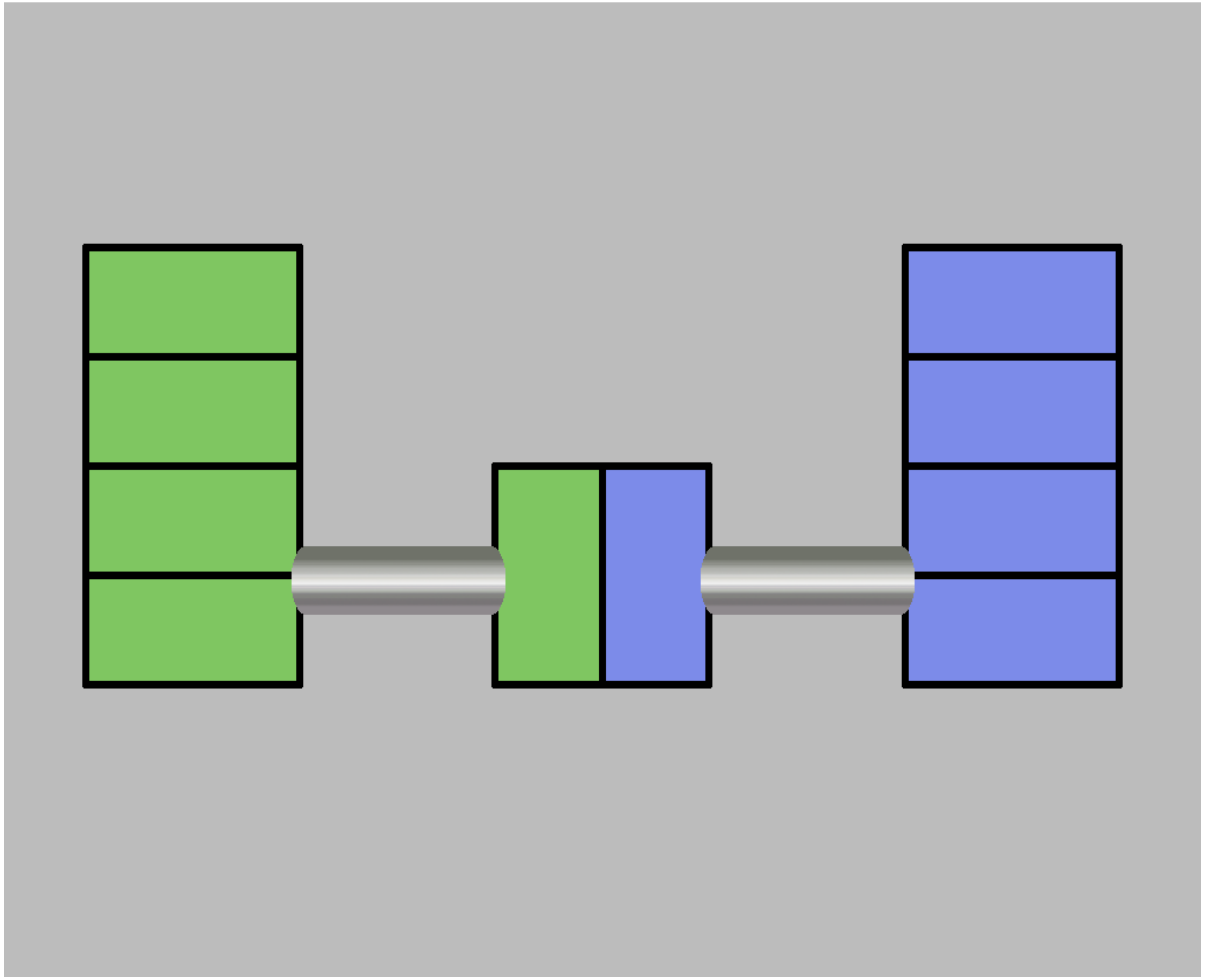
Objetivos do Capítulo:

- Entender os principais princípios do serviço da camada de redes :
  - ✓ Modelos de serviço da camada de redes
  - ✓ Encaminhamento e roteamento
  - ✓ Como um roteador funciona
  - ✓ Roteamento (seleção de caminhos)
  - ✓ Lidando com escala
  - ✓ IPv
- Implementações na Internet

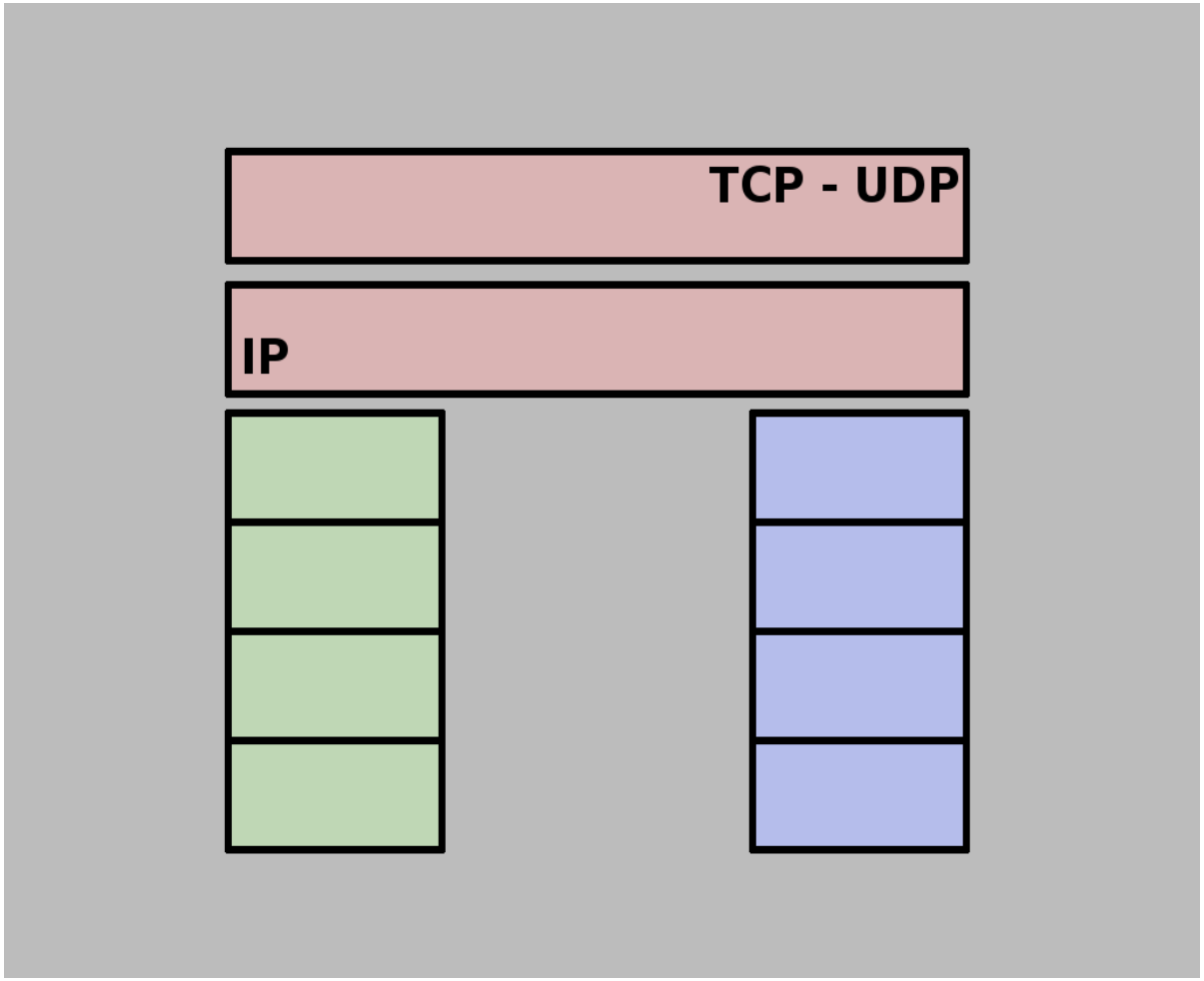
Roteiro

- 4. 1 Introdução
- 4.2 Circuitos virtuais x datagrama
- 4.3 Como é um roteador
- 4.4 Protocolo IP
  - ✓ Formato datagarma
  - ✓ endereçamento IPv4
  - ✓ ICMP
  - ✓ IPv6
- 4.5 Algoritmos de roteamento
  - ✓ Estado de enlace
  - ✓ Vetor distância
  - ✓ Roteamento hierarquico
- 4.6 Roteamento na Internet
  - ✓ RIP
  - ✓ OSPF
  - ✓ BGP
- 4.7 Roteamento Broadcast e multicast

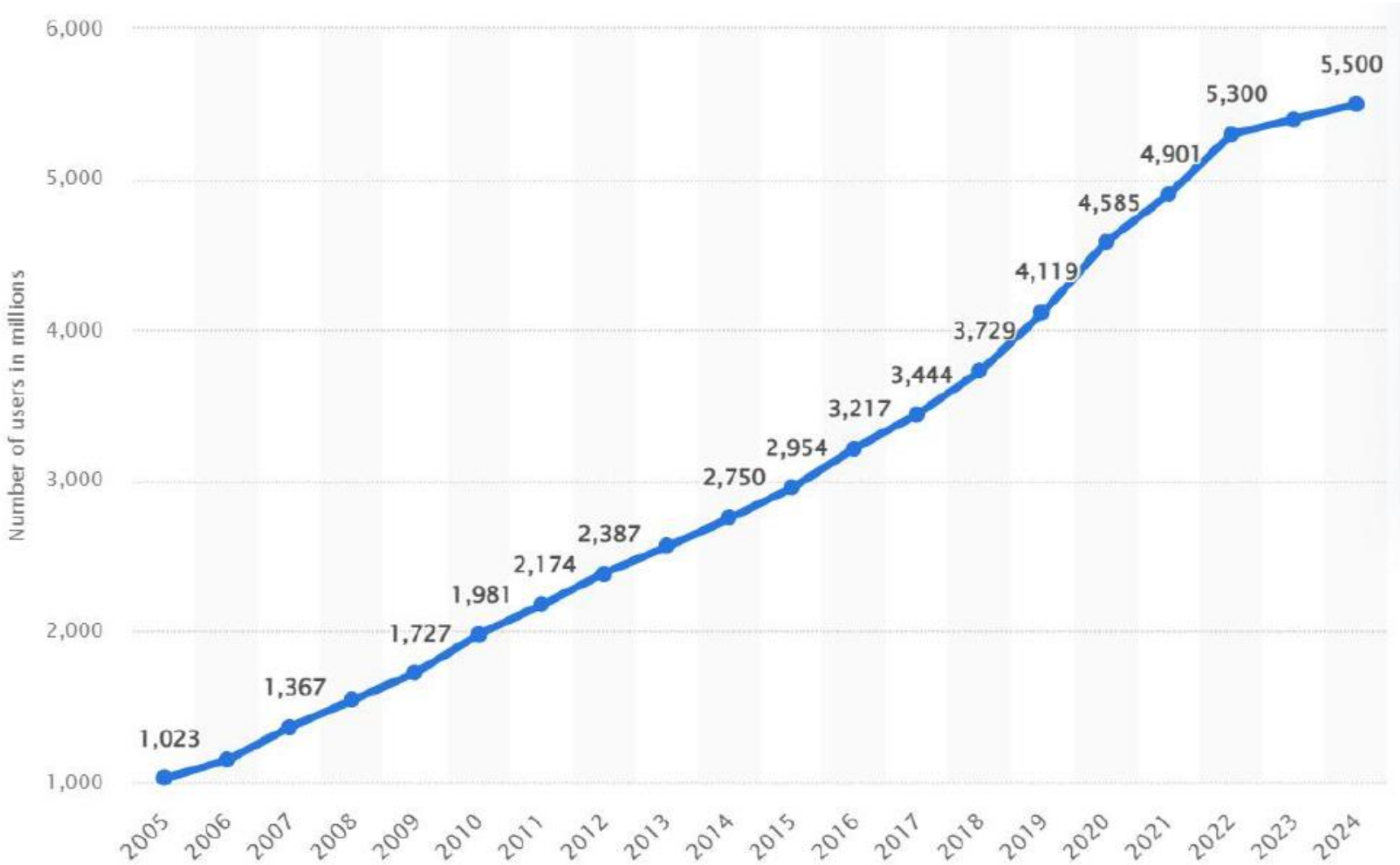
Interconexão de Redes



A Internet



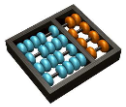
A Internet


Number of users in millions
6,000
5,000
4,000
3,000
2,000
1,000
1,023
1,367
1,727
1,981
2,174
2,387
2,750
2,954
3,217
3,444
3,729
4,119
4,585
4,901
5,300
5,500
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024

Países com maior número de Usuários da Internet

	COUNTRY	NUMBER OF INTERNET USERS 2024 ↓	DATA YEAR	% OF POPULATION USING INTERNET	INTERNET TRAFFIC 2020
	China	1.1B	2022	77.3%	989.1M
	India	881.3M	2023	62.6%	749.3M
	United States	311.3M	2023	92.4%	312.3M
	Indonesia	215.6M	2023	78.8%	196.4M
	Pakistan	170M	2022	70.8%	76.4M
	Brazil	165.3M	2022	77.1%	149.1M
	Nigeria	136.2M	2020	63.8%	154.3M
	Russia	129.8M	2022	89.5%	116.4M

UNICAMP



Estatística Usuários Internet Brasil

Uso de internet

- Usuários de internet: 187.9 milhões
- Penetração da internet: 86.6%
- Crescimento anual de usuários de internet: +3.3% (6.1 milhões)

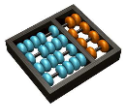
Conexões móveis

- Conexões móveis: 210.3 milhões
- Penetração de conexões móveis: 96.9%

Velocidades de conexão à internet

- Velocidade média de internet móvel: 47.09 Mbps
- Velocidade média de internet fixa: 140.46 Mbps

UNICAMP



Architectural Principles of the Internet

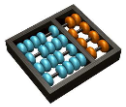
RFC 1958

"Many members of the Internet community would argue that there is no architecture, but only a tradition, which was not written down for the first 25 years (or at least not by the IAB). However, in very general terms, the community believes that the goal is connectivity, the tool is the Internet Protocol, and the intelligence is end to end rather than hidden in the network."

Three cornerstone beliefs:

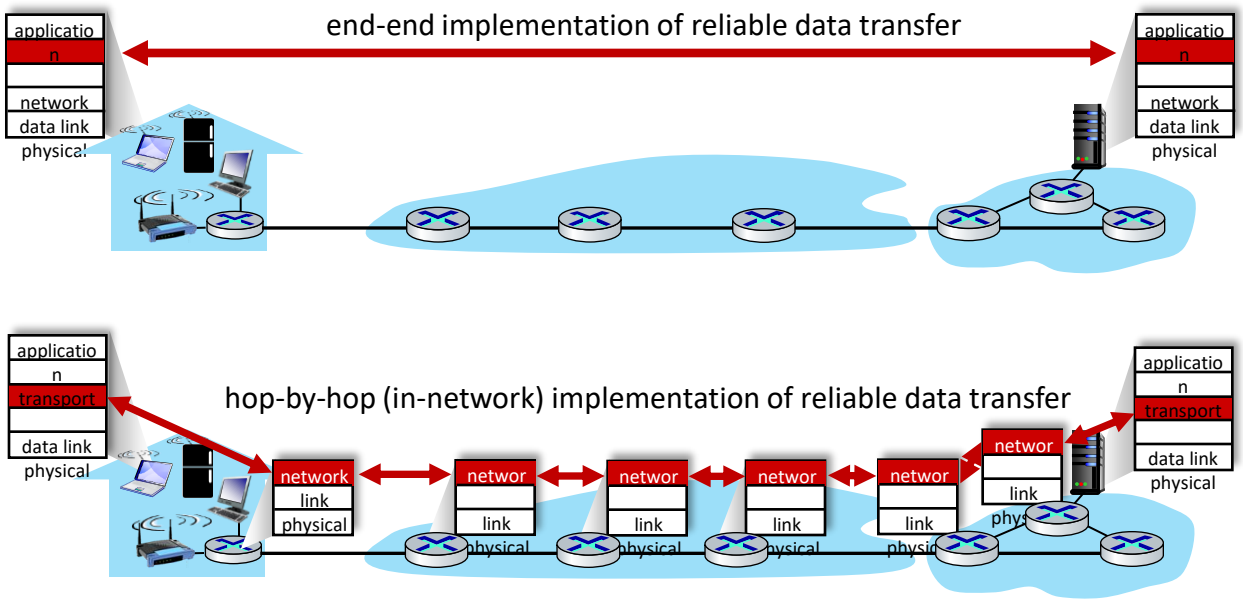
- simple connectivity
- IP protocol: that narrow waist
- intelligence, complexity at network edge

UNICAMP



The end-end argument

- some network functionality (e.g., reliable data transfer, congestion) can be implemented in network, or at network edge



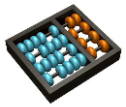
UNICAMP

The end-end argument

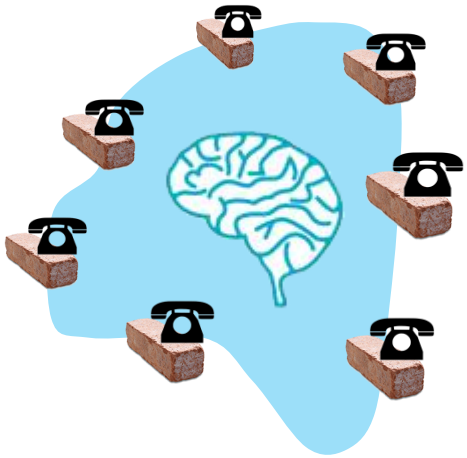
- some network functionality (e.g., reliable data transfer, congestion) can be implemented in network, or at network edge

> "The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the end points of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible. (Sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.)
>
> We call this line of reasoning against low-level function implementation the "end-to-end argument."
>
> Saltzer, Reed, Clark 1981

UNICAMP

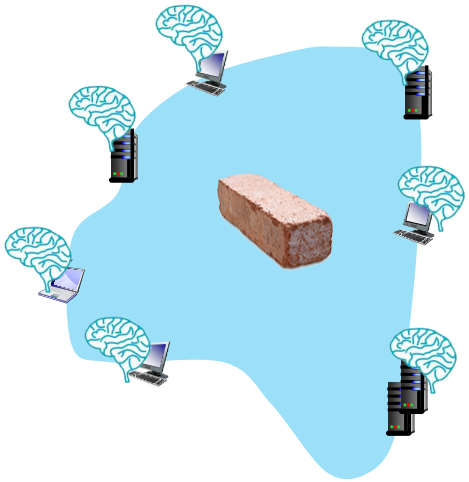


Where's the intelligence?



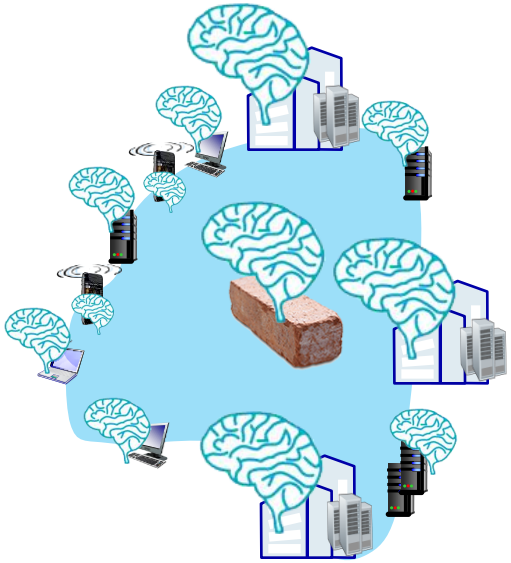
20th century phone net:

- intelligence/computing at network switches



Internet (pre-2005)

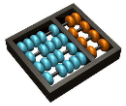
- intelligence, computing at edge



Internet (post-2005)

- programmable network devices
- intelligence, computing, massive application-level infrastructure at edge

UNICAMP

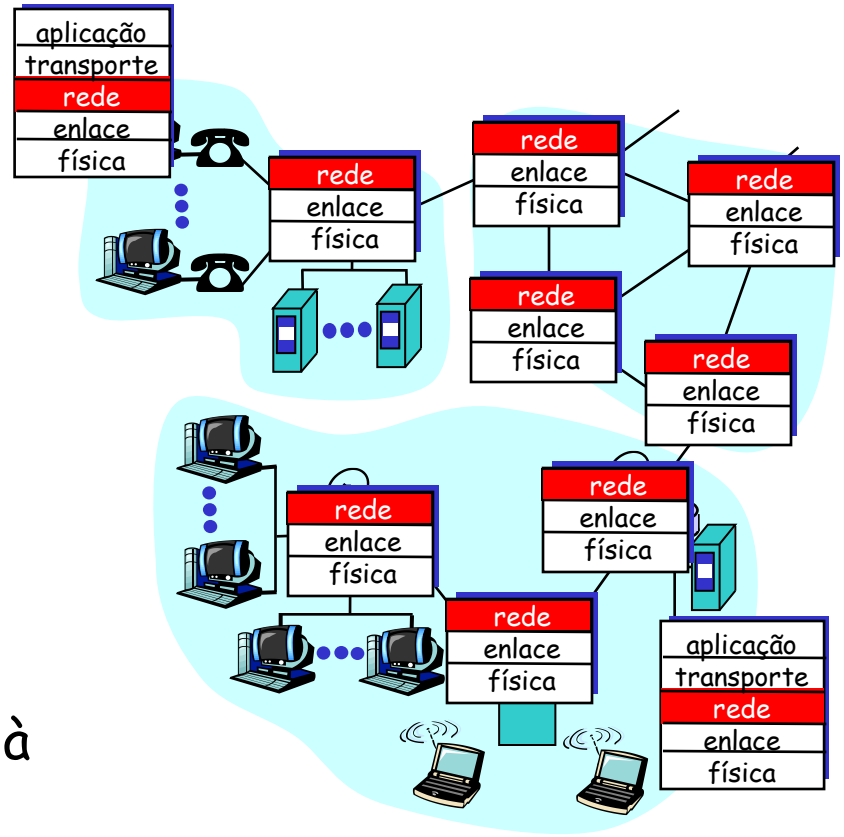


Funções da camada de rede

- transporta pacote da estação remetente à receptora
- protocolos da camada de rede em *cada* estação, roteador

Duas funções importantes:

- *determinação do caminho:* rota seguida por pacotes da origem ao destino. *Algoritmos de roteamento*
- *comutação:* mover pacotes dentro do roteador da entrada à saída apropriada



Two key network-layer functions

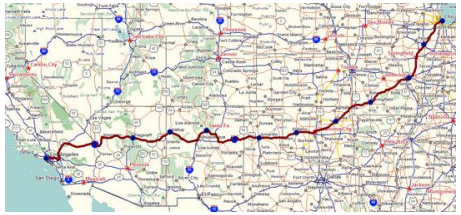
network-layer functions:

- *forwarding:* move packets from a router's input link to appropriate router output link
- *routing:* determine route taken by packets from source to destination
  - *routing algorithms*

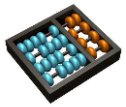
analogy: taking a trip

- *forwarding:* process of getting through single interchange
- *routing:* process of planning trip from source to destination

forwarding



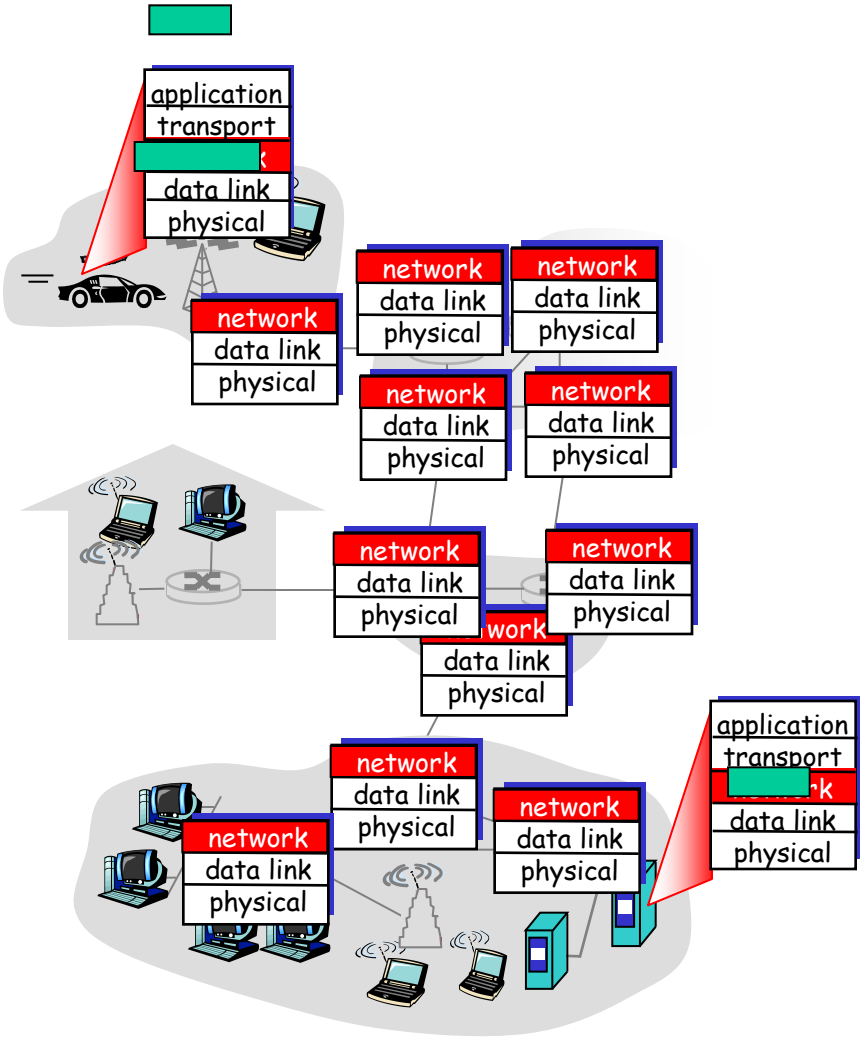
routing



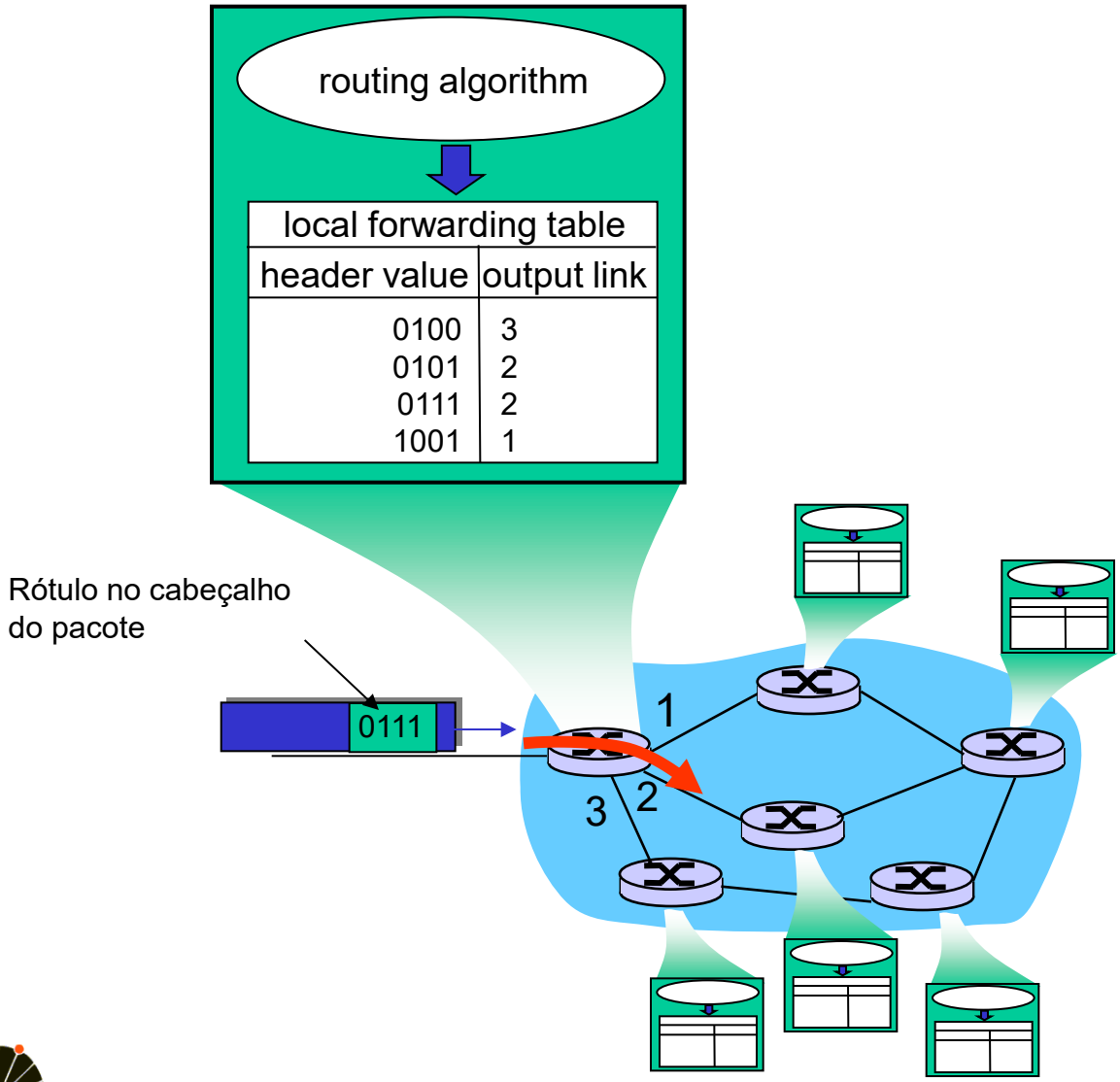
UNICAMP

Network layer

- Transporta segmentos do transmissor ao repector
- Encapsula segmentos em datagramas
- No receptor, desencapsula e entrega a camada de transporte
- roteadores examina cabeçalho de todos os datagrams que passam pelo roteador



Roteamento e encaminhamento



Modelo de serviço de rede

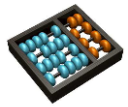
Q: Qual é o *modelo de serviço* para o "canal" que transporta pacotes do remetente ao receptor?

abstração do serviço

- largura de banda garantida?
- preservação de temporização entre pacotes (sem *jitter*)?
- entrega sem perdas?
- entrega ordenada?
- realimentar informação sobre congestionamento ao remetente?

A abstração mais importante provida pela camada de rede:

circuito virtual
ou
datagrama?



Rede de datagramas: o modelo da Internet

- não requer estabelecimento de chamada na camada de rede
- roteadores: não guardam estado sobre conexões fim a fim
  - ✓ não existe o conceito de "conexão" na camada de rede
- pacotes são roteados tipicamente usando endereços de destino
  - ✓ pacotes entre o mesmo par origem-destino podem seguir caminhos diferentes

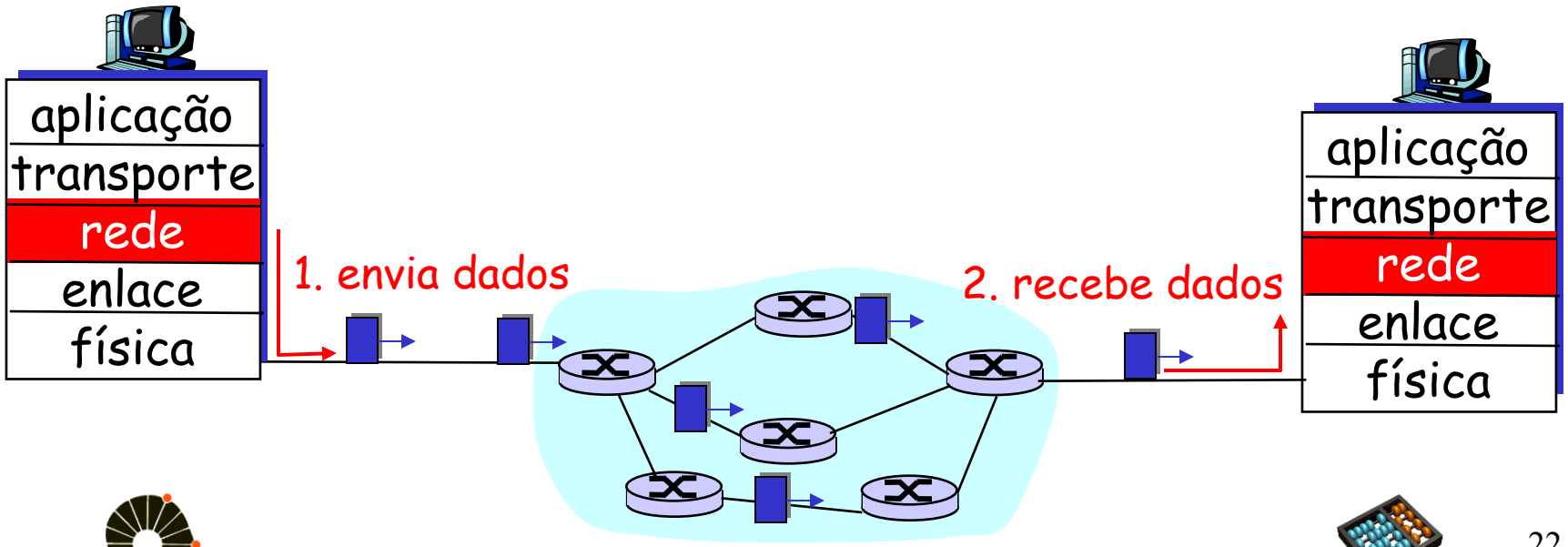


Tabela Encaminhamento

Faixa Endereço Destino	Interface Enlace
11001000 00010111 00010000 00000000 até 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 até 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 até 11001000 00010111 00011111 11111111	2
caso contrário	3

Casamento Prefixo mais longo (Longest prefix matching)

Prefix Match	Interface Enlace
11001000 00010111 00010	0
11001000 00010111 00011000	1
11001000 00010111 00011	2
caso contrario	3

Exemplos

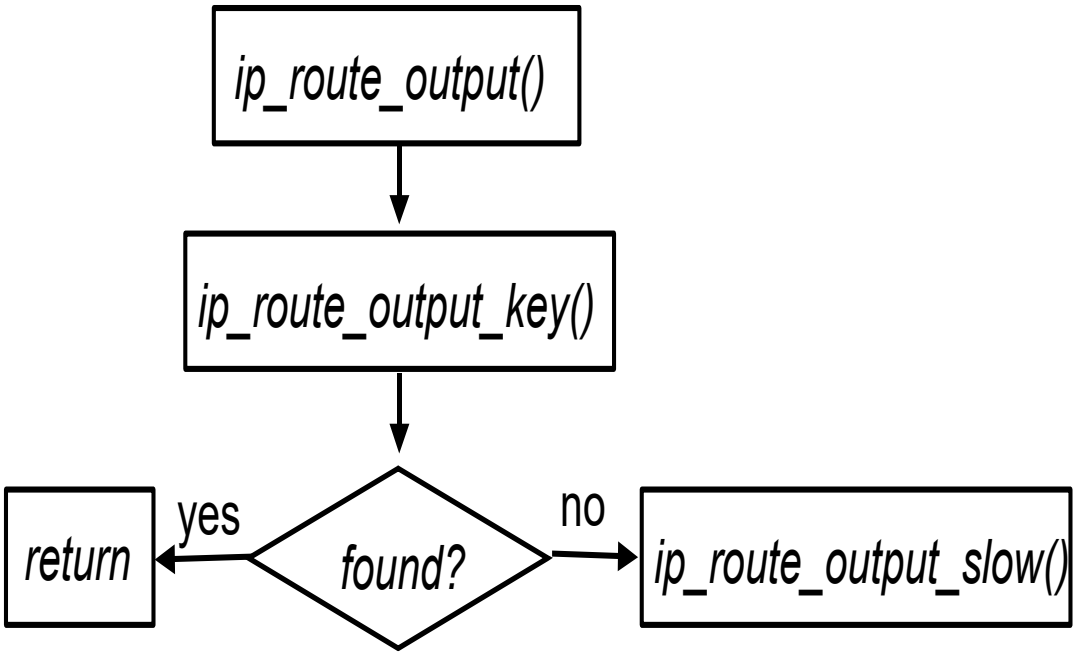
DA: 11001000 00010111 00010110 10100001

DA: 11001000 00010111 00011000 10101010

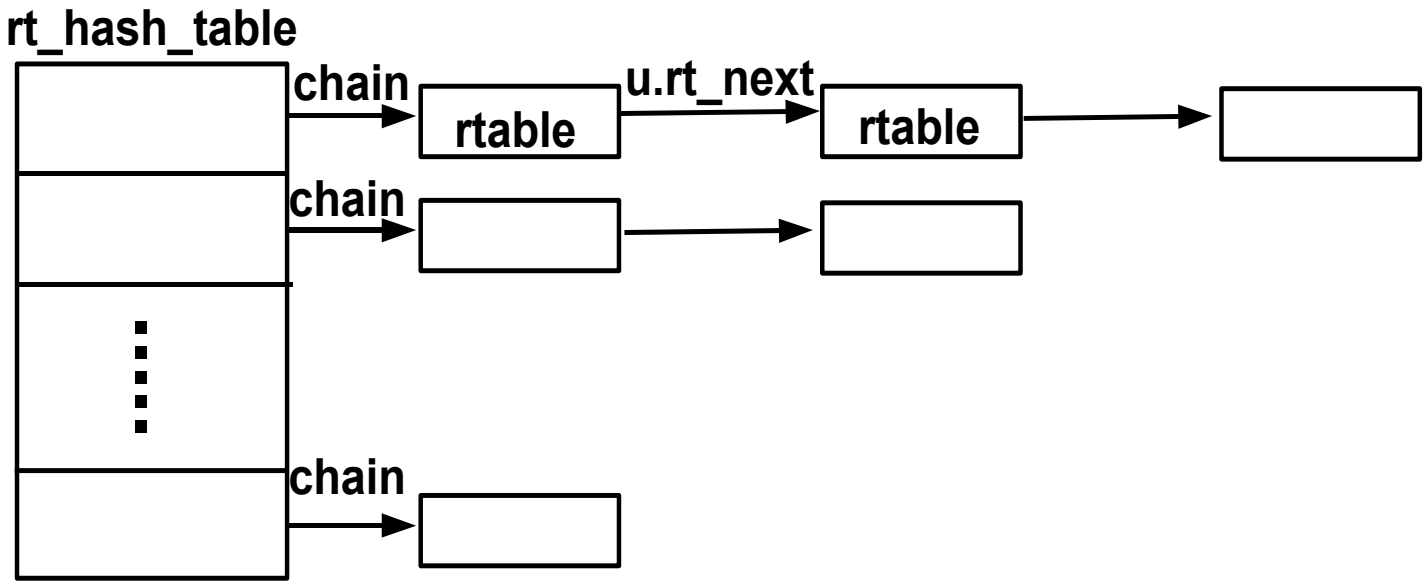
Longest prefix matching

- longest prefix matching: often performed using ternary content addressable memories (TCAMs)
  - ✓ *content addressable:* present address to TCAM: retrieve address in one clock cycle, regardless of table size
  - ✓ Cisco Catalyst: can up ~1M routing table entries in TCAM

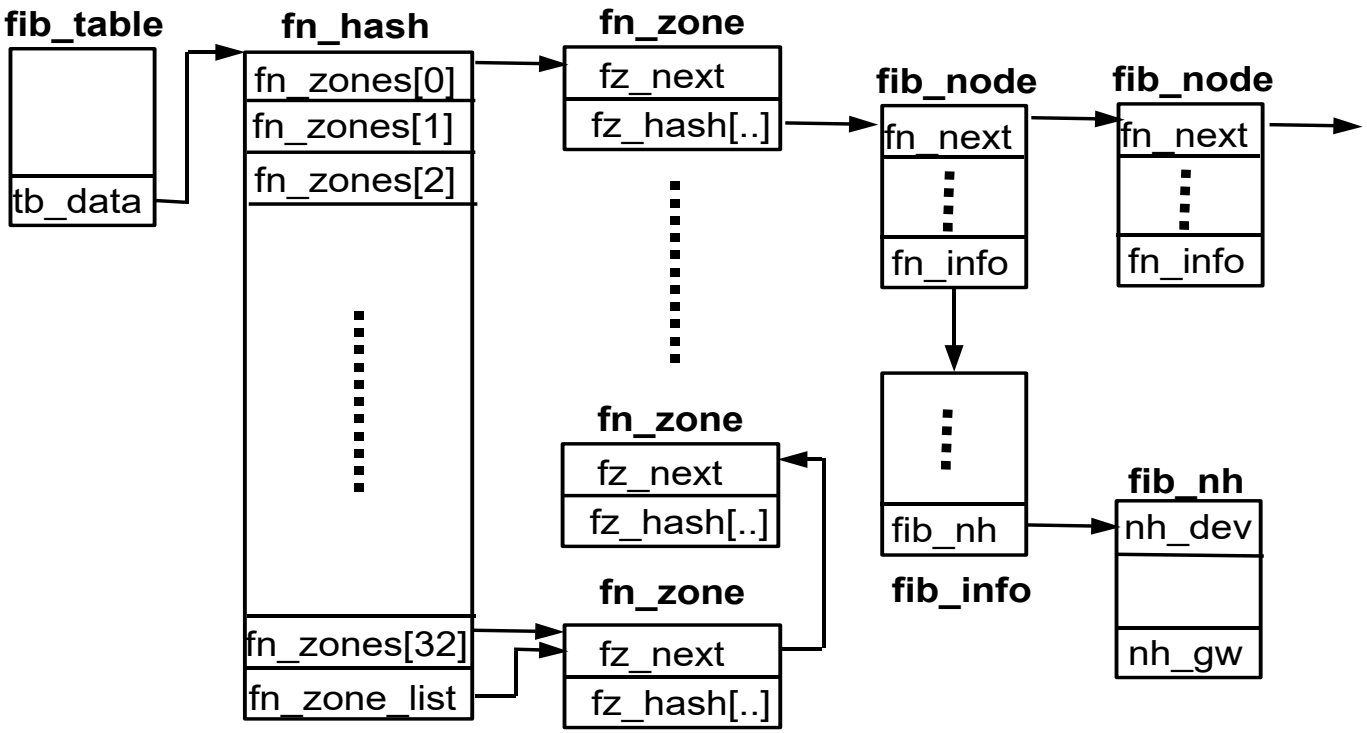
Consulta a Tabela no Linux



Cache para Acesso a Tabela



Estrutura da Tabela de Roteamento



Circuitos virtuais

"caminho fixo da-origem-ao-destino

- estabelecimento de cada chamada *antes* do envio dos dados
- cada pacote tem ident. de CV (e não endereços origem/dest)
- *cada* roteador no caminho da-origem-ao-destino mantém "estado" para cada conexão que o atravessa
  - ✓ conexão da camada de transporte só envolve os 2 sistemas terminais
- recursos de enlace, roteador (banda, *buffers*) podem ser *alocados* ao CV
  - ✓ para permitir desempenho como de um circuito

Implementação CV

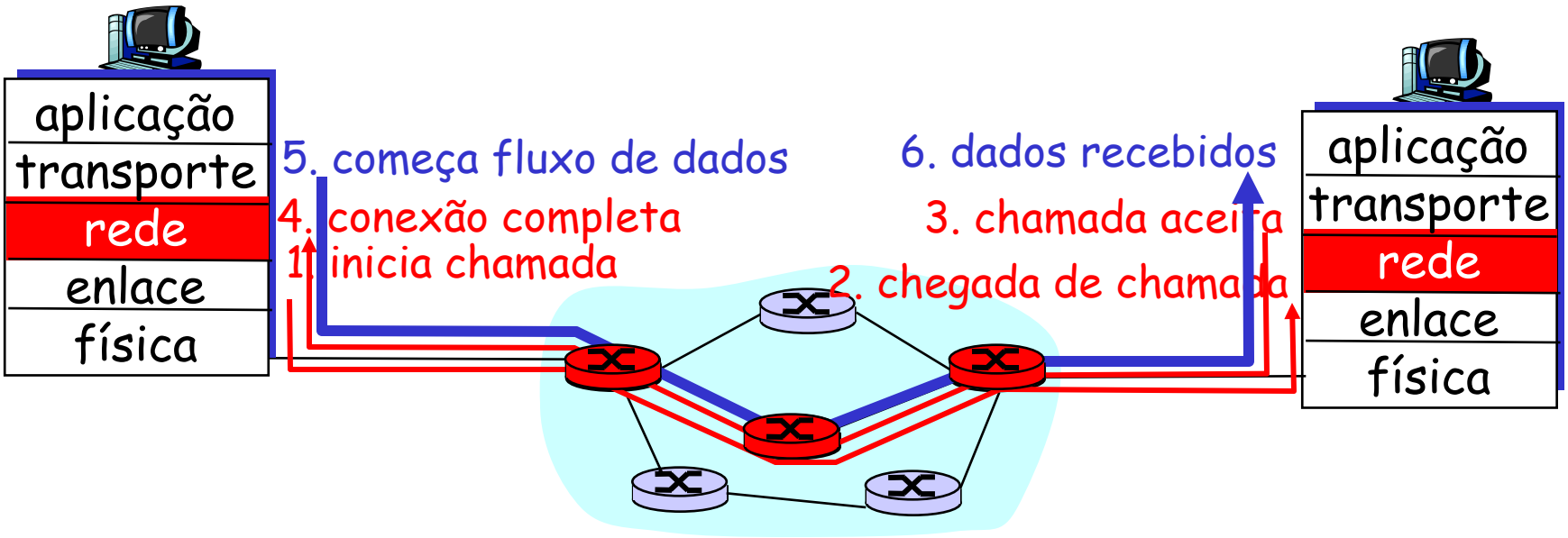
Um circuito virtual consiste de:

1. Caminho entre origem e destino
2. Identificador CV, um para cada enlace ao longo do caminho
3. Entradas nas tabelas de roteamento nos roteadores ao longo do caminho

- Pacote carrega identificador de CV ao invés de endereço destino
- Identificador de CV pode mudar a cada enlace
  - ✓ Novos identificadores de VC são gerados nas tabelas de roteamento

Circuitos virtuais: protocolos de sinalização

- usados para estabelecer, manter, destruir CV
- usados em ATM, frame-relay, X.25
- não usados na Internet de hoje



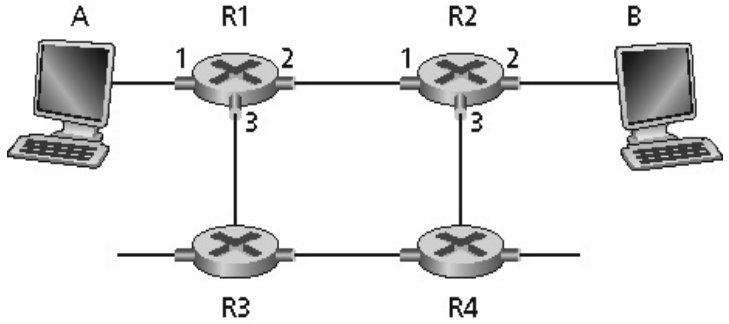


Tabela de comutação no roteador a noroeste:

Interface de entrada	VC # de entrada	Interface de saída	VC # de saída
1	12	2	22
2	63	1	18
3	7	2	17
1	97	3	87
...	...	...	...

Roteadores mantêm informações de estado de conexão

Rede de datagramas ou CVs: por quê?

Internet

- troca de dados entre computadores
  - ✓ serviço "elástico", sem reqs. temporais estritos
- sistemas terminais "inteligentes" (computadores)
  - ✓ podem se adaptar, exercer controle, recuperar de erros
  - ✓ núcleo da rede simples, complexidade na "borda"
- muitos tipos de enlaces
  - ✓ características diferentes

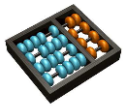
ATM/Frame Relay/X.25

- evoluiu da telefonia
- sistemas terminais "burros"
  - ✓ telefones
  - ✓ complexidade dentro da rede

Circuitos Virtuais

- ATM: alguns sistemas de telecomunicações e financeiros legados
- Frame Relay: alguns sistemas coorperativo legado
- X.25: alguns sistemas militares legados
- MPLS: emulação de CV – caminhos pré-definidos, label-switching
- SDN com segment routing – emulação de CVs

UNICAMP



Network service model

Q: What *service model* for "channel" transporting datagrams from sender to receiver?

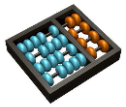
example services for *individual* datagrams:

- guaranteed delivery
- guaranteed delivery with less than 40 msec delay

example services for a *flow* of datagrams:

- in-order datagram delivery
- guaranteed minimum bandwidth to flow
- restrictions on changes in inter-packet spacing

UNICAMP



Network-layer service model

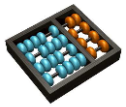
Network Architecture	Service Model	Quality of Service (QoS) Guarantees ?			
		Bandwidth	Loss	Order	Timing
Internet	best effort	none	no	no	no

Internet "best effort" service model

No guarantees on:

i. successful datagram delivery to destination
ii. timing or order of delivery
iii. bandwidth available to end-end flow

UNICAMP



Network-layer service model

Network Architecture	Service Model	Quality of Service (QoS) Guarantees ?			
		Bandwidth	Loss	Order	Timing
Internet	best effort	none	no	no	no
ATM	Constant Bit Rate	Constant rate	yes	yes	yes
ATM	Available Bit Rate	Guaranteed min	no	yes	no
Internet	Intserv Guaranteed (RFC 1633)	yes	yes	yes	yes
Internet	Diffserv (RFC 2475)	possible	possibly	possibly	no

UNICAMP

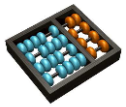


Tabela 2.3: Características dos portadores de tráfego para rede móvel celular 3GPP.

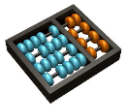
QCI	Tipo do recurso	Prioridade	Atraso máximo (ms)	Taxa de erro	Serviços
1	GBR	2	100	10^{-2}	Voz
2	GBR	4	150	10^{-3}	Voz (live streaming)
3	GBR	3	50	10^{-3}	Jogo em tempo real
4	GBR	5	300	10^{-6}	Vídeo (buffered streaming)
5	Non-GBR	1	100	10^{-6}	Sinalização IMS
6	Non-GBR	6	300	10^{-6}	Vídeo (buffered streaming) TCP-based (HTTP, P2P, etc.)
7	Non-GBR	7	100	10^{-3}	Voz, Vídeo (live streaming)
8	Non-GBR	8	300	10^{-6}	Vídeo (buffered streaming) TCP-based (HTTP, P2P, etc.)
9	Non-GBR	9	300	10^{-6}	Vídeo (buffered streaming) TCP-based (HTTP, P2P, etc.)

Reflections on best-effort service:

- simplicity of mechanism has allowed Internet to be widely deployed adopted
- sufficient provisioning of bandwidth allows performance of real-time applications (e.g., interactive voice, video) to be "good enough" for "most of the time"
- replicated, application-layer distributed services (datacenters, content distribution networks) connecting close to clients' networks, allow services to be provided from multiple locations
- congestion control of "elastic" services helps

It's hard to argue with success of best-effort service model

UNICAMP



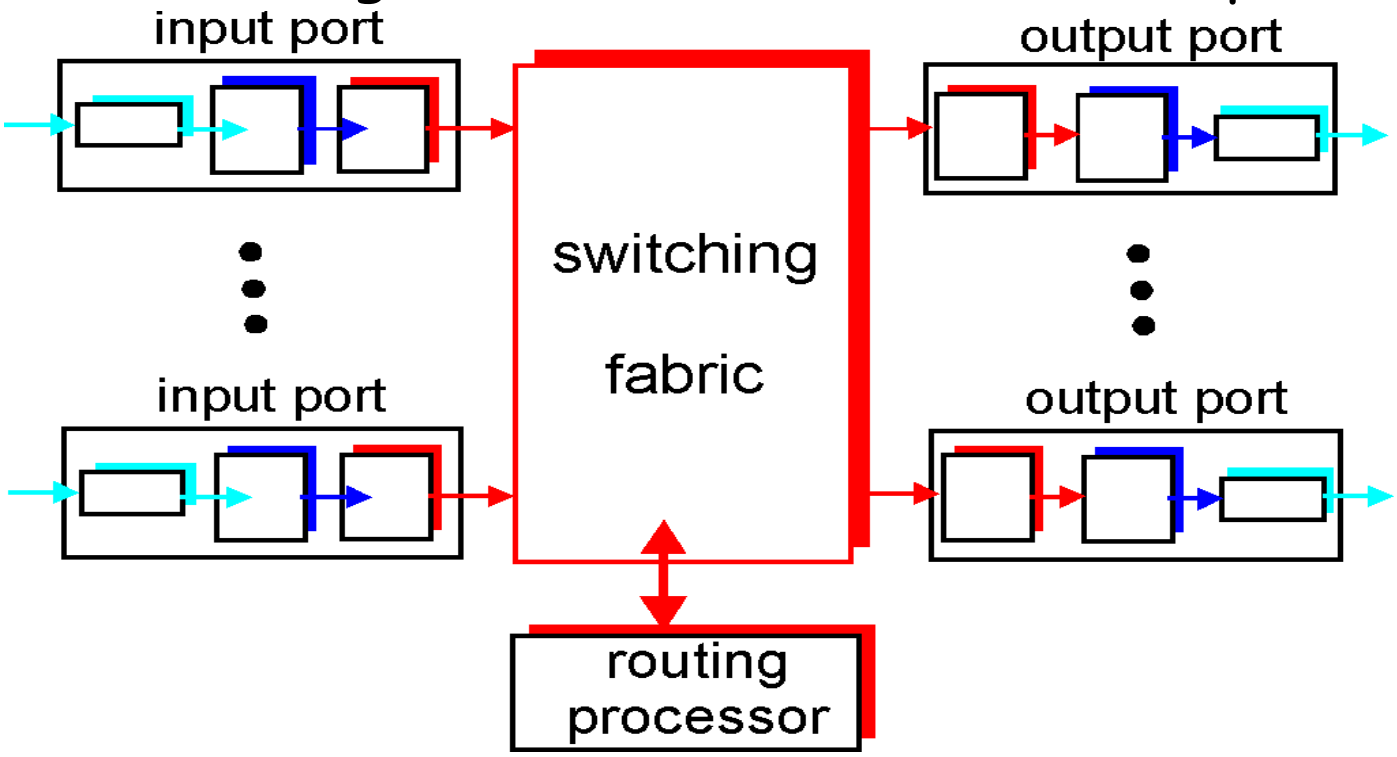
Roteiro

- 4. 1 Introdução
- 4.2 Circuitos virtuais x datagrama
- 4.3 Como é um roteador
- 4.4 Protocolo IP
  - Formato datagarma
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - Estado de enlace
  - Vetor distância
  - Roteamento hierarquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento Broadcast e multicast

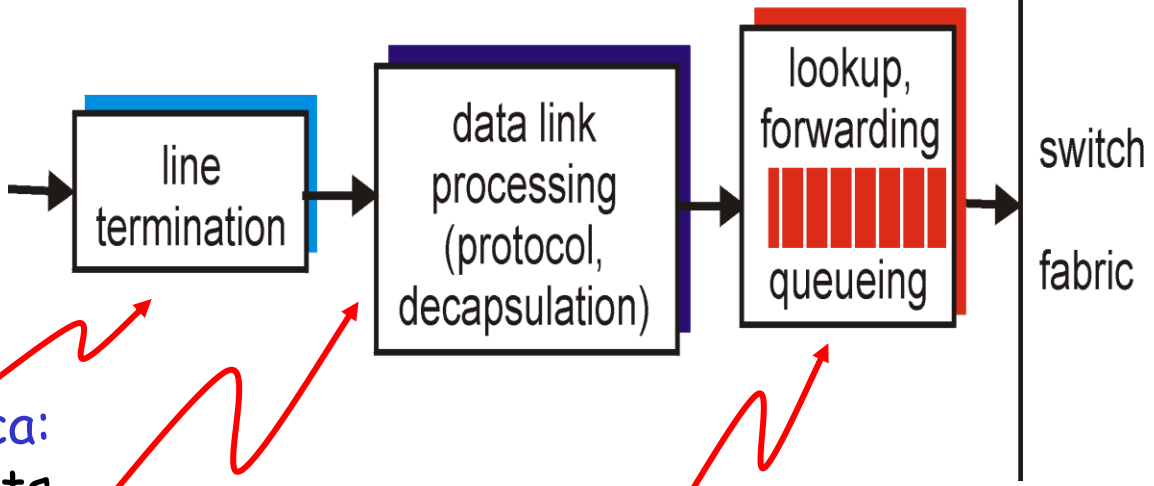
Sumário de Arquitetura de Roteadores

Duas funções chave de roteadores:

- usam algoritmos/protocolos de roteamento (RIP, OSPF, BGP)
- *comutam* datagramas do enlace de entrada para a saída



Funções da Porta de Entrada



Camada f'ísica:
recepção de bits

Camada de enlace:
p.ex., Ethernet
veja capítulo 5

Comutação descentralizada:

- dado o dest do datagrama, procura porta de saída usando tab. de rotas na memória da porta de entrada
- meta: completar processamento da porta de entrada na 'velocidade da linha'
- filas: se datagramas chegam mais rápido que taxa de re-envio para matriz de comutação

Tamanho do Buffer

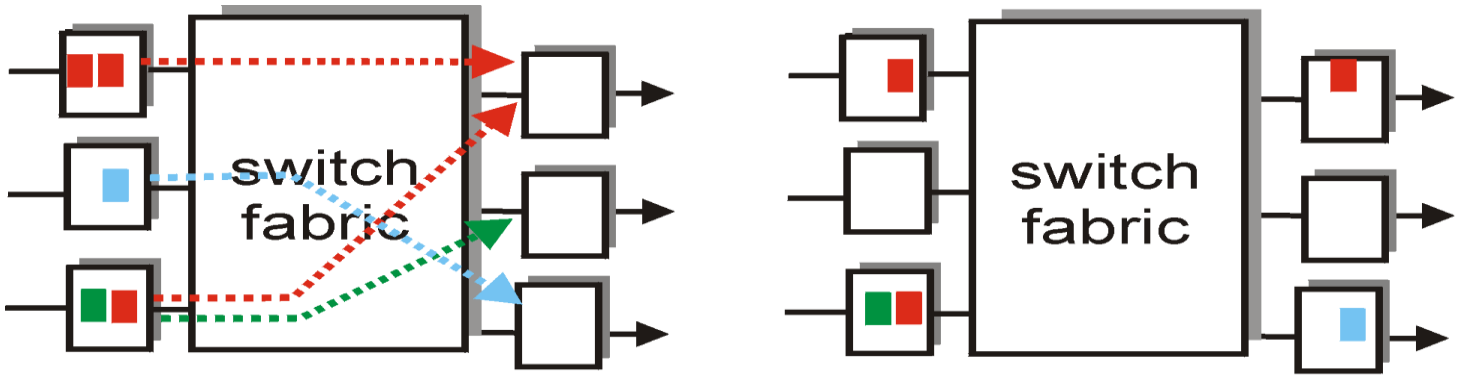
- RFC 3439 recomenda que tamano médio do buffer deve corresponder a um RTT típico vezes a capacidade do enlace
  - ✓ 250 seg, C = 10 Gps link: 2.5 Gbit buffer
- Recomendação recente para N fluxos:

$$\frac{RTT \cdot C}{\sqrt{N}}$$

  - but *too* much buffering can increase delays (particularly in home routers)
    - long RTTs: poor performance for real-time apps, sluggish TCP response
    - recall delay-based congestion control: “keep bottleneck link just full enough (busy) but no fuller”

Filas na Porta de Entrada

- Se matriz de comutação for mais lenta do que a soma das portas de entrada juntas -> pode haver filas nas portas de entrada
- Bloqueio cabeça-de-linha (Head-of-the-Line - HOL): datagrama na cabeça da fila impede outros na mesma fila de avançarem
- *retardo de enfileiramento e perdas devido ao transbordo do buffer de entrada!*

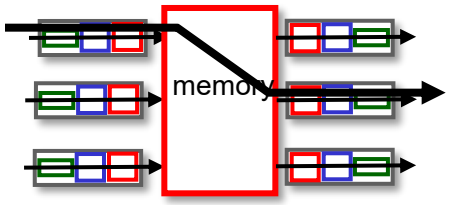


output port contention at time t - only one red packet can be transferred

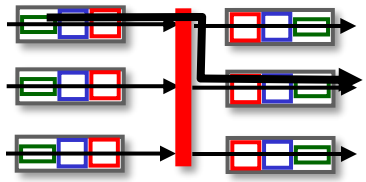
green packet experiences HOL blocking

Switching fabrics

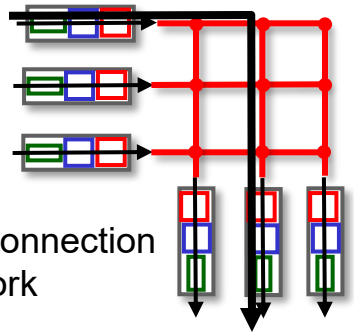
- transfer packet from input link to appropriate output link
- switching rate: rate at which packets can be transfer from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- three major types of switching fabrics:



memory

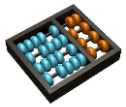


bus



interconnection network

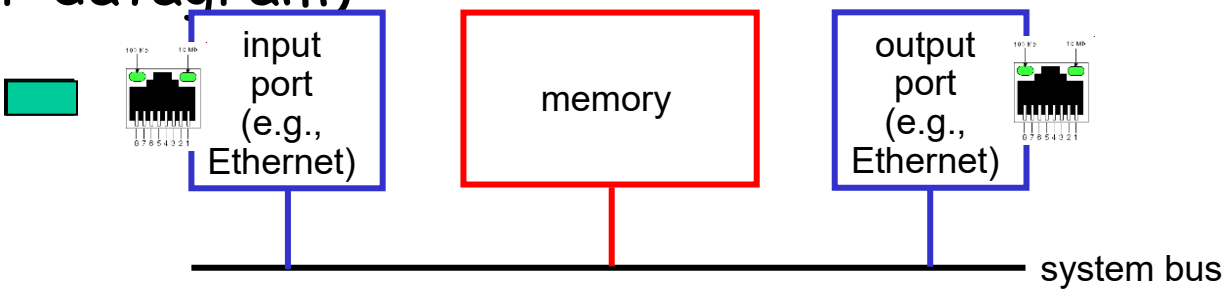
UNICAMP



Switching via memory

first generation routers:

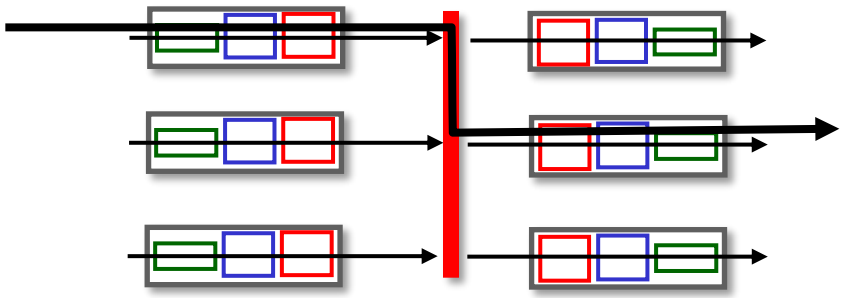
- traditional computers with switching under direct control of CPU
- packet copied to system's memory
- speed limited by memory bandwidth (2 bus crossings per datagram)



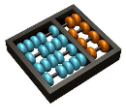
UNICAMP

Switching via a bus

- datagram from input port memory to output port memory via a shared bus
- *bus contention:* switching speed limited by bus bandwidth
- 32 Gbps bus, Cisco 5600: sufficient speed for access routers

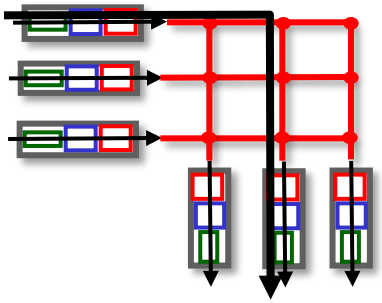


UNICAMP

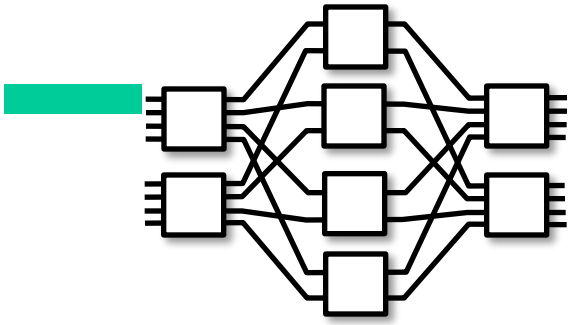


Switching via interconnection network

- Crossbar, Clos networks, other interconnection nets initially developed to connect processors in multiprocessor
- multistage switch: *nxn* switch from multiple stages of smaller switches
- exploiting parallelism:
  - fragment datagram into fixed length cells on entry
  - switch cells through the fabric, reassemble datagram at exit

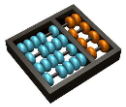


3x3 crossbar



8x8 multistage switch
built from smaller-sized switches

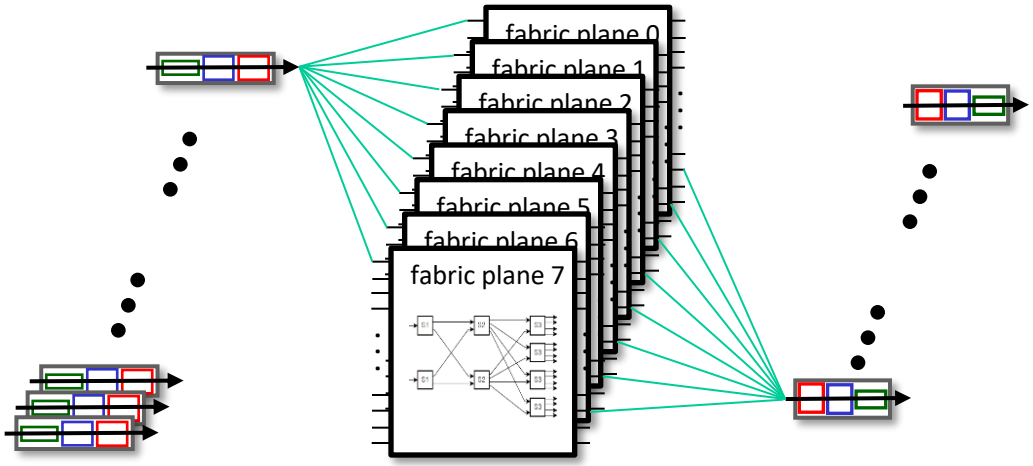
UNICAMP



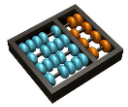
Switching via interconnection network

- scaling, using multiple switching "planes" in parallel:
  - speedup, scaleup via parallelism

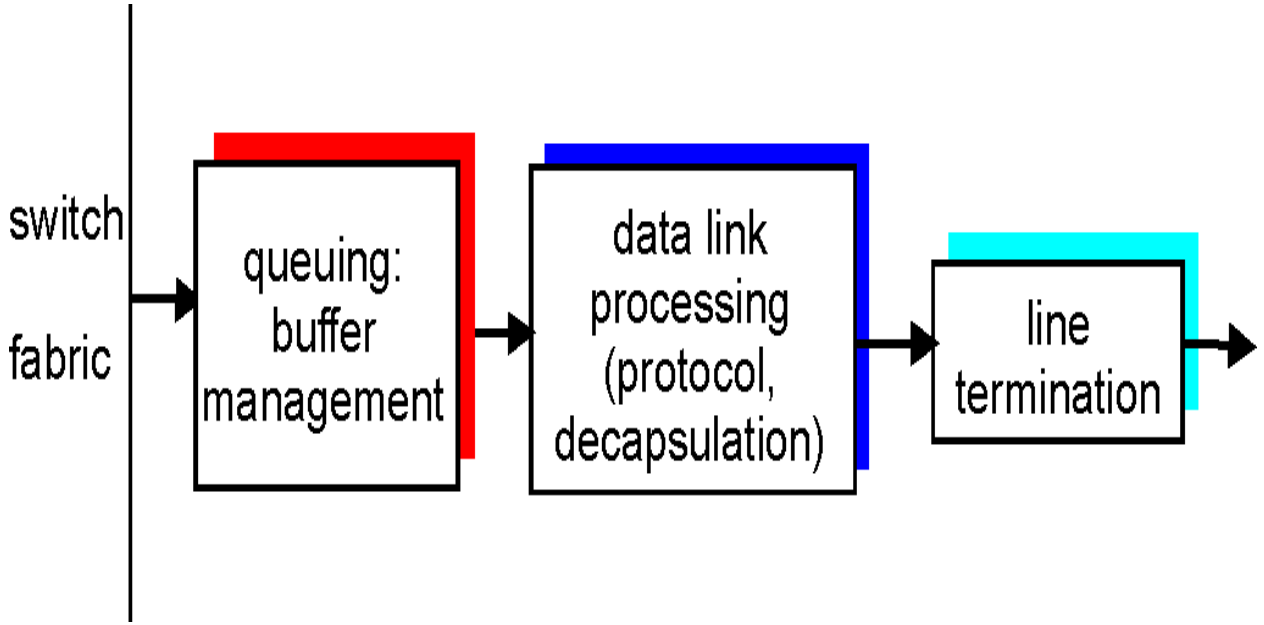
- Cisco CRS router:
  - basic unit: 8 switching planes
  - each plane: 3-stage interconnection network
  - up to 100's Tbps switching capacity



UNICAMP

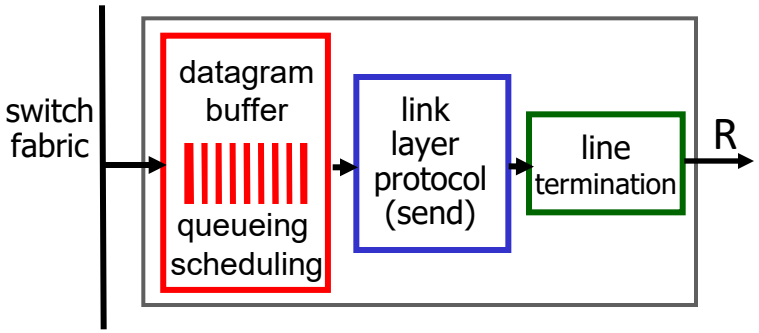


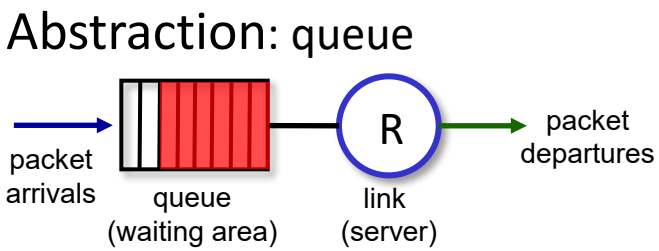
Porta de Saída



- *Buffers* necessários quando datagramas chegam da matriz de comutação mais rapidamente que a taxa de transmissão
- *Disciplina de escalonamento* escolhe um dos datagramas enfileirados para transmissão

Buffer Management

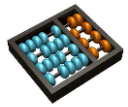




buffer management:

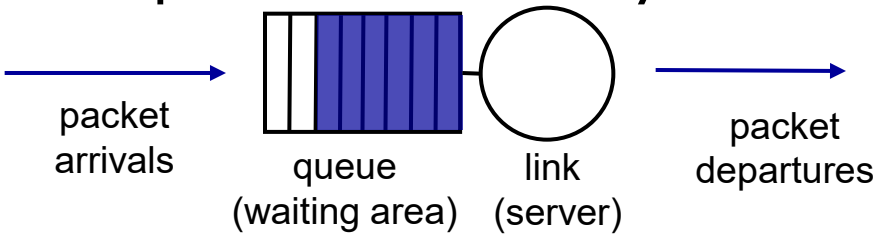
- drop: which packet to add, drop when buffers are full
  - tail drop: drop arriving packet
  - priority: drop/remove on priority basis
- marking: which packets to mark to signal congestion (ECN, RED)

UNICAMP



Scheduling mechanisms

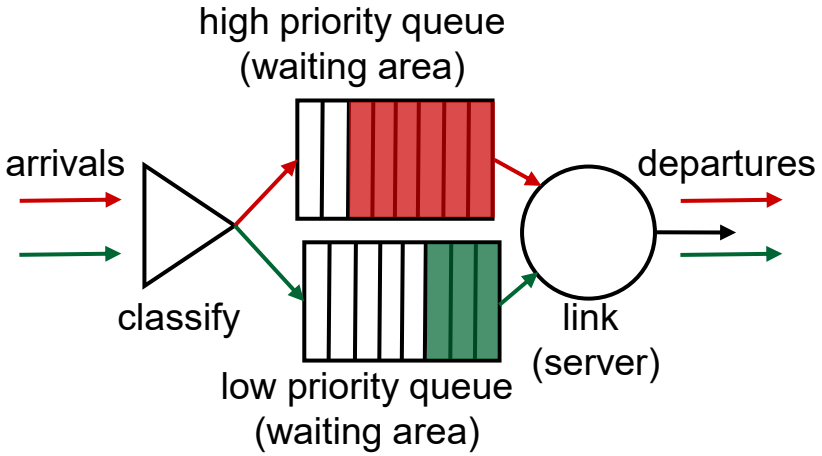
- *scheduling:* choose next packet to send on link
- *FIFO (first in first out) scheduling:* send in order of arrival to queue
  - ✓ real-world example?
  - ✓ *discard policy:* if packet arrives to full queue: who to discard?
    - *tail drop:* drop arriving packet
    - *priority:* drop/remove on priority basis
    - *random:* drop/remove randomly

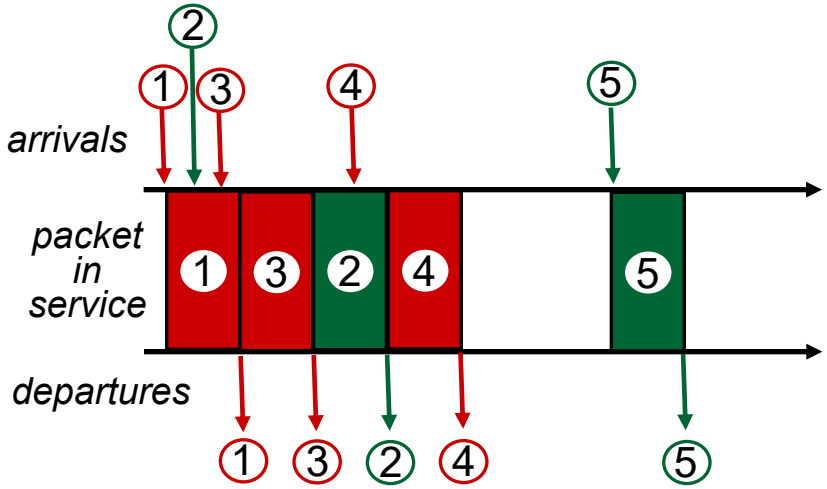


Scheduling policies: priority

priority scheduling: send highest priority queued packet

- multiple *classes*, with different priorities
  - class may depend on marking or other header info, e.g. IP source/dest, port numbers, etc.
  - real world example?

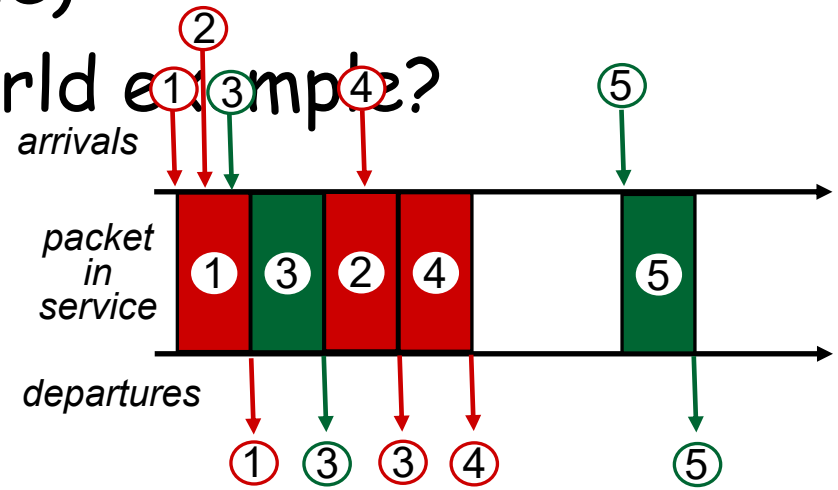




Scheduling policies: still more

Round Robin (RR) scheduling:

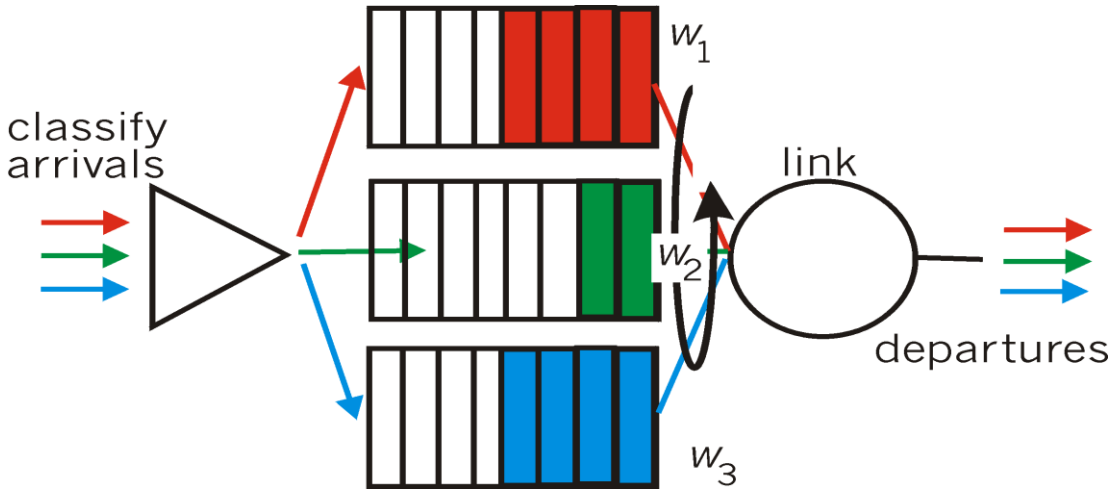
- multiple classes
- cyclically scan class queues, sending one complete packet from each class (if available)
- real world example?



Scheduling policies: still more

Weighted Fair Queuing (WFQ):

- generalized Round Robin
- each class gets weighted amount of service in each cycle
- real-world example?



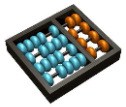
Sidebar: Network Neutrality

What is network neutrality?

- *technical:* how an ISP should share/allocation its resources
  - ✓ packet scheduling, buffer management are the *mechanisms*
- *social, economic* principles
  - protecting free speech
  - encouraging innovation, competition
- enforced *legal* rules and policies

Different countries have different "takes" on network neutrality

UNICAMP

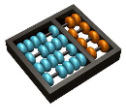


Sidebar: Network Neutrality

2015 US FCC *Order on Protecting and Promoting an Open Internet*: three "clear, bright line" rules:

- no blocking … "shall not block lawful content, applications, services, or non-harmful devices, subject to reasonable network management."
- no throttling … "shall not impair or degrade lawful Internet traffic on the basis of Internet content, application, or service, or use of a non-harmful device, subject to reasonable network management."
- no paid prioritization. … "shall not engage in paid prioritization"

UNICAMP



ISP: telecommunications or information service?

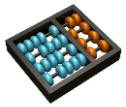
Is an ISP a "telecommunications service" or an "information service" provider?

- the answer *really* matters from a regulatory standpoint!

US Telecommunication Act of 1934 and 1996:

- ✓ *Title II:* imposes "common carrier duties" on *telecommunications services*: reasonable rates, non-discrimination and *requires regulation*
- ✓ *Title I:* applies to *information services:*
  - no common carrier duties (*not regulated*)
  - but grants FCC authority "… as may be necessary in the execution of its functions".

UNICAMP

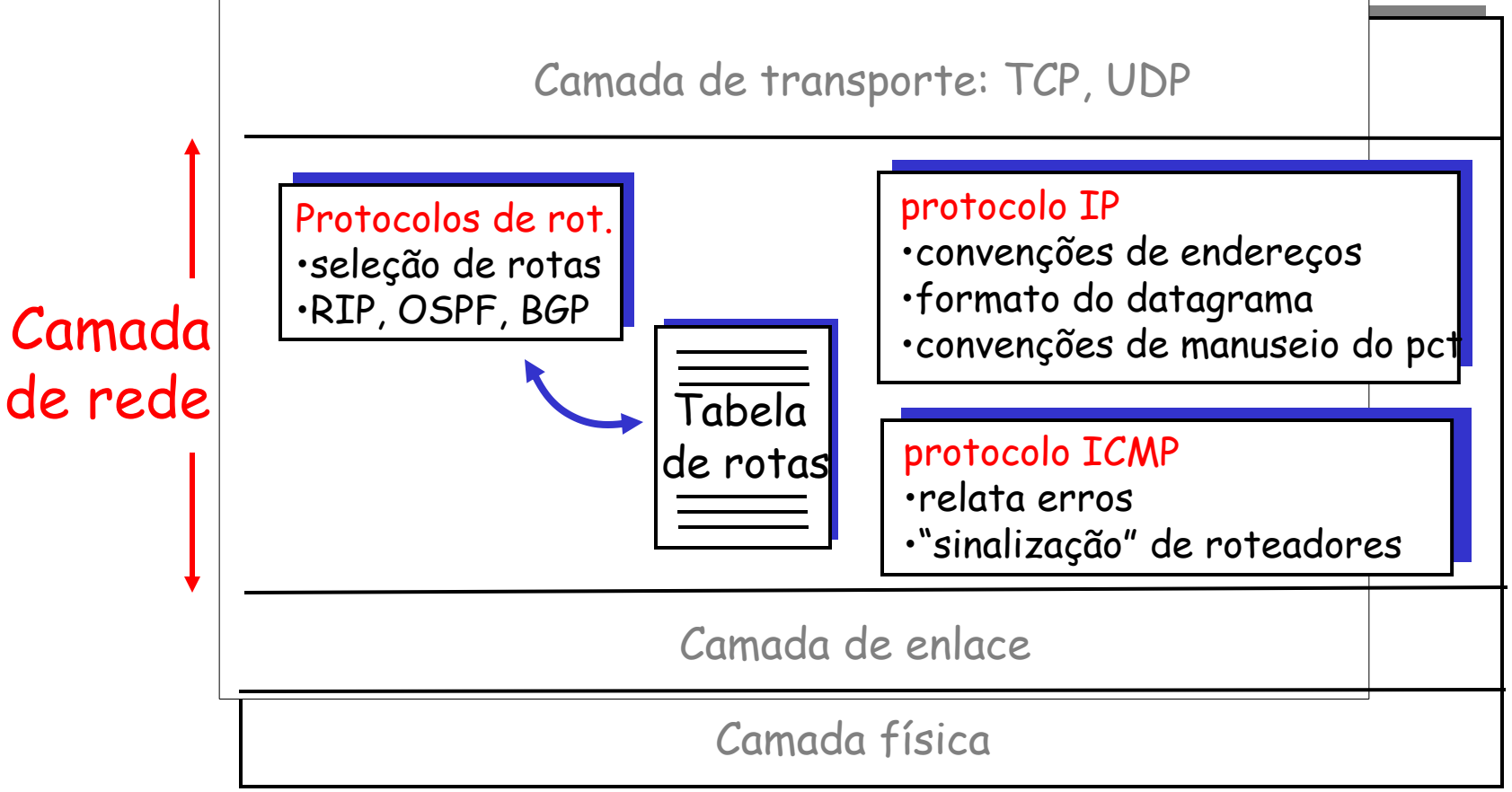


Roteiro

- 4. 1 Introdução
- 4.2 Circuitos virtuais x datagrama
- 4.3 Como é um roteador
- 4.4 Protocolo IP
  - Formato datagarma
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - Estado de enlace
  - Vetor distância
  - Roteamento hierarquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento Broadcast e multicast

A Camada de Rede na Internet

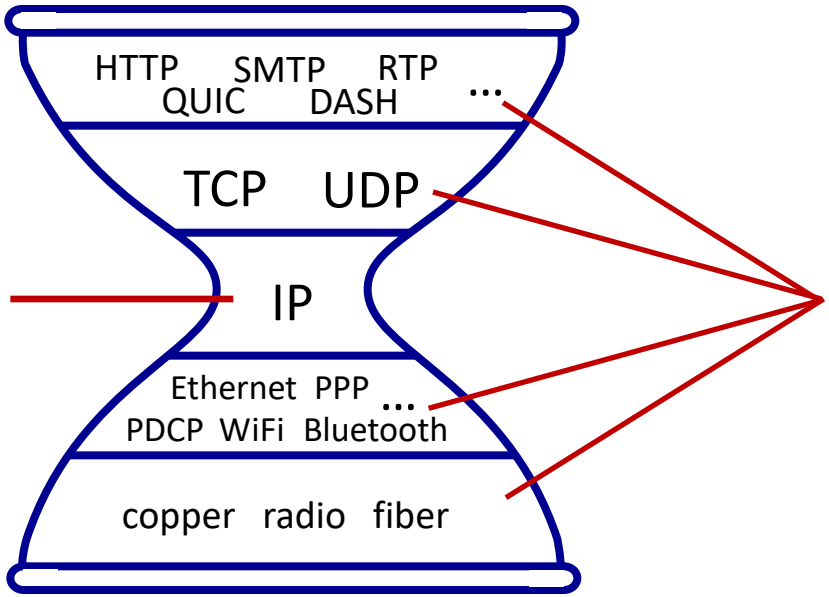
Funções da camada de rede em estações, roteadores:



The IP hourglass

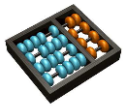
Internet's "thin waist":

- *one* network layer protocol: IP
- *must* be implemented by every (billions) of Internet-connected devices



many protocols in physical, link, transport, and application layers

UNICAMP

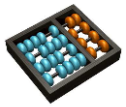


Middleboxes

Middlebox (RFC 3234)

"any intermediary box performing functions apart from normal, standard functions of an IP router on the data path between a source host and destination host"

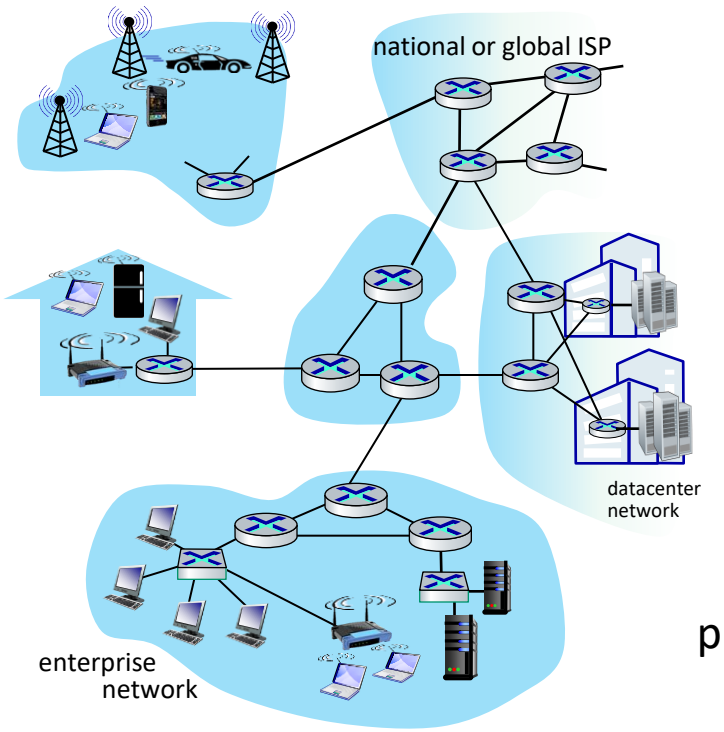
UNICAMP



Middleboxes everywhere!

NAT: home, cellular, institutional

Application-specific: service providers, institutional, CDN

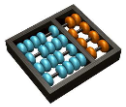


Firewalls, IDS: corporate, institutional, service providers, ISPs

Load balancers: corporate, service provider, data center, mobile nets

Caches: service provider, mobile, CDNs

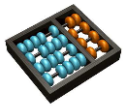
UNICAMP



Middleboxes

- initially: proprietary (closed) hardware solutions
- move towards "whitebox" hardware implementing open API
  - move away from proprietary hardware solutions
  - programmable local actions via match+action
  - move towards innovation/differentiation in software
- SDN: (logically) centralized control and configuration management often in private/public cloud
- network functions virtualization (NFV): programmable services over white box networking, computation, storage

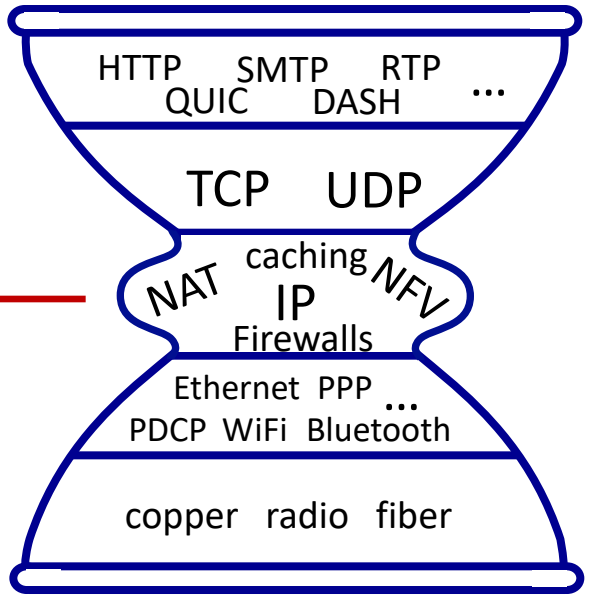
UNICAMP



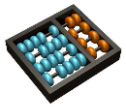
The IP hourglass, at middle age

Internet's middle age "love handles"?

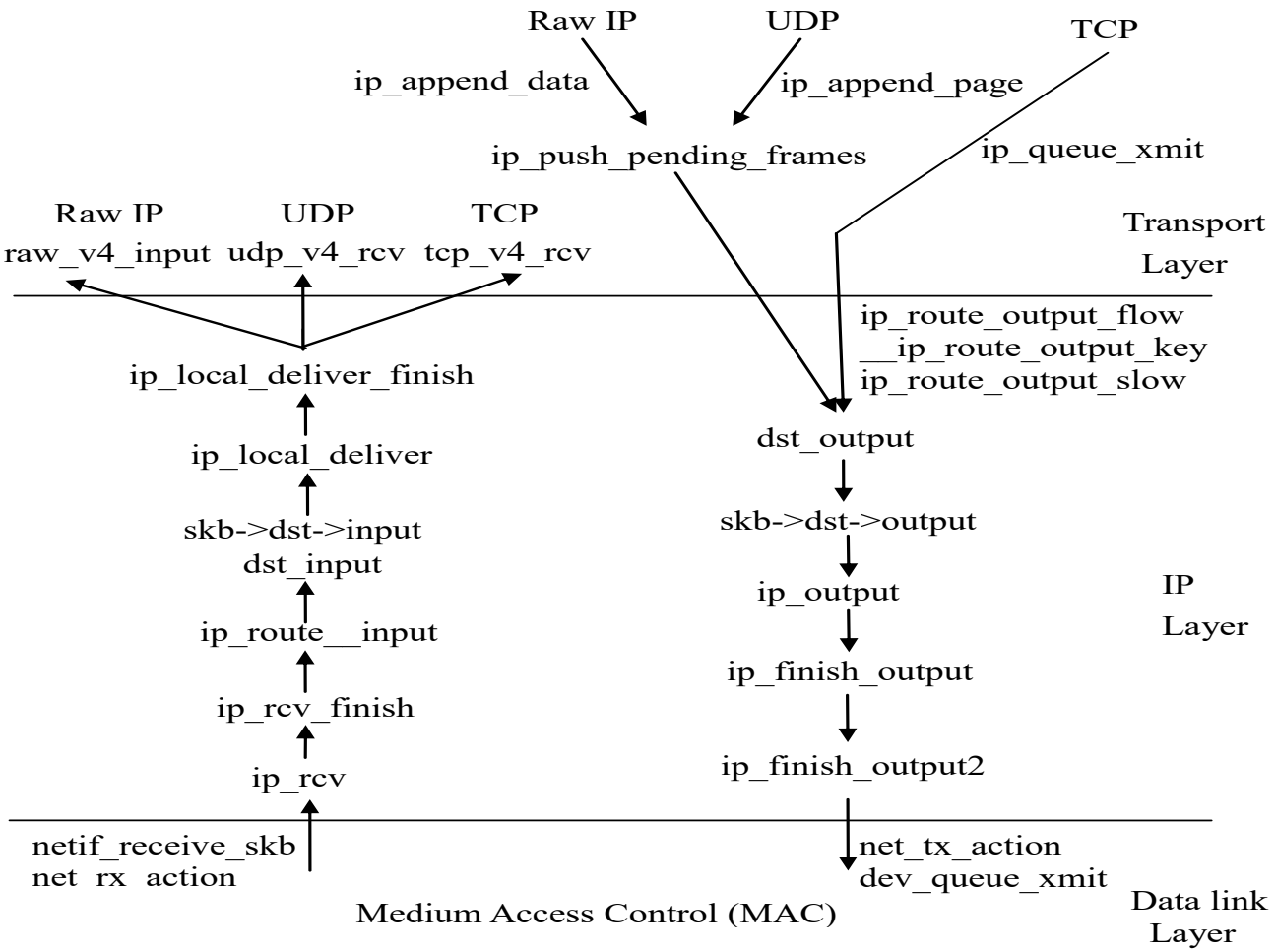
- middleboxes, operating inside the network



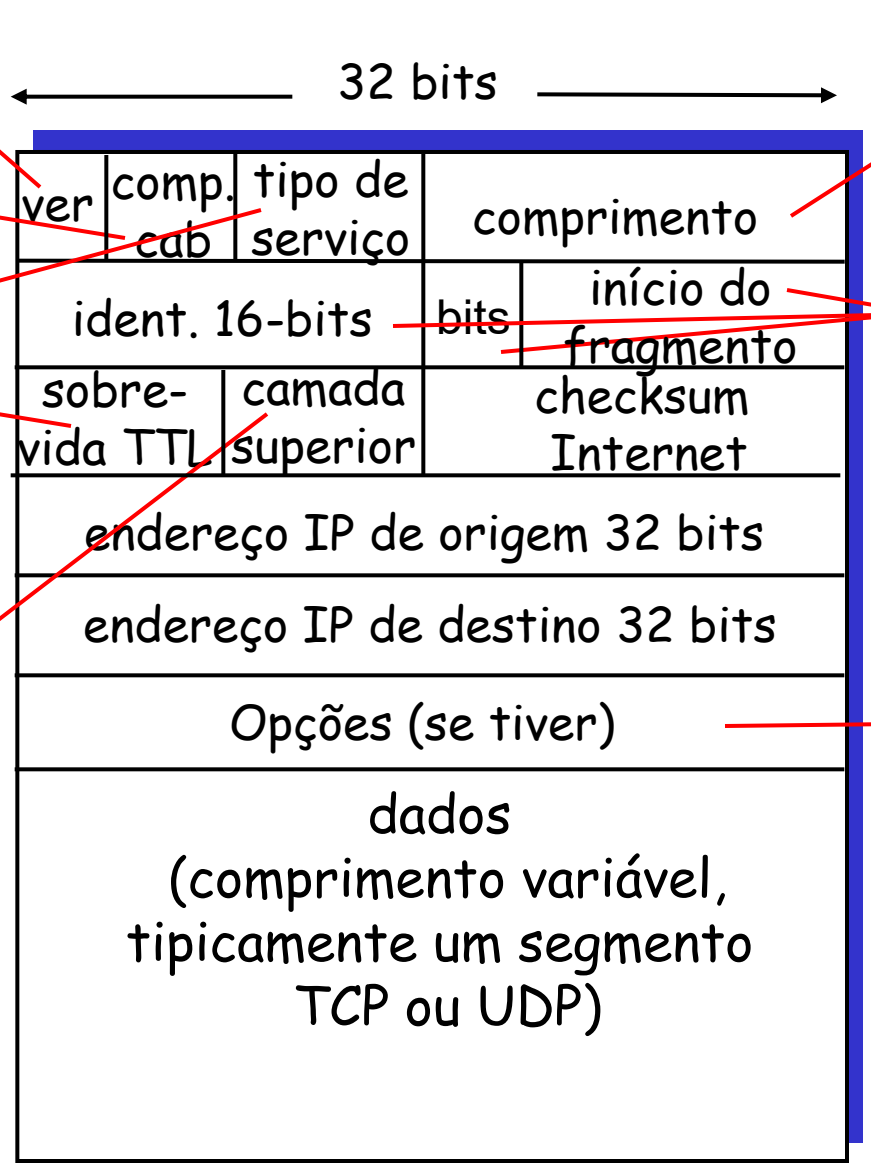
UNICAMP



Implementação IP - Linux



Formato do datagrama IP



número da versão do protocolo IP

comprimento do cabeçalho (bytes)

"tipo" dos dados (DS)

número máximo de enlaces restantes (decrementado a cada roteador)

protocolo da camada superior ao qual entregar os dados

comprimento total do datagrama (bytes)

para fragmentação/ remontagem

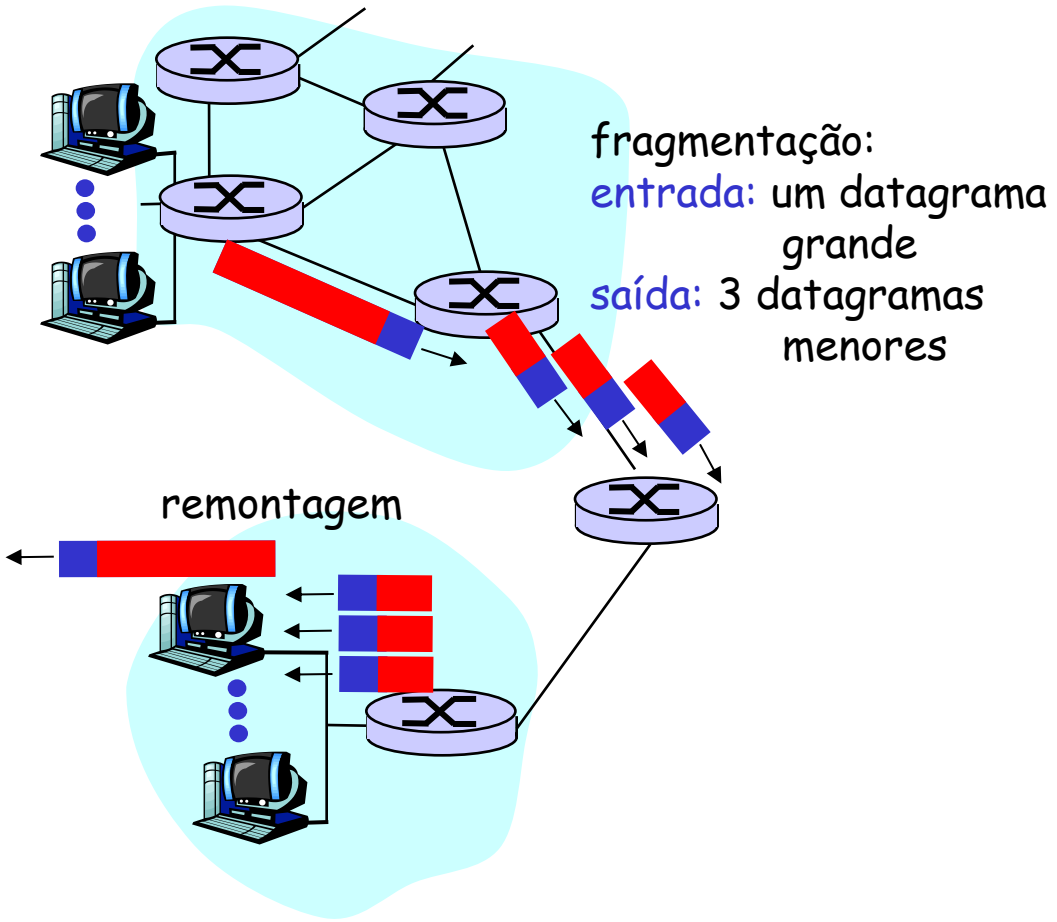
p.ex. temporizador, registrar rota seguida, especificar lista de roteadores a visitar.

Qual o overhead com TCP?

- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + overhead aplic.

IP: Fragmentação & Remontagem

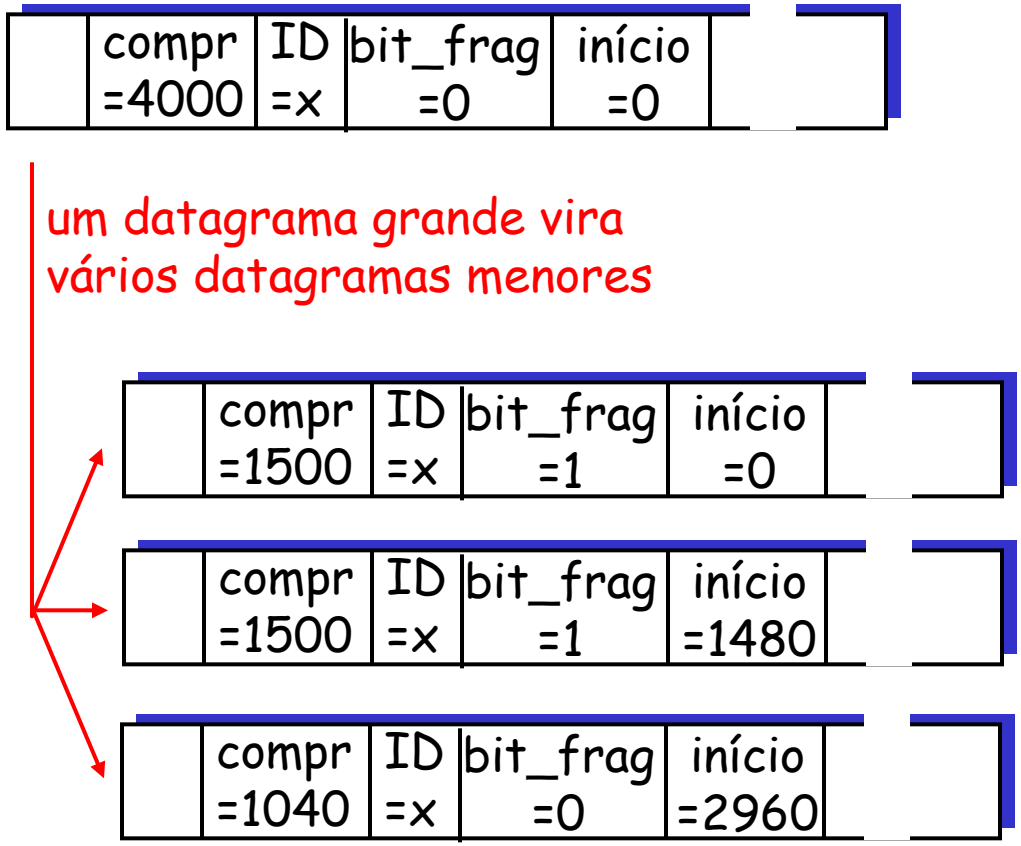
- cada enlace de rede tem MTU (max.transmission unit) - maior tamanho possível de quadro neste enlace.
  - tipos diferentes de enlace têm MTUs diferentes
- datagrama IP muito grande dividido ("fragmentado") dentro da rede
  - um datagrama vira vários datagramas
  - "remontado" apenas no destino final
  - bits do cabeçalho IP usados para identificar, ordenar fragmentos relacionados



IP: Fragmentação & Remontagem

Exemplo

- Datagrama com 4000 bytes
- MTU = 1500 bytes

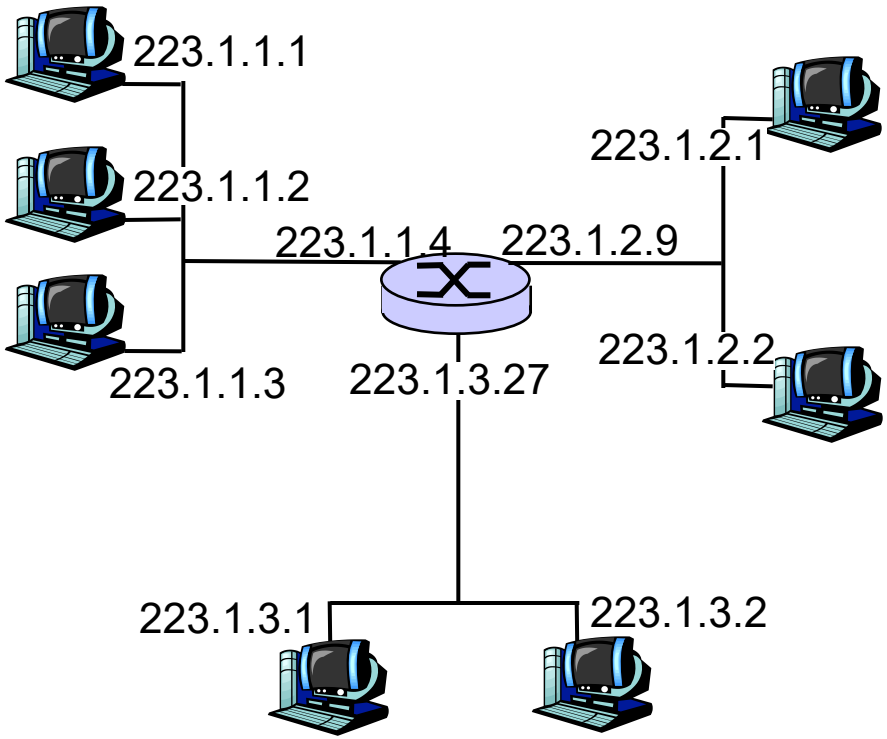


Roteiro

- 4. 1 Introdução
- 4.2 Circuitos virtuais x datagrama
- 4.3 Como é um roteador
- 4.4 Protocolo IP
  - Formato datagarma
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - Estado de enlace
  - Vetor distância
  - Roteamento hierarquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento Broadcast e multicast

Endereçamento IP: introdução

- endereço IP: ident. de 32-bits para *interface* de estação, roteador
- *interface:* conexão entre estação, roteador e enlace físico
  - ✓ roteador típico tem múltiplas interfaces
  - ✓ estação pode ter múltiplas interfaces
  - ✓ endereço IP associado à interface

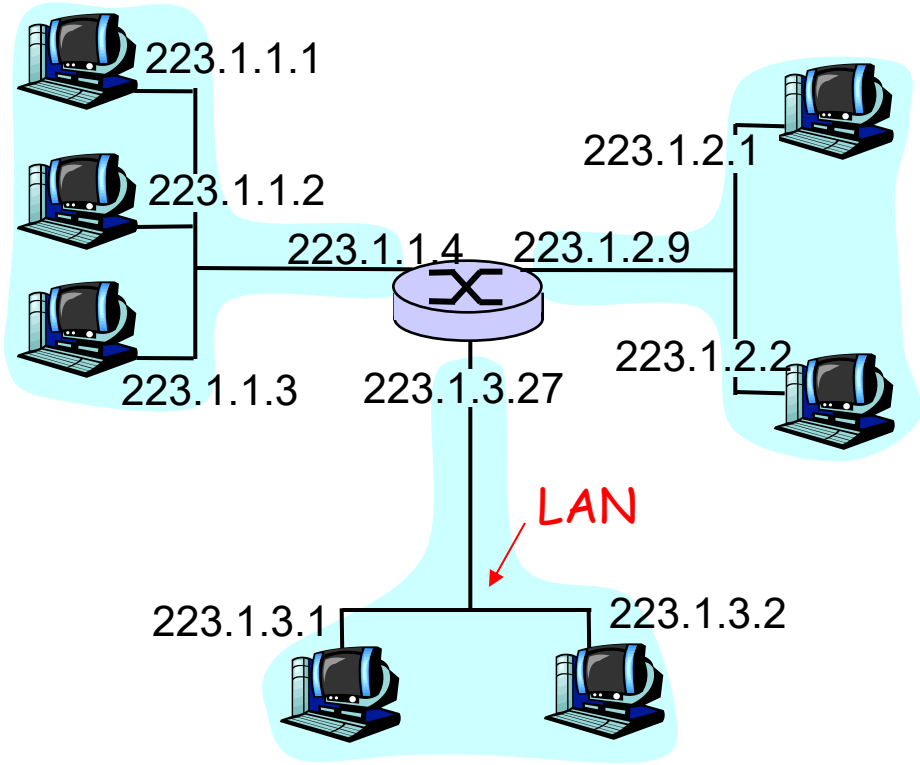


223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

Endereçamento IP

- endereço IP:
  - ✓ parte de rede (bits de mais alta ordem)
  - ✓ parte de estação (bits de mais baixa ordem)
- *O que é uma rede IP?* (da perspectiva do endereço IP)
  - ✓ interfaces de dispositivos com a mesma parte de rede nos seus endereços IP
  - ✓ podem alcançar um ao outro sem passar por um roteador



Esta rede consiste de 3 redes IP (para endereços IP começando com 223, os primeiros 24 bits são a parte de rede)

Enviando um datagrama da origem ao destino

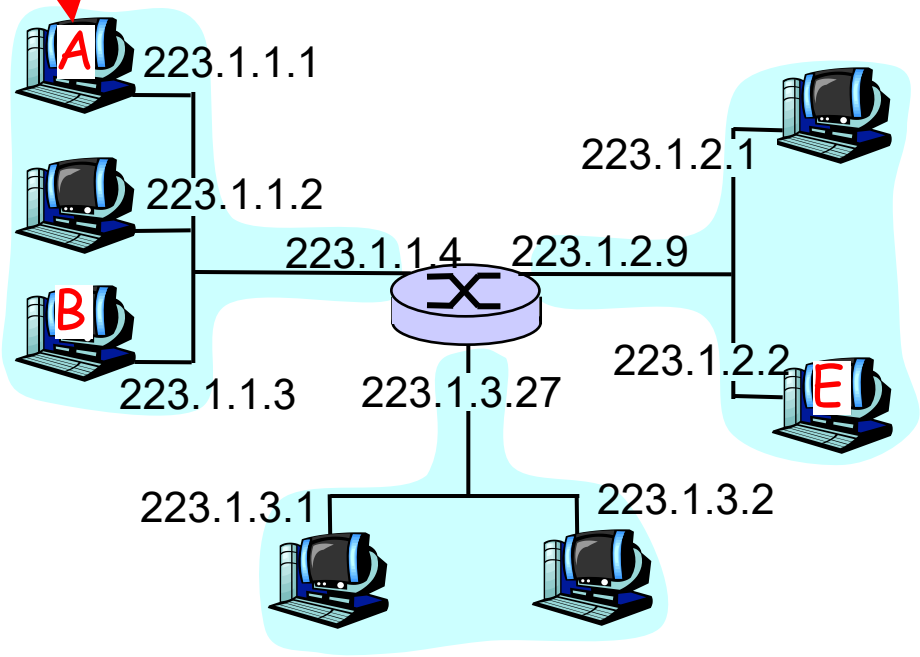
datagrama IP:

aampos misc	end. IP origem	end. IP dest	dados

- datagrama permanece inalterado, enquanto passa da origem ao destino
- campos de endereços de interesse aqui

tabela de rotas em *A*

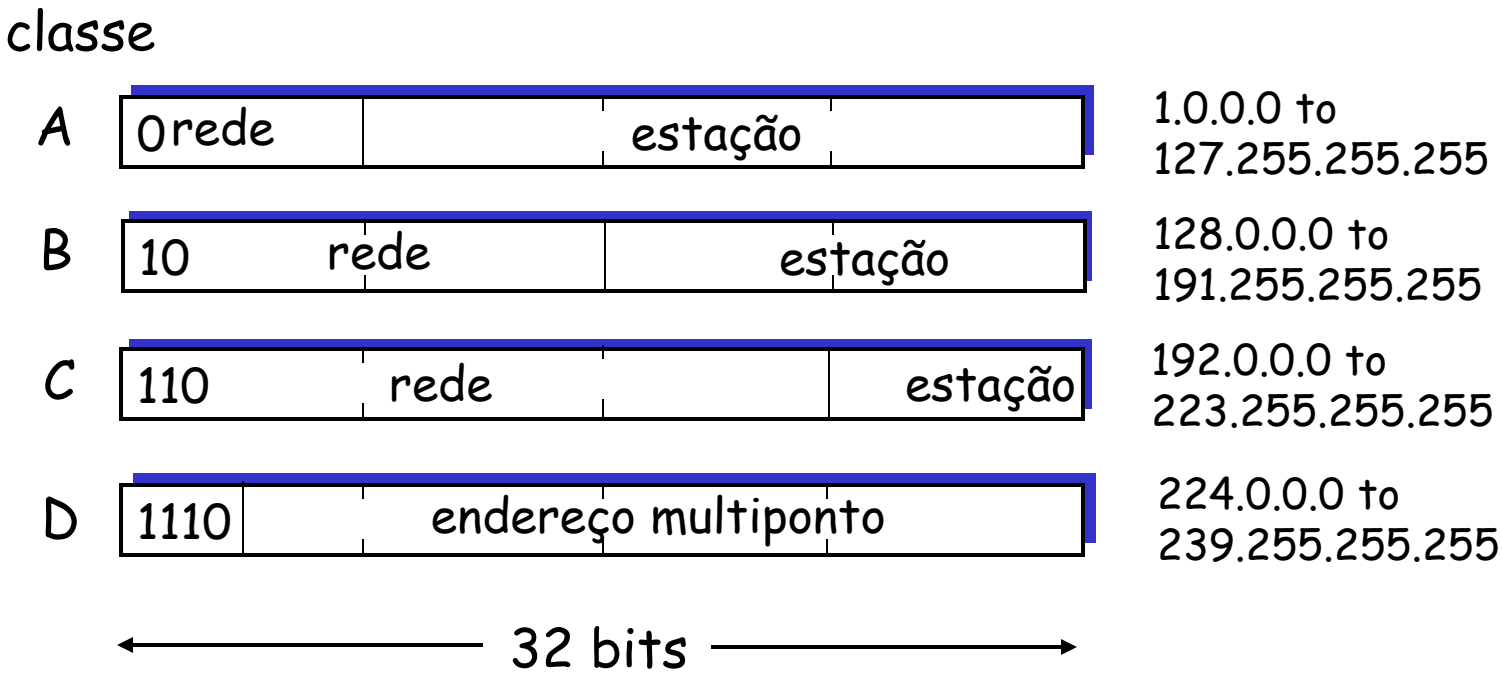
rede dest.	próx. rot.	Nenlaces
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Endereços IP

dada a noção de "rede", vamos reexaminar endereços IP:

endereçamento "baseado em classes":



Endereçamento IP: CIDR

- Endereçamento baseado em classes:
  - uso ineficiente e esgotamento do espaço de endereços
  - p.ex., rede da classe B aloca endereços para 65K estações, mesmo se houver apenas 2K estações nessa rede
- CIDR: Classless InterDomain Routing
  - parte de rede do endereço de comprimento arbitrário
  - formato de endereço: a.b.c.d/x, onde x é no. de bits na parte de rede do endereço

200.23.16.0/23

Endereçamento IP: a última palavra...

P: Como um provedor IP consegue um bloco de endereços?

A: ICANN: Internet Corporation for Assigned Names and Numbers

- ✓ aloca endereços
- ✓ gerencia DNS
- ✓ aloca nomes de domínio, resolve disputas

(no Brasil, estas funções foram delegadas ao Comitê Gestor Internet BR)

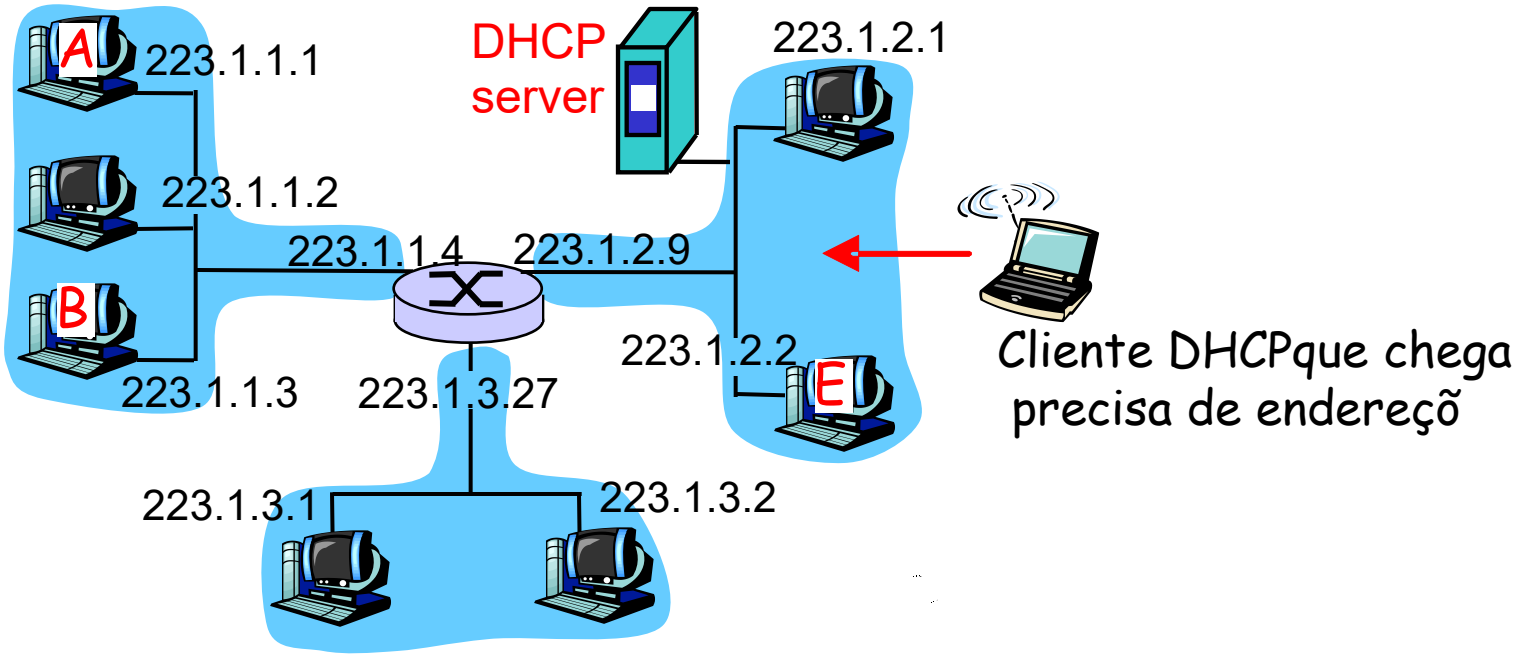
Endereços IP: como conseguir um?

Rede (parte de rede):

- conseguir alocação a partir do espaço de endereços do seu provedor IP

Bloco do provedor	11001000 00010111 00010000	00000000	200.23.16.0/20
Organização 0	11001000 00010111 00010000	00000000	200.23.16.0/23
Organização 1	11001000 00010111 00010010	00000000	200.23.18.0/23
Organização 2	11001000 00010111 00010100	00000000	200.23.20.0/23
...	.....	....	....
Organização 7	11001000 00010111 00011110	00000000	200.23.30.0/23

DHCP

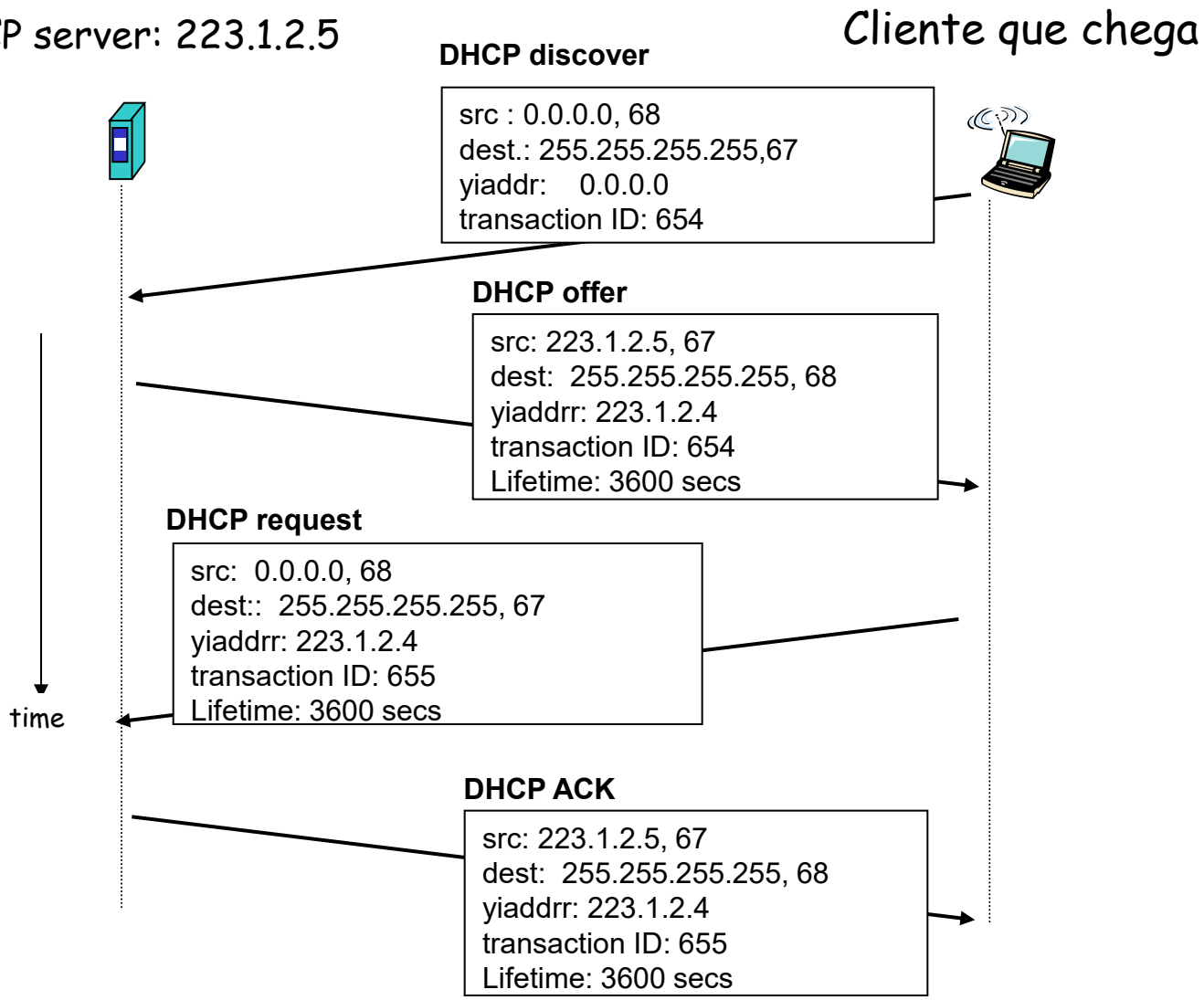

A
223.1.1.1
DHCP
server
223.1.2.1
223.1.1.2
223.1.1.4
223.1.2.9
B
223.1.2.2
E
223.1.1.3
223.1.3.27
Cliente DHCPque chega
precisa de endereçõ
223.1.3.1
223.1.3.2

Endereços IP: como conseguir um?

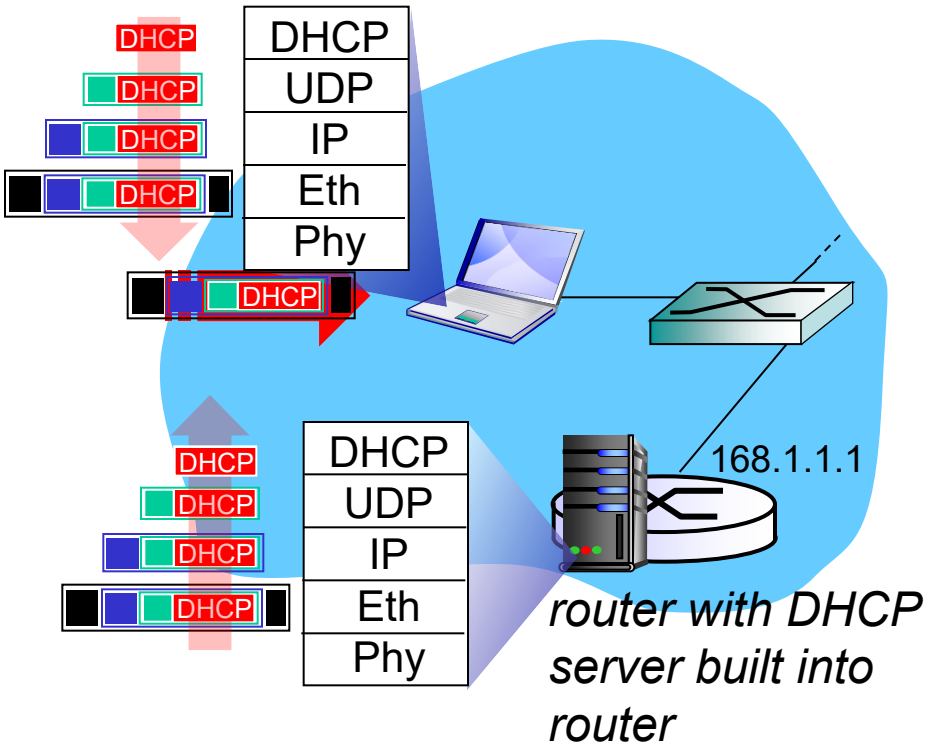
Estações (parte de estação):

- codificado pelo administrador num arquivo
  - ✓ Windows: control-panel->network->configuration->tcp/ip->properties
  - ✓ UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: obtém endereço dinamicamente: "plug-and-play"
  - ✓ estação difunde mensagem "DHCP discover"
  - ✓ servidor DHCP responde com "DHCP offer"
  - ✓ estação solicita endereço IP: "DHCP request"
  - ✓ servidor DHCP envia endereço: "DHCP ack"

DHCP client-server scenario

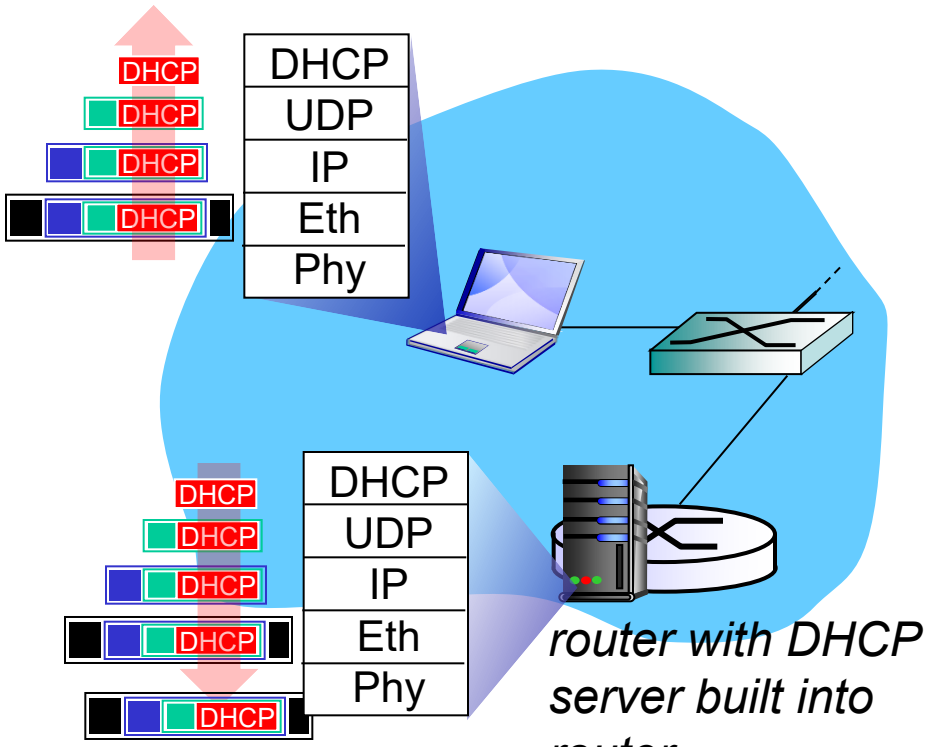


DHCP: example



- connecting laptop needs its IP address, addr of first-hop router, addr of DNS server: use DHCP
- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.1 Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

DHCP: example



- DCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulation of DHCP server, frame forwarded to client, demuxing up to DHCP at client
- client now knows its IP address, name and IP address of DSN server, IP address of its first-hop router

DHCP: Wireshark output (home LAN)

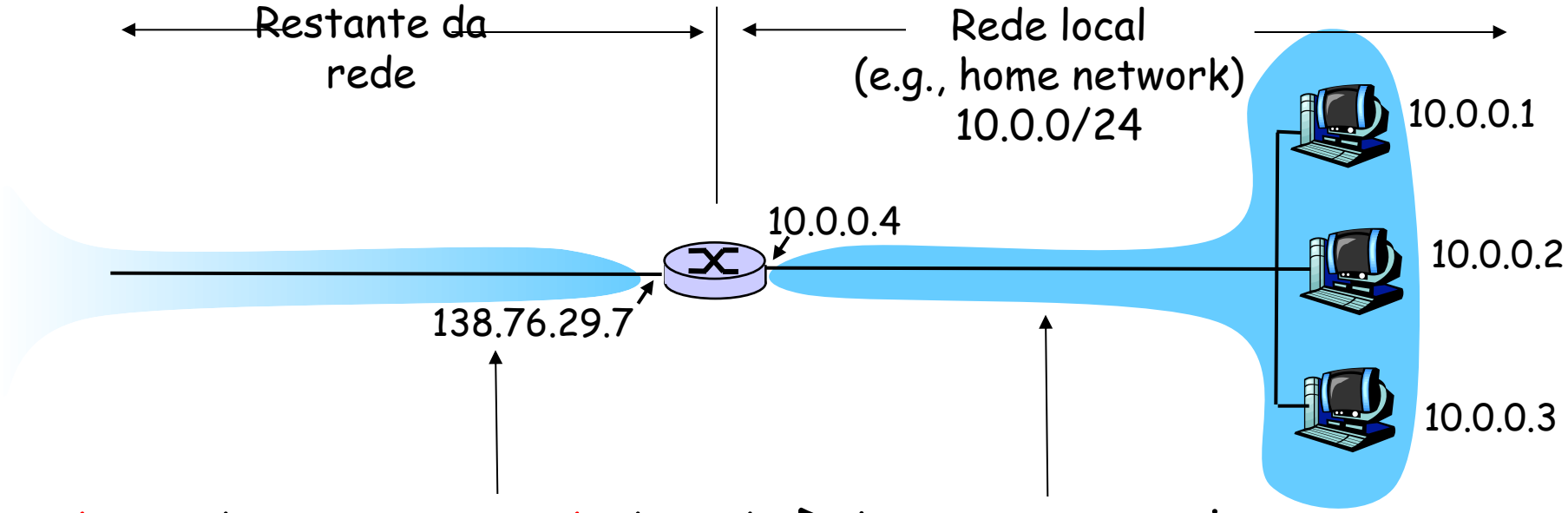
request

Message type: **Boot Request (1)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) **DHCP Message Type = DHCP Request**
Option: (61) Client identifier
    Length: 7; Value: 010016D323688A;
    Hardware type: Ethernet
    Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Option: (t=50,l=4) Requested IP Address = 192.168.1.101
Option: (t=12,l=5) Host Name = "nomad"
Option: (55) Parameter Request List
    Length: 11; Value: 010F03062C2E2F1F21F92B
    1 = Subnet Mask; 15 = Domain Name
    3 = Router; 6 = Domain Name Server
    44 = NetBIOS over TCP/IP Name Server
    ……

reply

Message type: **Boot Reply (2)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 192.168.1.101 (192.168.1.101)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 192.168.1.1 (192.168.1.1)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) DHCP Message Type = DHCP ACK
Option: (t=54,l=4) Server Identifier = 192.168.1.1
Option: (t=1,l=4) Subnet Mask = 255.255.255.0
Option: (t=3,l=4) Router = 192.168.1.1
Option: (6) Domain Name Server
    Length: 12; Value: 445747E2445749F244574092;
    IP Address: 68.87.71.226;
    IP Address: 68.87.73.242;
    IP Address: 68.87.64.146
Option: (t=15,l=20) Domain Name = "hsd1.ma.comcast.net."

NAT: Network Address Translation



Todos os datagramas *saindo* da rede local tem o mesmo endereço NAT IP: 138.76.29.7, diferentes números de portas fontes

Datagramas com origem ou destino nesta rede tem endereço 10.0.0/24 para fonte, e de destino o usual

NAT: Network Address Translation

- Motivação: rede local usa apenas um endereço IP:
  - ✓ Não há necessidade de alocar faixas de endereços de um ISP
    - apenas um endereço IP é usado para todos os dispositivos
  - ✓ Permite mudar o endereço dos dispositivos internos sem necessitar notificar o mundo externo;
  - ✓ Permite a mudança de ISPs sem necessitar mudar os endereços dos dispositivos internos da rede local
  - ✓ Dispositivos internos a rede, não são visíveis nem endereçaveis pelo mundo externo (melhora segurança);

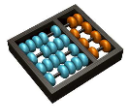
NAT: Network Address Translation

Implementação: roteador NAT deve;

- ✓ *Datagramas que saem: trocar* (endereço IP fonte, porta #) de cada datagrama de saída para (endereço NAT IP, nova porta #)

  . . . clientes/servidores remotos irão responder usando (endereço NAT IP, nova porta #) como endereço destino.

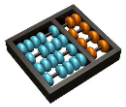
- ✓ *guardar (na tabela de tradução de endereços NAT): os pares de tradução de endereços* (endereço IP fonte, porta #) para (endereços NAT IP, nova porta #)
- ✓ *Datagramas qeu chegam: trocar* (endereço NAT IP, nova porta #) no campo de destino de cada datagrama que chega com o correspondente (endereço IP fonte, porta #) armazenado na tabela NAT



NAT: network address translation

- all devices in local network have 32-bit addresses in a “private” IP address space (10/8, 172.16/12, 192.168/16 prefixes) that can only be used in local network
- advantages:
  - just one IP address needed from provider ISP for *all* devices
  - can change addresses of host in local network without notifying outside world
  - can change ISP without changing addresses of devices in local network
  - security: devices inside local net not directly addressable, visible by outside world

UNICAMP

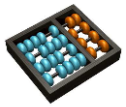


NAT: network address translation

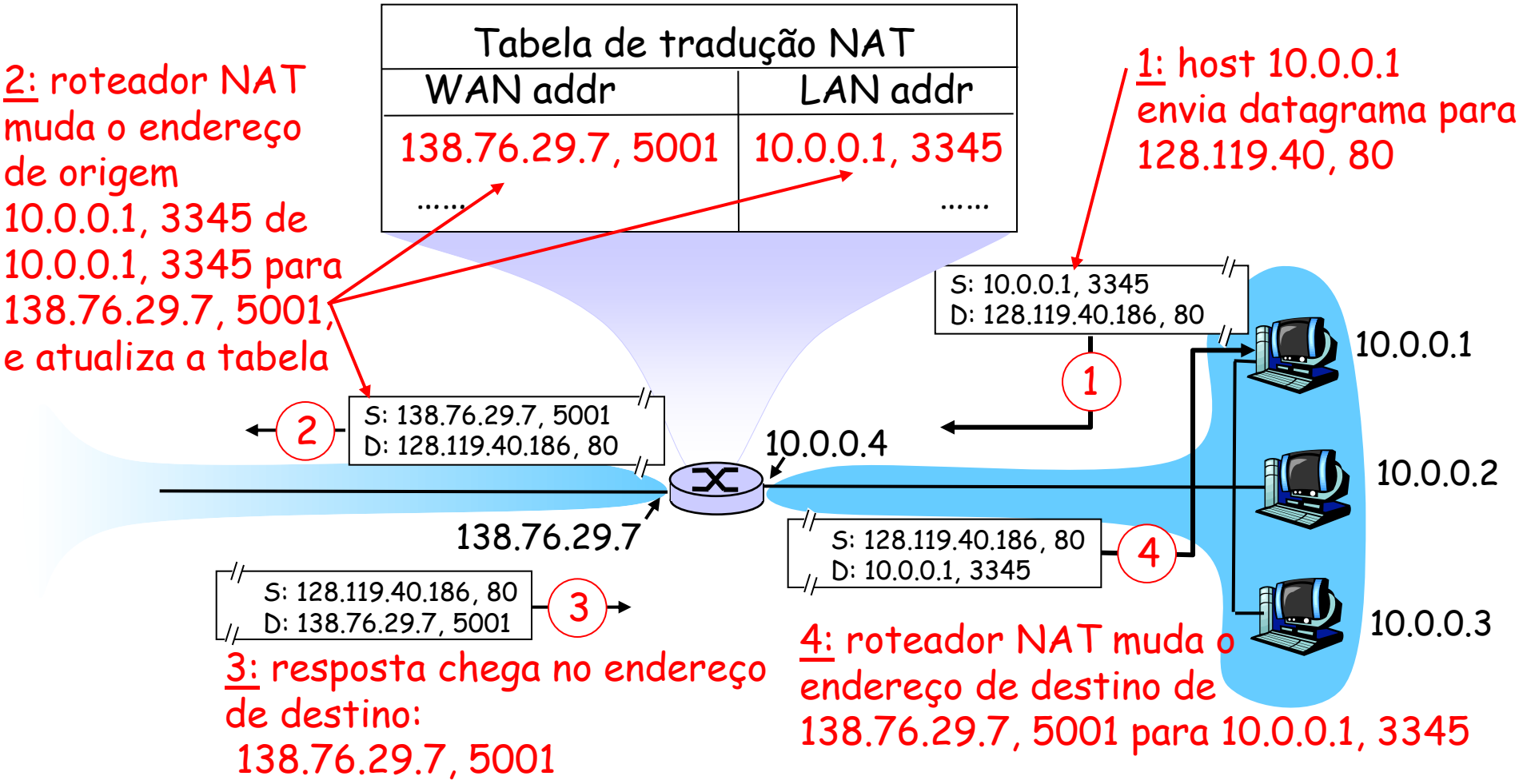
implementation: NAT router must (transparently):

- outgoing datagrams: replace (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
  - remote clients/servers will respond using (NAT IP address, new port #) as destination address
- remember (in NAT translation table) every (source IP address, port #) to (NAT IP address, new port #) translation pair
- incoming datagrams: replace (NAT IP address, new port #) in destination fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

UNICAMP



NAT: Network Address Translation



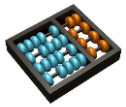
NAT: Network Address Translation

- Campo de porta de 16-bit :
  - ✓ 60,000 conexões simultâneas com um único endereço de rede;
- NAT é controverso:
  - ✓ Roteadores devem fazer processamentos até no máximo a camada 3;
  - ✓ Viola o "conceito fim-a-fim"
    - A possibilidade de suporte a NAT deve ser levado em consideração pelos desenvolvedores de aplicações;
  - ✓ O problema de diminuição do número de endereços deveria ser tratada por IPv6;

NAT: network address translation

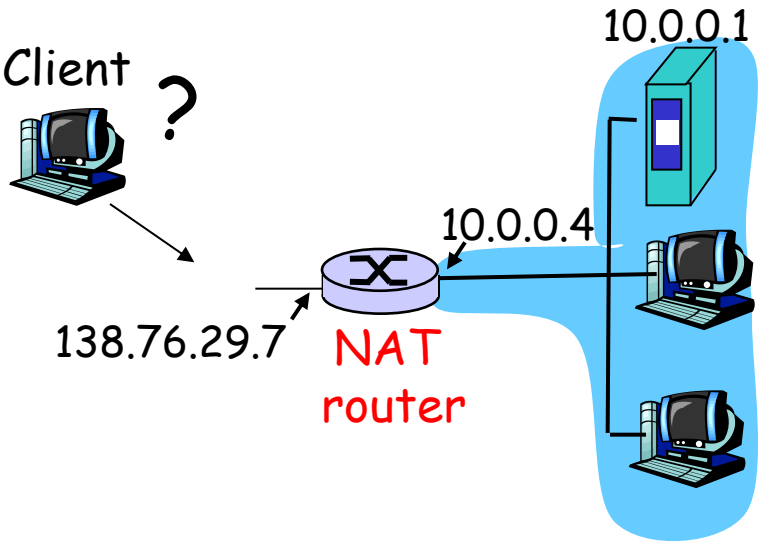
- NAT has been controversial:
  - routers “should” only process up to layer 3
  - address “shortage” should be solved by IPv6
  - violates end-to-end argument (port # manipulation by network-layer device)
  - NAT traversal: what if client wants to connect to server behind NAT?
- but NAT is here to stay:
  - extensively used in home and institutional nets, 4G/5G cellular nets

UNICAMP



Problema NAT traversal

- Cliente deseja conectar-se ao servidor com endereço 10.0.0.1
  - ✓ Endereço servidor 10.0.0.1 na rede local (cliente não pode usar como endereço destino)
  - ✓ Somente um endereço (NATed) externamente visível: 138.76.29.7
- Uma solução: configurar NAT manualmente para encaminhar requisição de conexão a uma certa porta do servidor
- exemplo., (123.76.29.7, port 2500) sempre envia para 10.0.0.1 port 25000



Roteiro

- 4. 1 Introdução
- 4.2 Circuitos virtuais x datagrama
- 4.3 Como é um roteador
- 4.4 Protocolo IP
  - Formato datagarma
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - Estado de enlace
  - Vetor distância
  - Roteamento hierarquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento Broadcast e multicast

ICMP: Internet Control Message Protocol

- usado por estações, roteadores para comunicar informação s/ camada de rede
  - ✓ relatar erros: estação, rede, porta, protocolo inalcançáveis
  - ✓ pedido/resposta de eco (usado por ping)
- camada de rede "acima de" IP:
  - ✓ msgs ICMP transportadas em datagramas IP
- mensagem ICMP: tipo, código mais primeiros 8 bytes do datagrama IP causando erro

Tipo	Código	descrição
0	0	resposta de eco (ping)
3	0	rede dest. inalcançável
3	1	estação dest inalcançável
3	2	protocolo dest inalcançável
3	3	porta dest inalcançável
3	6	rede dest desconhecida
3	7	estação dest desconhecida
4	0	abaixar fonte (controle de congestionamento - ñ usado)
8	0	pedido eco (ping)
9	0	anúncio de rota
10	0	descobrir roteador
11	0	TTL (sobrevida) expirada
12	0	erro de cabeçalho IP

ICMP eTraceroute

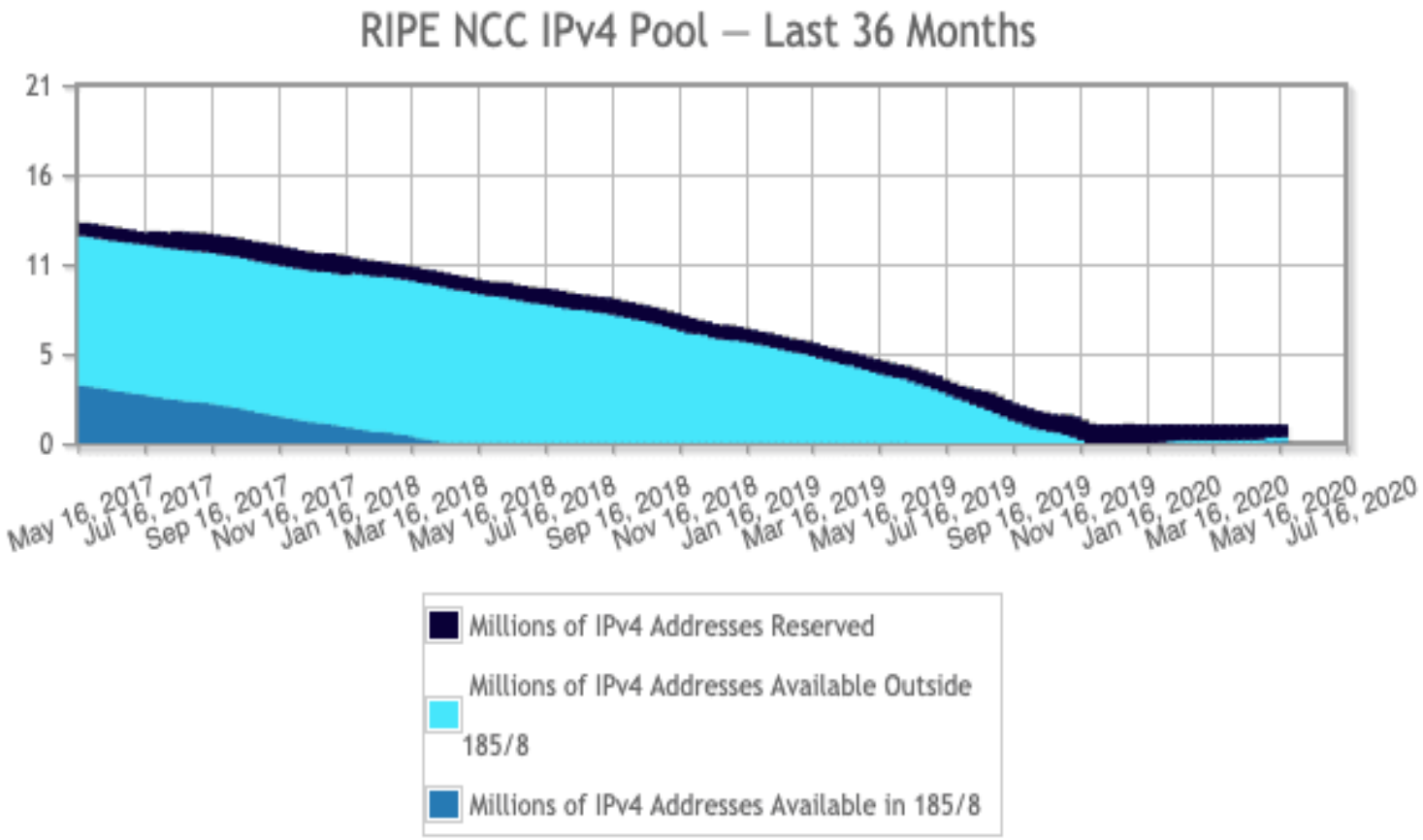
- Fonte UDP envia uma série de segmentos
  - ✓ Primeiro TTL = 1
  - ✓ Segundo TTL=2, etc.
- Quando n-ésimo datagrama chega no nésimo roteador :
  - ✓ Roteador descarta datagrama
  - ✓ Envia para a fonte um datagrama ICMP (type 11, code 0)
  - ✓ Mensagem inclui nome de roteador e endereço IP
- Quando mensagem ICMP chega, finte calcula RTT
- Traceroute faz isso três vezes

Critério de parada

- Segmento UDP eventualmente chega ao destinatário
- Destinatário ICMP retorna mensagem "host unreachable" (type 3, code 3)
- Quando fonte recebe mensagem, ICMP para.

Roteiro

- 4. 1 Introdução
- 4.2 Circuitos virtuais x datagrama
- 4.3 Como é um roteador
- 4.4 Protocolo IP
  - Formato datagarma
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - Estado de enlace
  - Vetor distância
  - Roteamento hierarquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento Broadcast e multicast

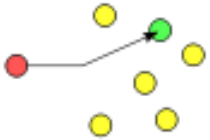

RIPE NCC IPv4 Pool — Last 36 Months
21
16
11
5
0
May 16, 2017
Jul 16, 2017
Sep 16, 2017
Nov 16, 2017
Jan 16, 2018
Mar 16, 2018
May 16, 2018
Jul 16, 2018
Sep 16, 2018
Nov 16, 2018
Jan 16, 2019
Mar 16, 2019
May 16, 2019
Jul 16, 2019
Sep 16, 2019
Nov 16, 2019
Jan 16, 2020
Mar 16, 2020
May 16, 2020
Jul 16, 2020
Millions of IPv4 Addresses Reserved
Millions of IPv4 Addresses Available Outside 185/8
Millions of IPv4 Addresses Available in 185/8

IPv6

- Motivação inicial: espaço de endereços de 32-bits completamente alocado até 2008.
- Motivação adicional :
  - ✓ formato do cabeçalho facilita acelerar processamento/re-encaminhamento
  - ✓ mudanças no cabeçalho para facilitar QoS
  - ✓ novo endereço "anycast": rota para o "melhor" de vários servidores replicados
- formato do datagrama IPv6:
  - ✓ cabeçalho de tamanho fixo de 40 bytes
  - ✓ não admite fragmentação

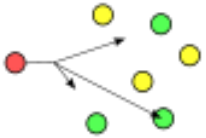
Anycast

- Unicast

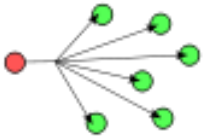


- Multicast

- Anycast

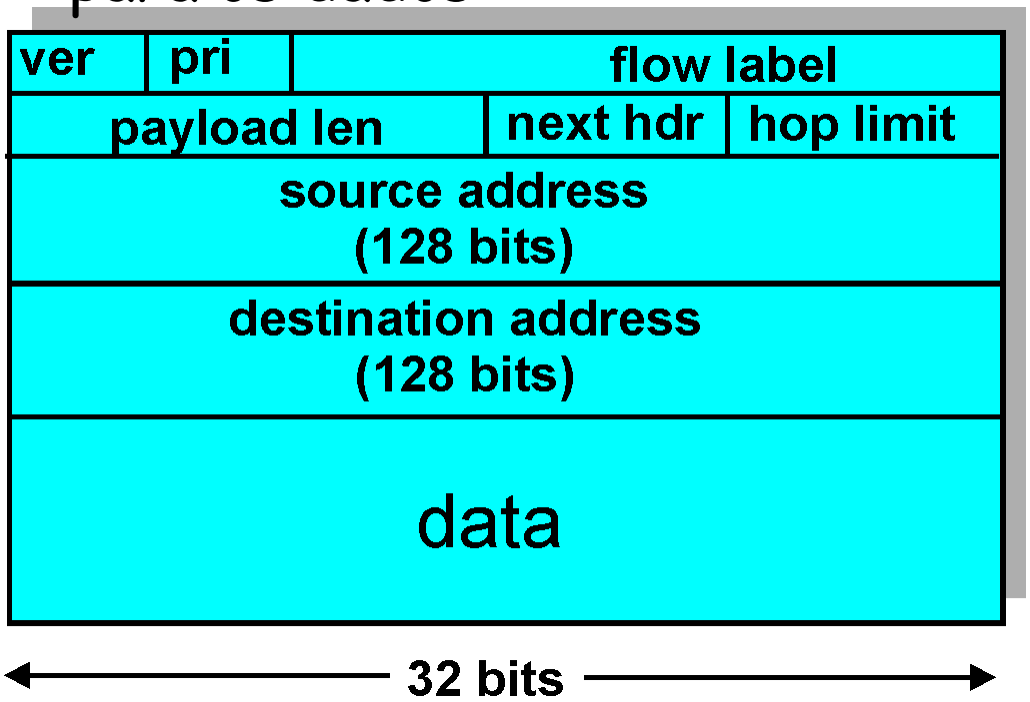


- Broadcast



Cabeçalho IPv6

Prioridade: identifica prioridade entre datagramas no fluxo
Rótulo do Fluxo: identifica datagramas no mesmo "fluxo" (conceito de "fluxo" mal definido).
Próximo cabeçalho: identifica protocolo da camada superior para os dados

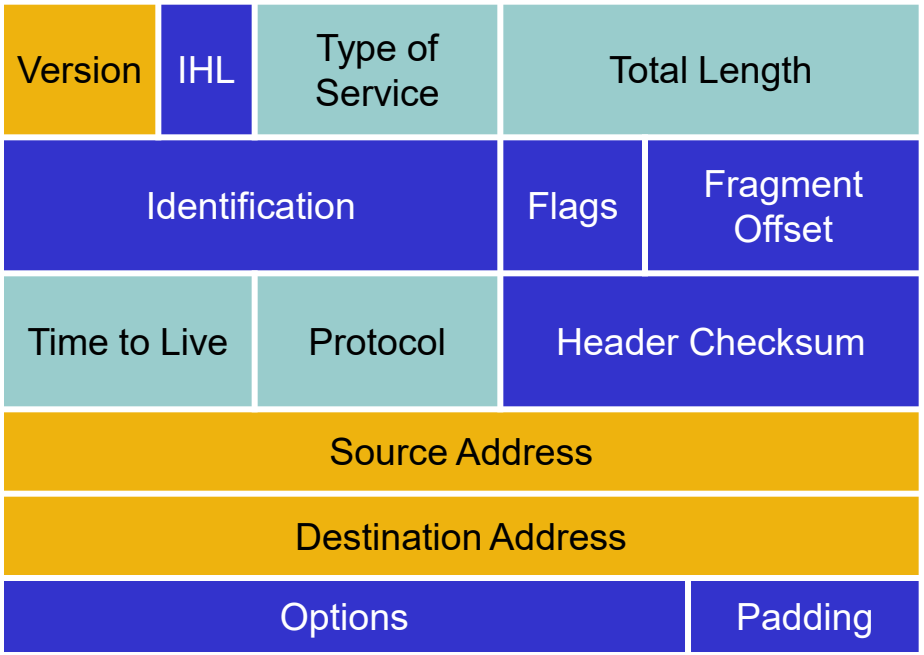


Outras mudanças de IPv4

- *Checksum:* removido completamente para reduzir tempo de processamento a cada roteador
- *Opções:* permitidas, porém fora do cabeçalho, indicadas pelo campo "Próximo Cabeçalho"
- *ICMPv6:* versão nova de ICMP
  - ✓ tipos adicionais de mensagens, p.ex. "Pacote Muito Grande"
  - ✓ funções de gerenciamento de grupo multiponto

Comparação IPv4 and IPv6

IPv4 Header

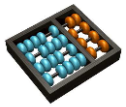


IPv6 Header

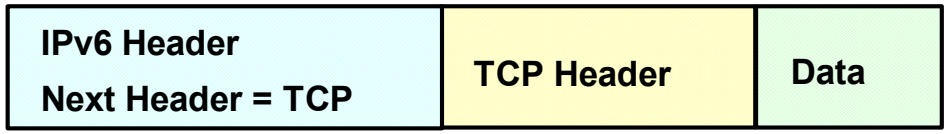
Legend

- Field's name kept from IPv4 to IPv6
- Fields not kept in IPv6
- Name and position changed in IPv6
- New field in IPv6

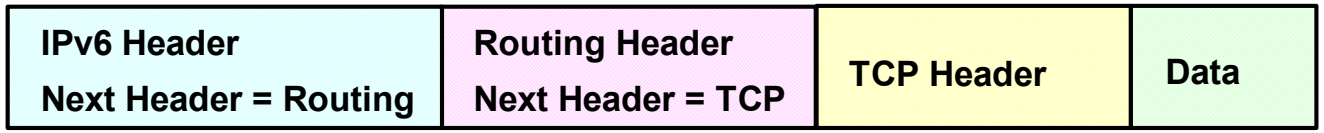
UNICAMP



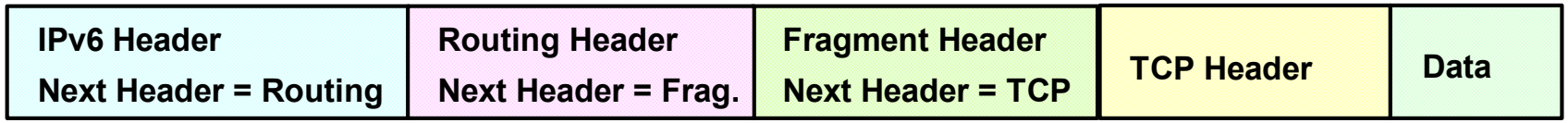
Extensão do cabeçalho - Opções em IPv6



(a) No extension header

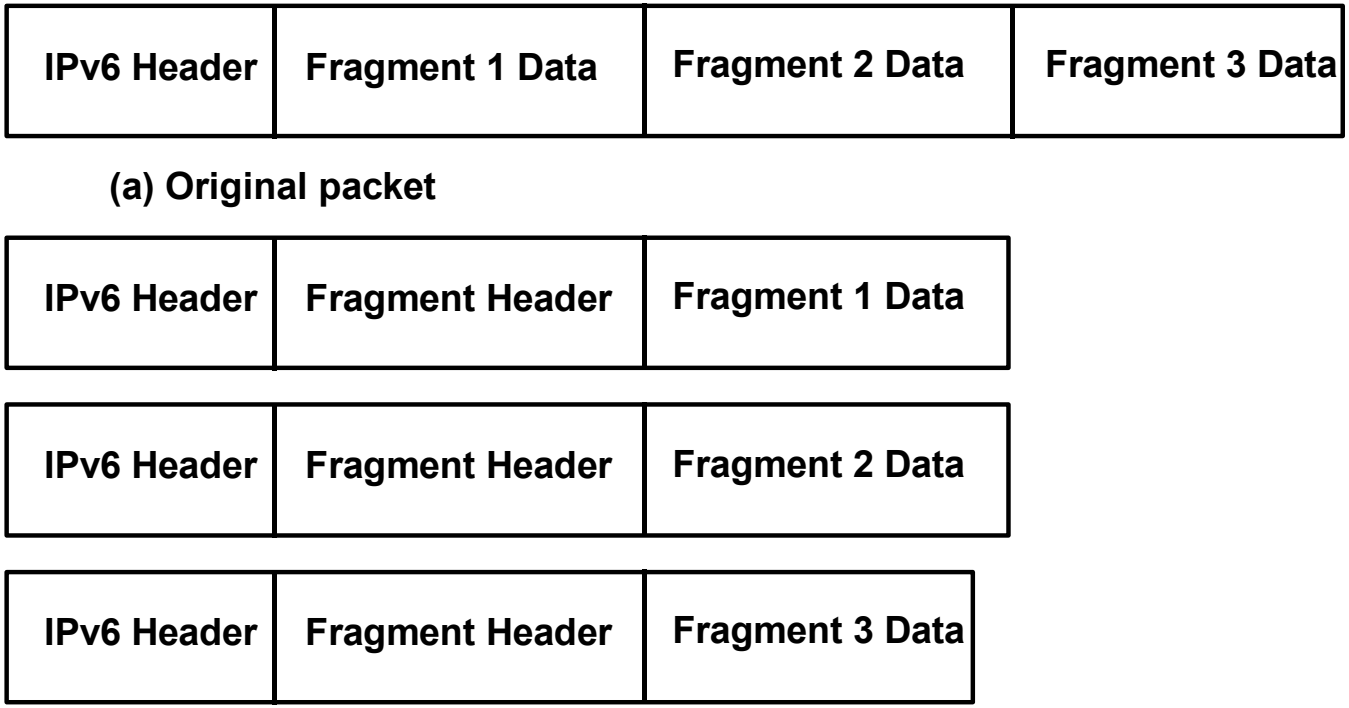


(b) IPv6 header followed by a routing header



(c) IPv6 header followed by a routing header and a fragment header

Exemplo de Fragmentação



Endereço IPv6

- 128 bits
- Notação hexadecimal separada por ":"

3FFD:3600:0000:0000:0302:B3FF:FE3C: C0DB

- Sequência de números de de 16 bits nulos separados por "::"

3FFD:3600:0:0:0:0:1:A =>3FFD:3600::1:A

Endereço IPv6

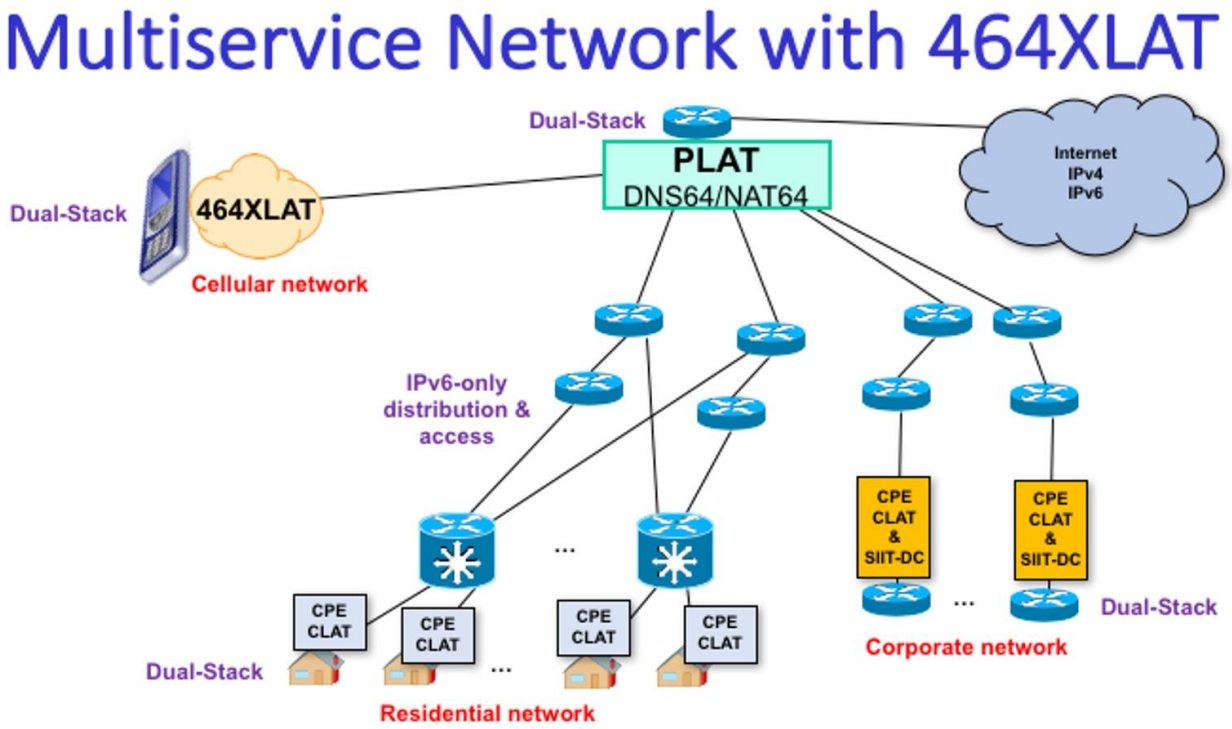
ICMPv6

- Versão do protocolo ICMP utilizada pelo protocolo Ipv6, basicamente as mesmas funcionalidades do ICMP
  - ✓ Realizar diagnósticos
  - ✓ Relatar erros de processamento de pacotes
- Possui também funções adicionais
  - ✓ *Descoberta de vizinhança (antes providas pelo protocolo ARP)*
  - ✓ *Gerenciamento de grupos Multicast (antes providas pelo protocolo IGMP).*

IPv6 DHCPv6

- Semelhante ao DHCP
- Opera em modo cliente – servidor
- Obtém endereço IPv6 e informações de configurações e segurança

IPV6 para usuário residencial

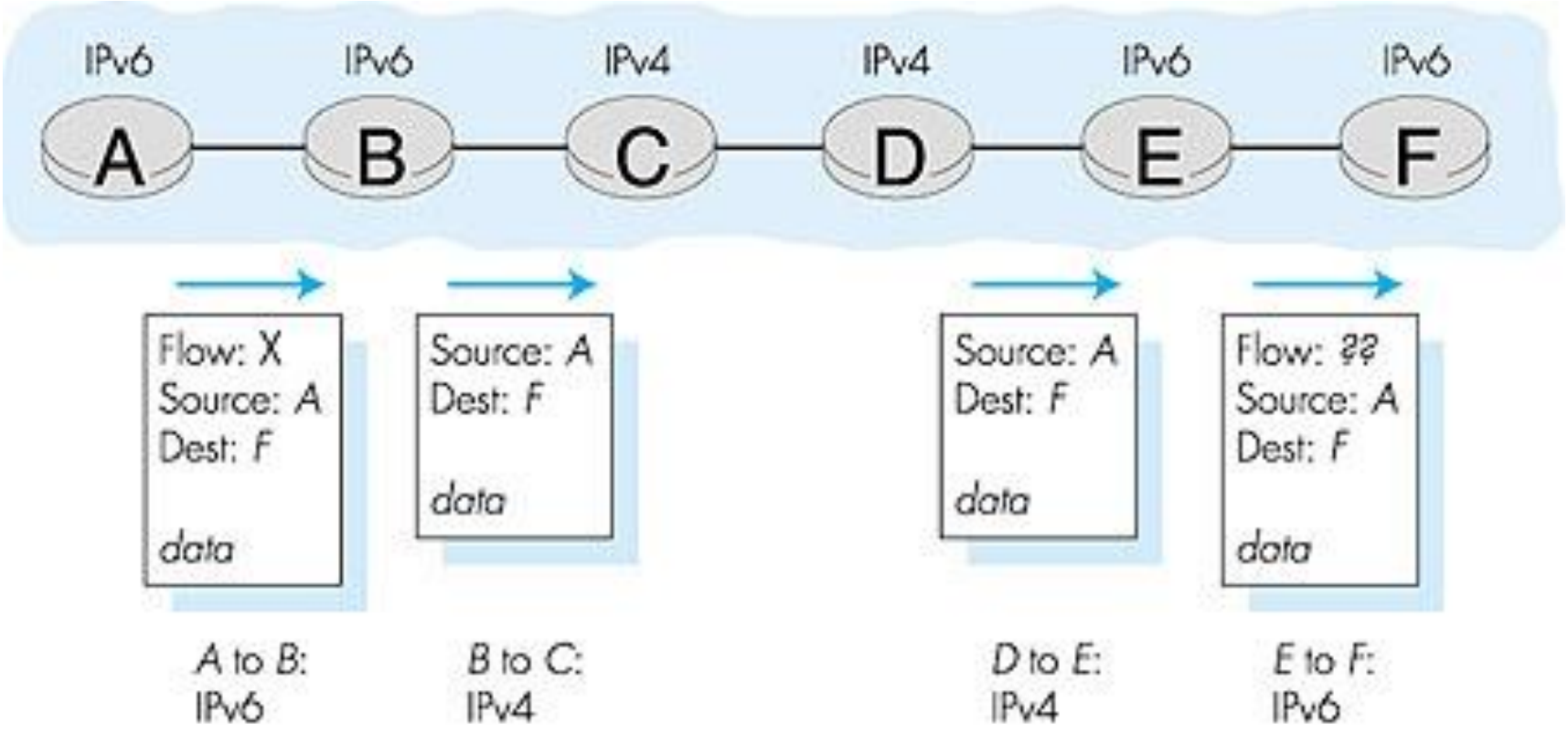


UNICAMP

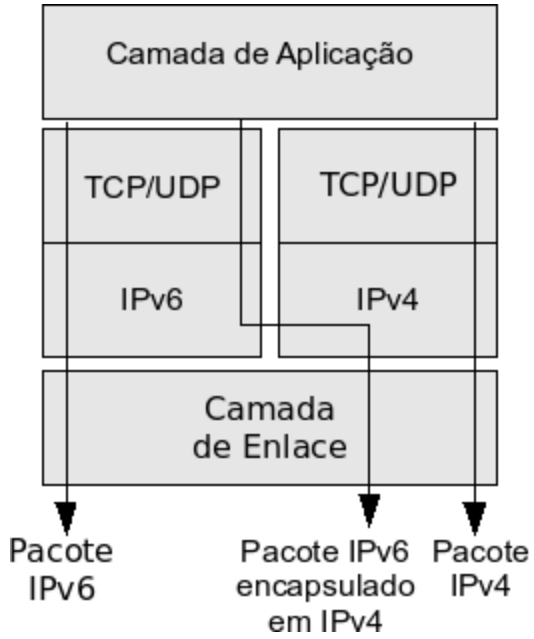
Transição de IPv4 para IPv6

- Nem todos roteadores podem ser atualizados simultaneamente
  - ✓ "dias de mudança geral" inviáveis
  - ✓ Como a rede pode funcionar com uma mistura de roteadores IPv4 e IPv6?
- Três abordagens propostas:
  - ✓ *Pilhas Duais*: alguns roteadores com duas pilhas (v6, v4) podem "traduzir" entre formatos
  - ✓ *Tunelamento*: datagramas IPv6 carregados em datagramas IPv4 entre roteadores IPv4
  - ✓ Tradutores de protocolos

Abordagem de Pilhas Duais



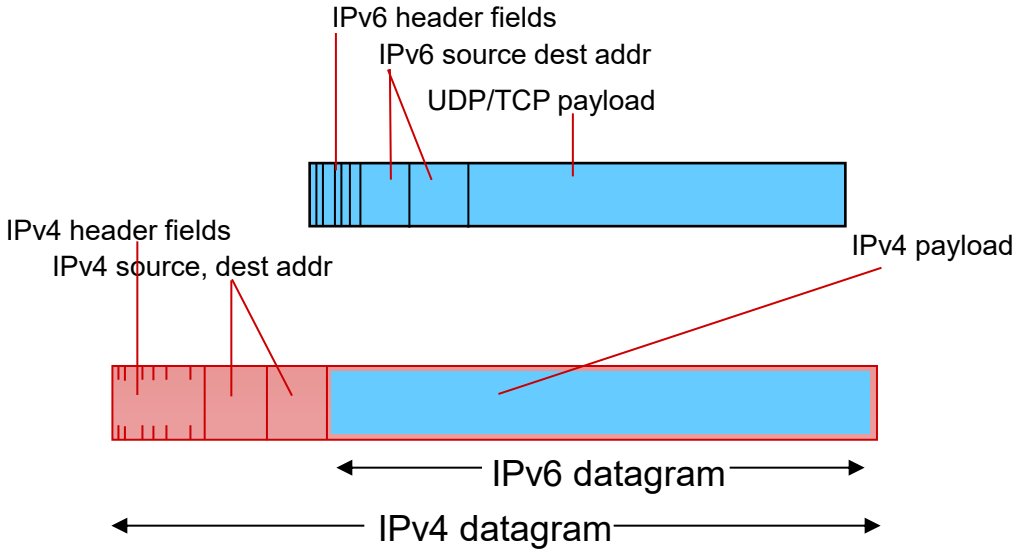
Pilhas Duais



Transition from IPv4 to IPv6

- not all routers can be upgraded simultaneously
  - ✓ no "flag days"
  - ✓ How will the network operate with mixed IPv4 and IPv6 routers?

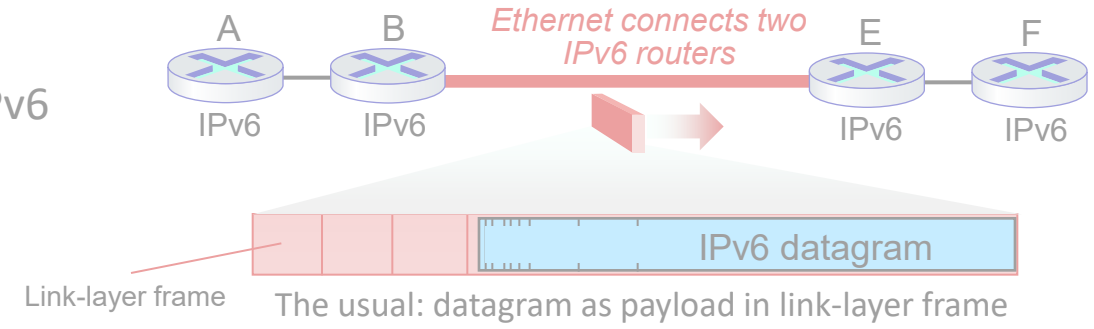
- tunneling: IPv6 datagram carried as *payload* in IPv4 datagram among IPv4 routers ("packet within a packet")
  - tunneling used extensively in other contexts (4G/5G)



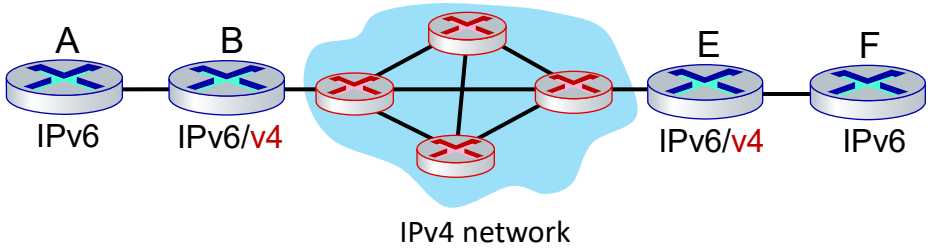
UNICAMP

Tunneling and encapsulation

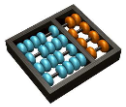
Ethernet connecting two IPv6 routers:



IPv4 network connecting two IPv6 routers

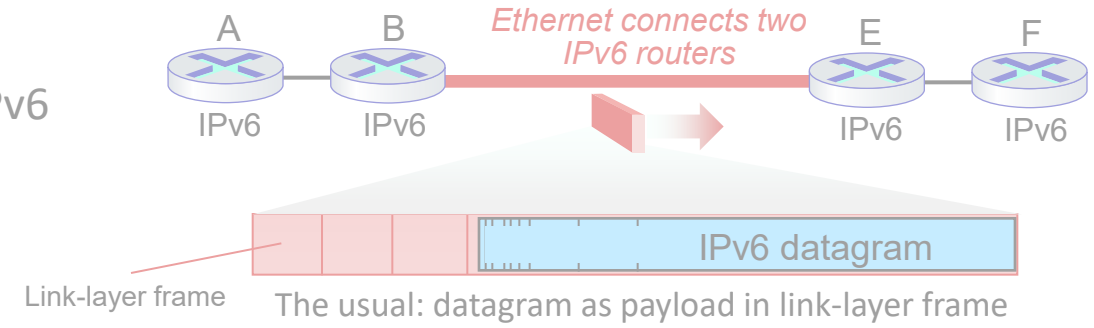


UNICAMP

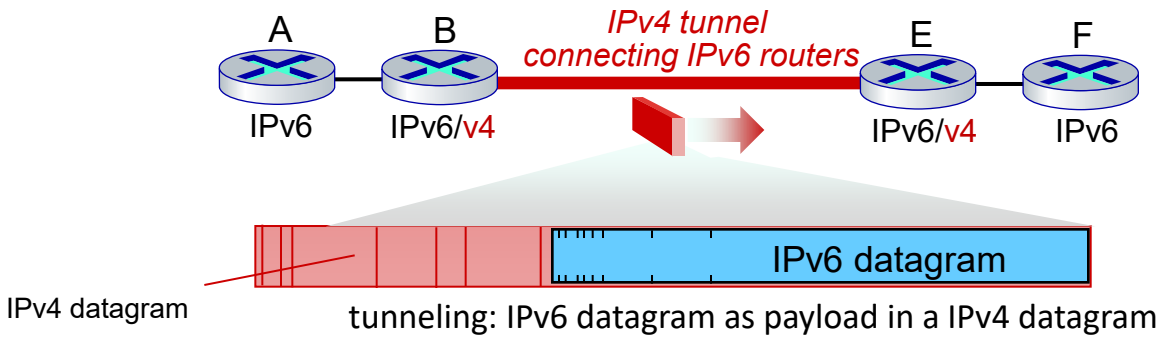


Tunneling and encapsulation

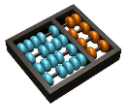
Ethernet connecting two IPv6 routers:



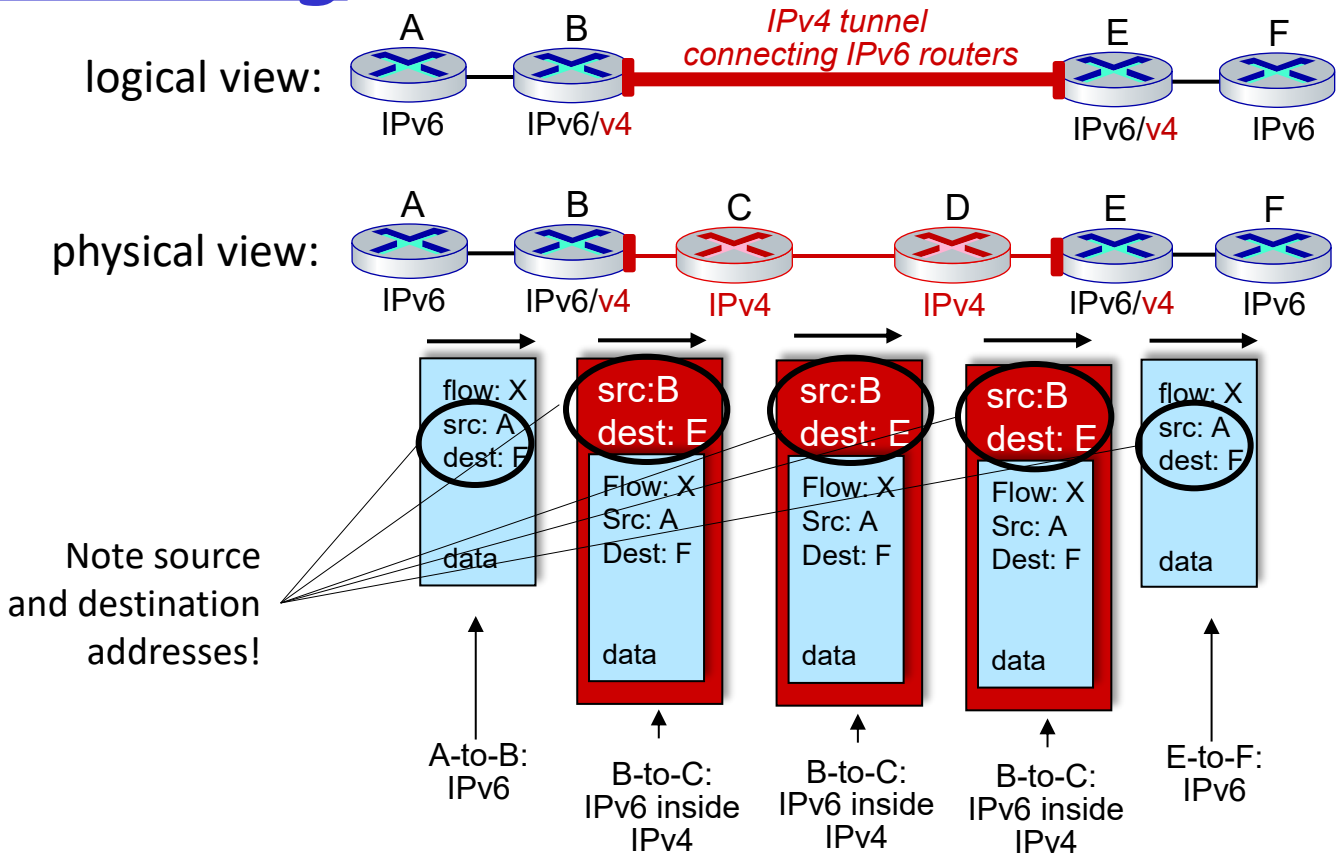
IPv4 tunnel connecting two IPv6 routers



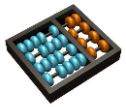
UNICAMP



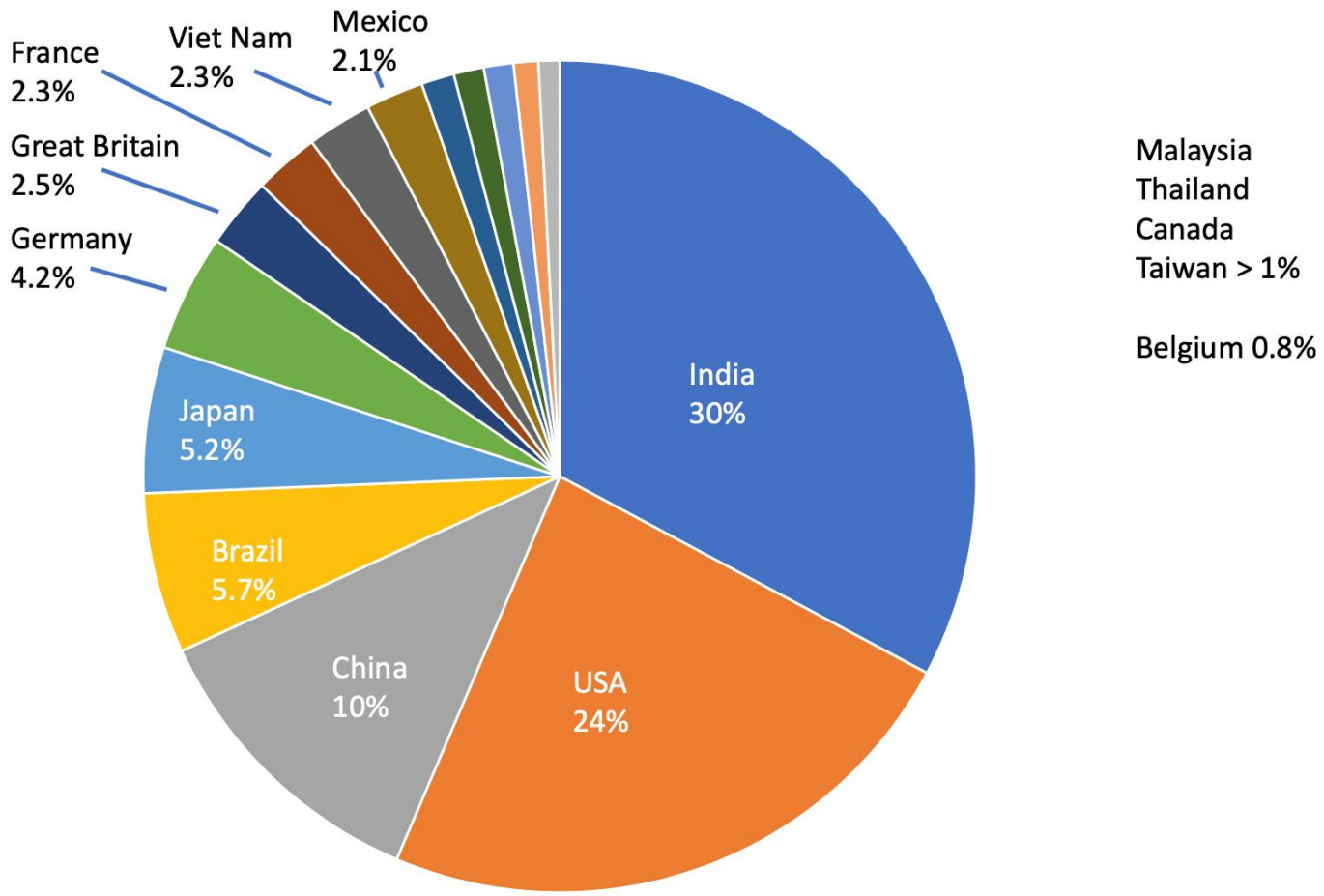
Tunneling



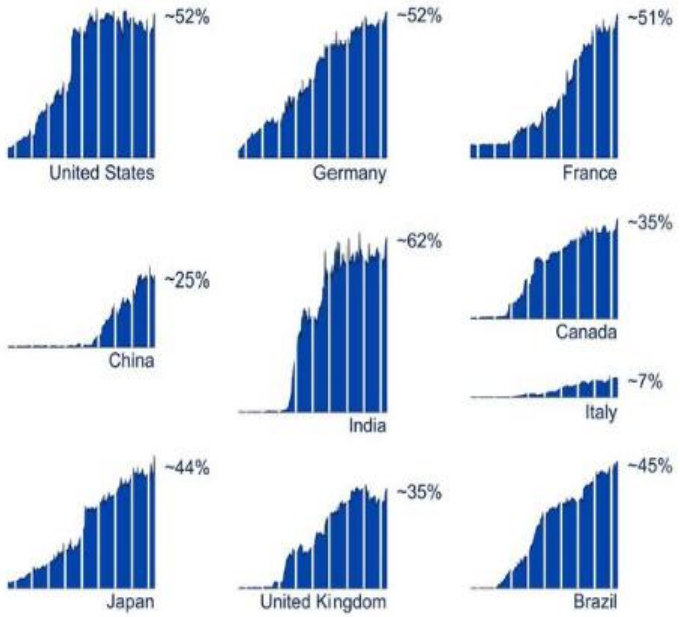
UNICAMP



Estatística IPv6



In those past years, we've seen an 1000x increase in IPv6 traffic, with an IPv6 peak over 41 Tbps. We observe end-user adoption levels in the approximately 44% to 62% range for 6 of the top 10 global economies, as measured by looking at the percentage of requests to a subset of IPv4+IPv6 dual-stacked content on the Akamai CDN.

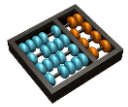


Percentage of requests over IPv6 to a subset of dual-stack sites on Akamai from July 2013 to May 2022 for top 10 global economies (by GDP in 2022, per IMF).

	World IPv6 Launch (2012)	10 years later (2022)
Peak IPv6 traffic	~1 Gbps	41 Tbps *(> 41,000 Gbps)*
Daily IPv6 requests	3.9 billion per day *(and 8 million in 2011)*	> 4,000 billion per day
IPv6 addresses observed per day	19 million	7.5 billion *(across 2.2 billion "/64" prefixes)*

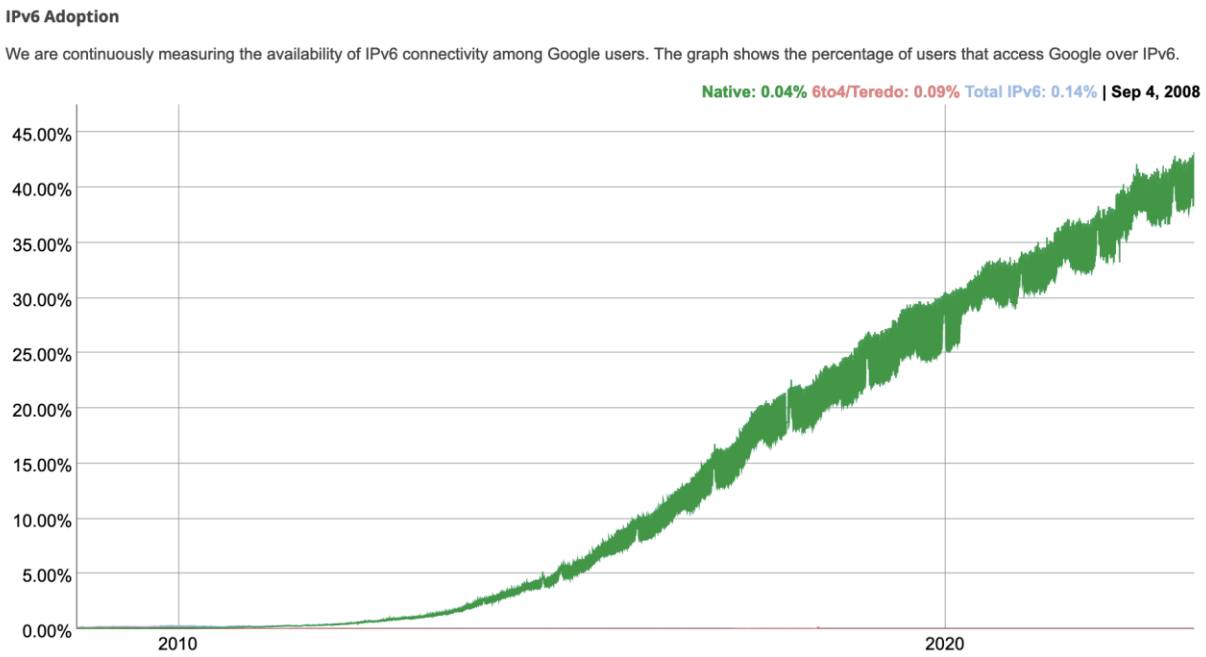
Fig. 1: How IPv6 traffic on the Akamai CDN has grown in the decade since World IPv6 Launch

UNICAMP

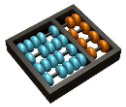


IPv6: adoption

- Google[1]: ~ 40% of clients access services via IPv6 (2023)
- NIST: 1/3 of all US government domains are IPv6 capable



UNICAMP

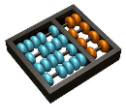


IPv6: adoption

- Google[1]: ~ 40% of clients access services via IPv6 (2023)
- NIST: 1/3 of all US government domains are IPv6 capable
- Long (long!) time for deployment, use
  - ✓ 25 years and counting!
  - ✓ think of application-level changes in last 25 years: WWW, social media, streaming media, gaming, telepresence, …
  - ✓ *Why?*

[1] https://www.google.com/intl/en/ipv6/statistics.html

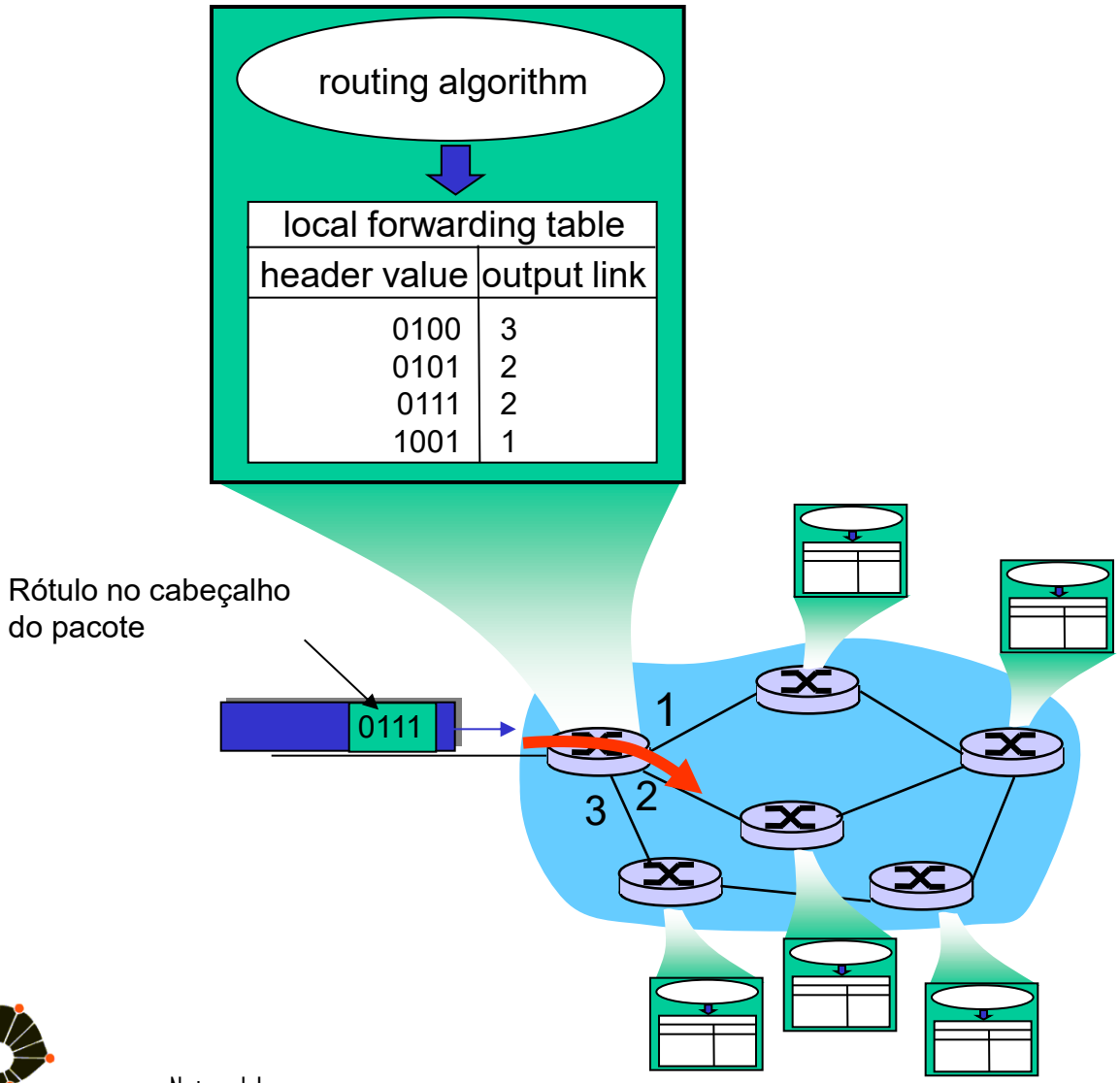
UNICAMP



Roteiro

- 4. 1 Introdução
- 4.2 Circuitos virtuais x datagrama
- 4.3 Como é um roteador
- 4.4 Protocolo IP
  - ✓ Formato datagarma
  - ✓ endereçamento IPv4
  - ✓ ICMP
  - ✓ IPv6
- 4.5 Algoritmos de roteamento
  - ✓ Estado de enlace
  - ✓ Vetor distância
  - ✓ Roteamento hierarquico
- 4.6 Roteamento na Internet
  - ✓ RIP
  - ✓ OSPF
  - ✓ BGP
- 4.7 Roteamento Broadcast e multicast

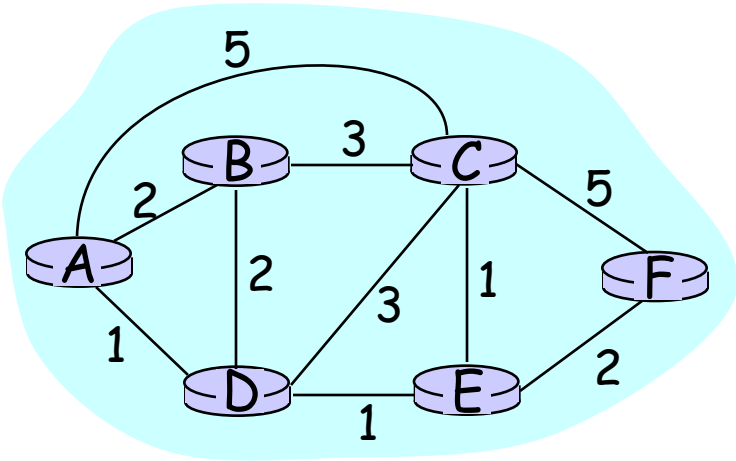
Roteamento


routing algorithm
local forwarding table
header value
output link
0100
3
0101
2
0111
2
1001
1
Rótulo no cabeçalho
do pacote
0111
1
2
3

Roteamento

protocolo de roteamento

meta: determinar caminho (seqüência de roteadores) "bom" pela rede da origem ao destino



Abstração de grafo para algoritmos de roteamento:

- nós do grafo são roteadores
- arestas do grafo são os enlaces físicos
  - custo do enlace: retardo, financeiro, ou nível de congestionamento

- caminho "bom":
  - tipicamente significa caminho de menor custo
  - outras definições são possíveis

Classificação de Algoritmos de Roteamento

Informação global ou descentralizada?

Global:

- todos roteadores têm info. completa de topologia, custos dos enlaces
- algoritmos "estado de enlaces"

Decentralizada:

- roteador conhece vizinhos diretos e custos até eles
- processo iterativo de cálculo, troca de info. com vizinhos
- algoritmos "vetor de distâncias"

Estático ou dinâmico?

Estático:

- rotas mudam lentamente com o tempo

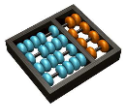
Dinâmico:

- rotas mudam mais rapidamente
  - ✓ atualização periódica
  - ✓ em resposta a mudanças nos custos dos enlaces

Intra-AS routing: routing within an AS

most common intra-AS routing protocols:

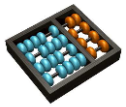
- RIP: Routing Information Protocol [RFC 1723]
  - classic DV: DVs exchanged every 30 secs
  - no longer widely used
- EIGRP: Enhanced Interior Gateway Routing Protocol
  - DV based
  - formerly Cisco-proprietary for decades (became open in 2013 [RFC 7868])
- OSPF: Open Shortest Path First [RFC 2328]
  - link-state routing
  - IS-IS protocol (ISO standard, not RFC standard) essentially same as OSPF



Software defined networking (SDN)

- Internet network layer: historically implemented via distributed, per-router control approach:
  - ✓ *monolithic* router contains switching hardware, runs proprietary implementation of Internet standard protocols (IP, RIP, IS-IS, OSPF, BGP) in proprietary router OS (e.g., Cisco IOS)
  - ✓ different "middleboxes" for different network layer functions: firewalls, load balancers, NAT boxes, ..
- ~2005: renewed interest in rethinking network control plane

UNICAMP



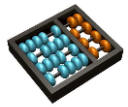
Um algoritmo de roteamento de "estado de enlaces" (EE)

Algoritmo de Dijkstra

- topologia da rede, custos dos enlaces conhecidos por todos os nós
  - ✓ realizado através de "difusão do estado dos enlaces"
  - ✓ todos os nós têm mesma info.
- calcula caminhos de menor custo de um nó ("origem") para todos os demais
  - ✓ gera tabela de rotas para aquele nó
- iterativo: depois de k iterações, sabemos menor custo p/ k destinos

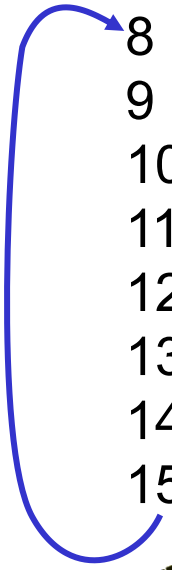
Notação:

- c(i,j): custo do enlace do nó i ao nó j. custo é infinito se não forem vizinhos diretos
- D(V): valor corrente do custo do caminho da origem ao destino V
- p(V): nó antecessor no caminho da origem ao nó V, imediatamente antes de V
- N: conjunto de nós cujo caminho de menor custo já foi determinado



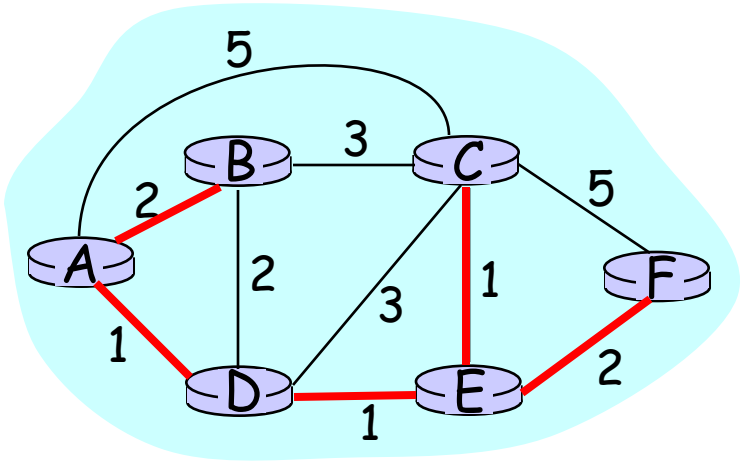
O algoritmo de Dijkstra

```
1  Inicialização:
2    N = {A}
3    para todos os nós V
4      se V for adjacente ao nó A
5        então D(V) = c(A,V)
6        senão D(V) = infinito
7
8   Repete
9     determina W não contido em N tal que D(W) é o mínimo
10    adiciona W ao conjunto N
11    atualiza D(V) para todo V adjacente ao nó W e ainda não em N:
12       D(V) = min( D(V), D(W) + c(W,V) )
13    /* novo custo ao nó V ou é o custo velho a V ou o custo do
14     menor caminho ao nó W, mais o custo de W a V */
15  até que todos nós estejam em N
```



Algoritmo de Dijkstra: exemplo

Passo	N inicial	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
→ 0	A	2,A	5,A	1,A	infinito	infinito
→ 1	AD	2,A	4,D		2,D	infinito
→ 2	ADE	2,A	3,E			4,E
→ 3	ADEB		3,E			4,E
→ 4	ADEBC					4,E
5	ADEBCF					



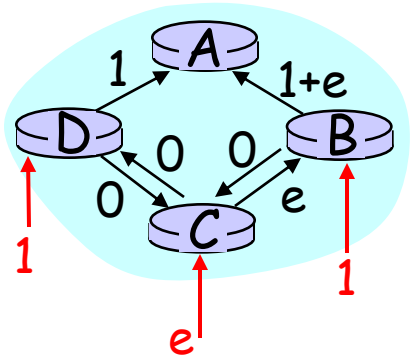
Algoritmo de Dijkstra, discussão

Complexidade algoritmica: n nós

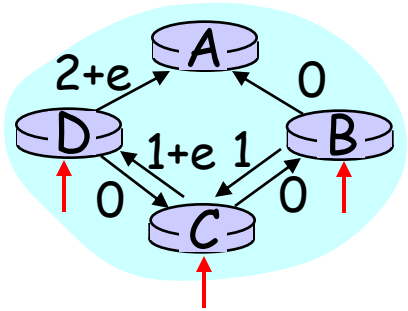
- a cada iteração: precisa checar todos nós, W, não em N
- n*(n+1)/2 comparações => O(n**2)
- implementações mais eficientes possíveis: O(nlogn)

Oscilações possíveis:

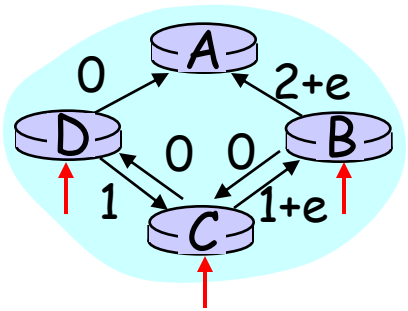
- p.ex., custo do enlace = carga do tráfego carregado



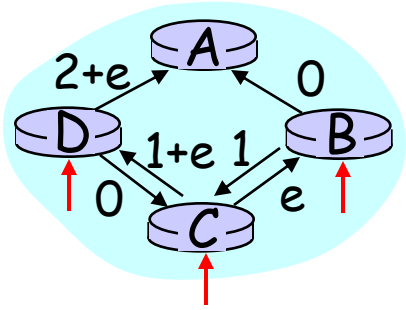
inicialmente



... recalcula rotas



... recalcula



... recalcula

Um algoritmo de roteamento de "vetor de distâncias" (VD)

iterativo:

- continua até que não haja mais troca de info. entre nós
- *se auto-termina*: não há "sinal" para parar

assíncrono:

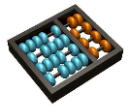
- os nós *não* precisam trocar info./iterar de forma sincronizada!

distribuído:

- cada nó comunica *apenas* com seus vizinhos diretos

Estrutura de dados: Tabela de Distâncias

- cada nós possui sua própria TD
- 1 linha para cada destino possível
- 1 coluna para cada vizinho direto



Roteamento vetor de distâncias: sumário

Iterativo, assíncrono: cada iteração local causada por:

- mudança do custo do enlace local
- mensagem do vizinho: mudança de caminho de menor custo para algum destino

Distribuído:

- cada nó avisa a seus vizinhos *apenas* quando muda seu caminho de menor custo para qualquer destino
  - ✓ os vizinhos então avisam a seus vizinhos, se for necessário

Cada nó:

espera (mudança no custo de mensagem do vizinho)

↓

recalcula tabela de distâncias

↓

se mudou o caminho de menor custo para qq. destino, *avisa* vizinhos

Algoritmo Vetor Distância

- $D_x(y)$ = estimativa do menor custo de x para y
- Nó x sabe o custo para seu vizinho v $c(x,v)$
- Nó x mantém vetor distância $\mathbf{D_x} = [D_x(y): y \in N]$
- Nó x mantém informação do vetor distância dos seus vizinhos
  - ✓ Para caa vizinho:
    $\mathbf{D_v} = [D_v(y): y \in N]$

Algoritmo de Vetor Distância

Algoritmo Bellman-Ford

$d_x(y)$:= menor custo de se ir de x para y

$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

Mínimo entre todos os vizinhos

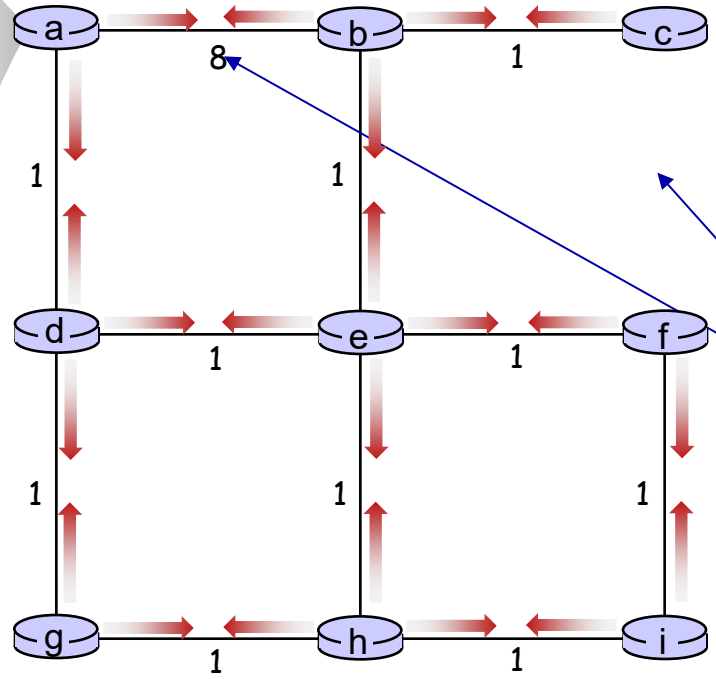
Distance vector: example

t=0

- All nodes have distance estimates to nearest neighbors (only)
- All nodes send their local distance vector to their neighbors

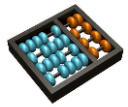
DV in

$D_a(a)=0$
$D_a(b) = 8$
$D_a(c) = \infty$
$D_a(d) = 1$
$D_a(e) = \infty$
$D_a(f) = \infty$
$D_a(g) = \infty$
$D_a(h) = \infty$
$D_a(i) = \infty$



A few asymmetries:
- missing link
- larger cost

UNICAMP

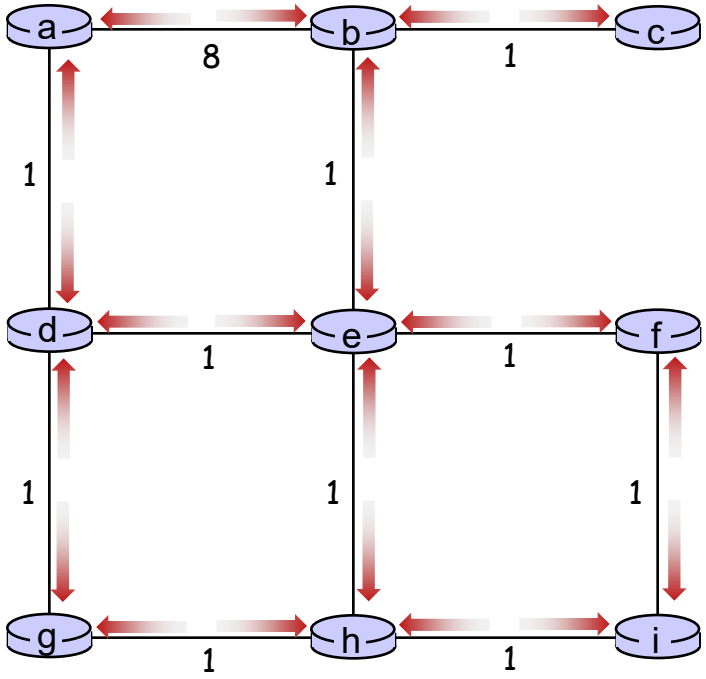


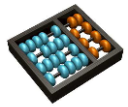
Distance vector example: iteration

t=1

All nodes:

- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to



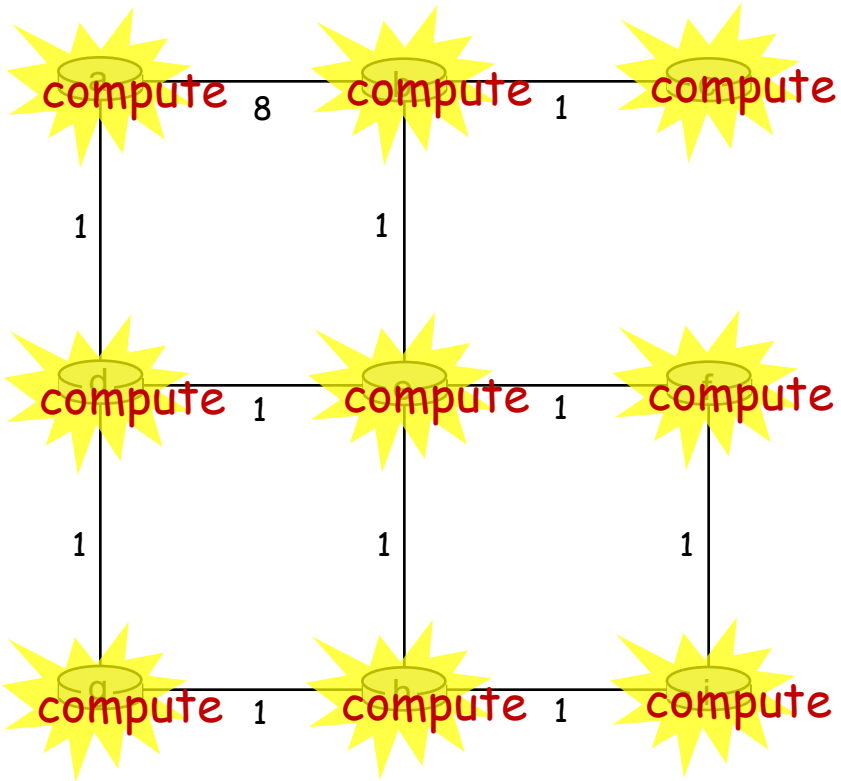


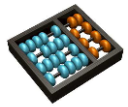
Distance vector example: iteration

t=1

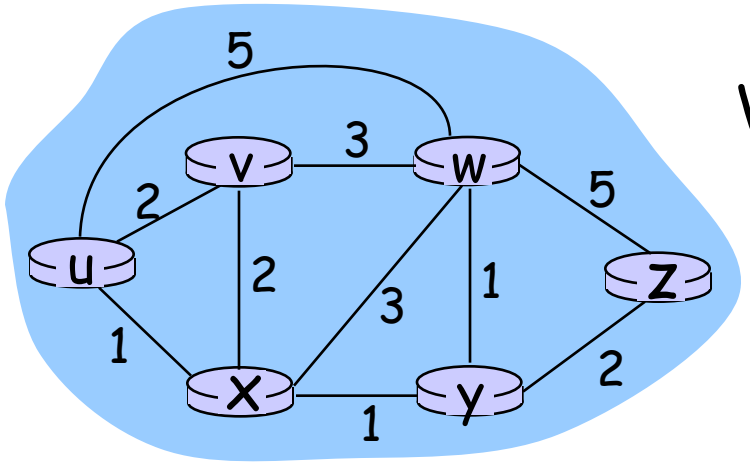
All nodes:

- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to





Exemplo



Vê-se que: $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) =$

$$d_u(z) = \min \{ c(u,v) + d_v(z),$$
$$c(u,x) + d_x(z),$$
$$c(u,w) + d_w(z) \}$$
$$= \min \{2 + 5,$$
$$1 + 3,$$
$$5 + 3\} = 4$$

$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$
$= \min\{2+0 , 7+1\} = 2$

$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$
$= \min\{2+1 , 7+0\} = 3$

Tabela nó x

De	Custo para x	Custo para y	Custo para z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

De	Custo para x	Custo para y	Custo para z
x	0	2	3
y	2	0	1
z	7	1	0

Tabela nó y node

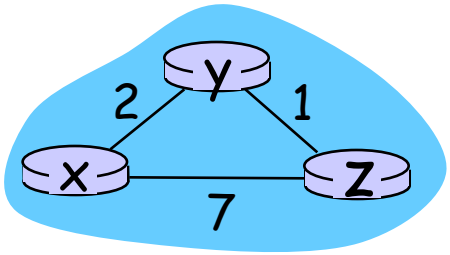
cost to

De	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

Tabela nó z

cost to

De	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



time

$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$
$= \min\{2+0 , 7+1\} = 2$

$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$
$= \min\{2+1 , 7+0\} = 3$

Tabela nó x

Custo para

De	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

De	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

De	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

Tabela nó y

Custo para

De	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

De	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

De	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

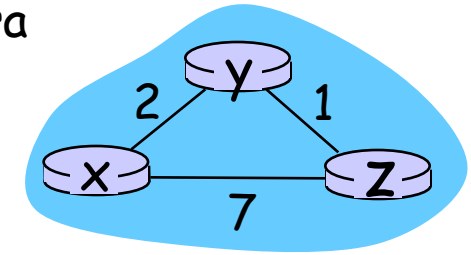
Tabela nó z

Custo para

De	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

De	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

De	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

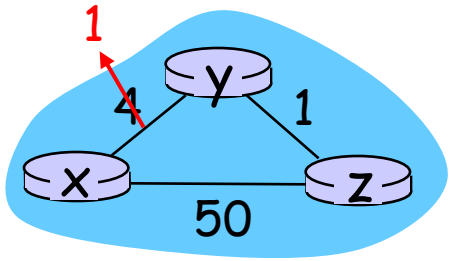


time

Dimuição no custo de enlace

Mudanca no estado do enlace:

- Nó detecta mudanca no custo do enlace
- Atualiza informação de roteamento, recalcula vetor
- Se distâncias alterada, notifica vizinhos



"Boas notícias Caminham rápido"

No tempo t_0, y detecta mudanca no custo e notifica vizinho

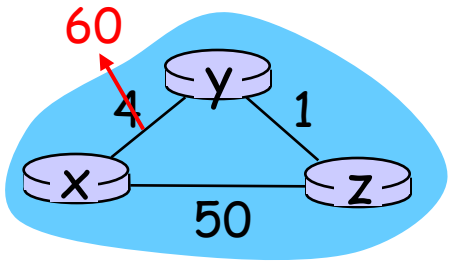
No tempo t_1, z recebe atualização de y e atualiza sua tabela , Computa um novo custo para x e envia para seus vizinhos a sua VD

No tempo t_2, y recebe a atualização de z e atualiza a sua tabela de distância, o menor custo de y não se altera e consequentemente não envia nenhuma mensagem para z

Aumento do custo de enlace

Mudanca de custo no enlace:

- Notícias ruins demoram a propagar, problema de "contagem até
- 44 interações até estabilizar



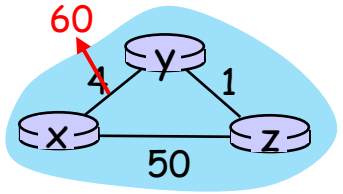
Reverso envenenado:

- Se Z rotea através de Y para chegar a X :
  - ✓ Z diz a Y que sua distância a x é infinita, de tal forma que Y não vai rotear através de Z

Distance vector: link cost changes

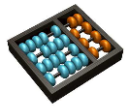
link cost changes:

- node detects local link cost change
- "bad news travels slow" – count-to-infinity problem:



- *y* sees direct link to *x* has new cost 60, but *z* has said it has a path at cost of 5. So *y* computes "my new cost to *x* will be 6, via *z*); notifies *z* of new cost of 6 to *x*.
- *z* learns that path to *x* via *y* has new cost 6, so *z* computes "my new cost to *x* will be 7 via *y*), notifies *y* of new cost of 7 to *x*.
- *y* learns that path to *x* via *z* has new cost 7, so *y* computes "my new cost to *x* will be 8 via *y*), notifies *z* of new cost of 8 to *x*.
- *z* learns that path to *x* via *y* has new cost 8, so *z* computes "my new cost to *x* will be 9 via *y*), notifies *y* of new cost of 9 to *x*.

UNICAMP



Problema de Convergência

- Soluções:
  - ✓ Horizonte dividido (Split horizon)
    - Anuncio de rotas para vizinho não deve conter rotas aprendidads por anuncios do próprio vizinho
  - ✓ Reverso envenenado (Poisson reverse)
    - Anuncia custo infinito para vizinho que faz parte do loop.
  - ✓ Temporização de retenção (Hold down timer)
    - Retem a informação de menor custo por um tempo igual ao período pré-estabelecido de temporização
    - Solução para loop envolvendo mais de dois
    - Aumenta tempo de convergência das atualizações das tabelas

Comparação dos algoritmos EE e VD

Complexidade de mensagens

- EE: com n nós, E enlaces, O(nE) mensagens enviadas
- VD: trocar mensagens apenas entre vizinhos
  - ✓ varia o tempo de convergência

Rapidez de Convergência

- EE: algoritmo O(n**2) requer O(nE) mensagens
  - ✓ podem ocorrer oscilações
- VD: varia tempo para convergir
  - ✓ podem ocorrer rotas cíclicas
  - ✓ problema de contagem ao infinito

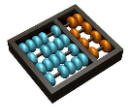
Robustez: o que acontece se houver falha do roteador?

EE:

- ✓ nó pode anunciar valores incorretos de custo de *enlace*
- ✓ cada nó calcula sua *própria* tabela

VD:

- ✓ um nó VD pode anunciar um custo de *caminho* incorreto
- ✓ a tabela de cada nó é usada pelos outros nós
  - erros se propagam pela rede



Roteamento Hierárquico

Neste estudo de roteamento fizemos uma idealização:

- todos os roteadores idênticos
- rede "não hierarquizada" ("flat")

… *não* é verdade, na prática

escala: 200 milhões de destinos:

- impossível guardar todos destinos na tabela de rotas!
- troca de tabelas de rotas afogaria os enlaces!

autonomia administrativa

- internet = rede de redes
- cada administrador de rede pode querer controlar roteamento em sua própria rede

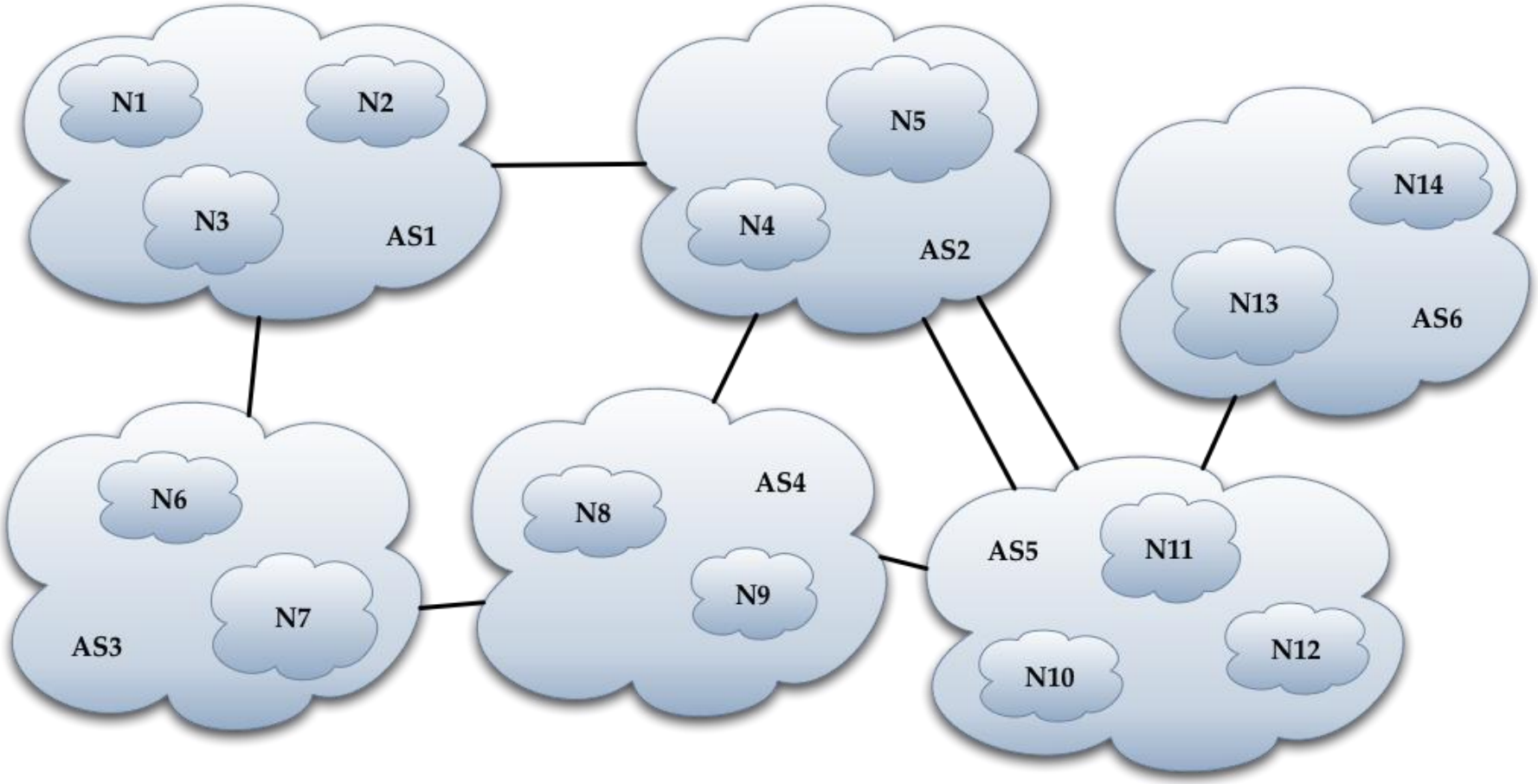
Roteamento Hierárquico

- agregar roteadores em regiões, "sistemas autônomos" (SAs)
- roteadores no mesmo SA usam o mesmo protocolo de roteamento
  - protocolo de roteamento "intra-SA"
  - roteadores em SAs diferentes podem usar diferentes protocolos de roteamento intra-SA

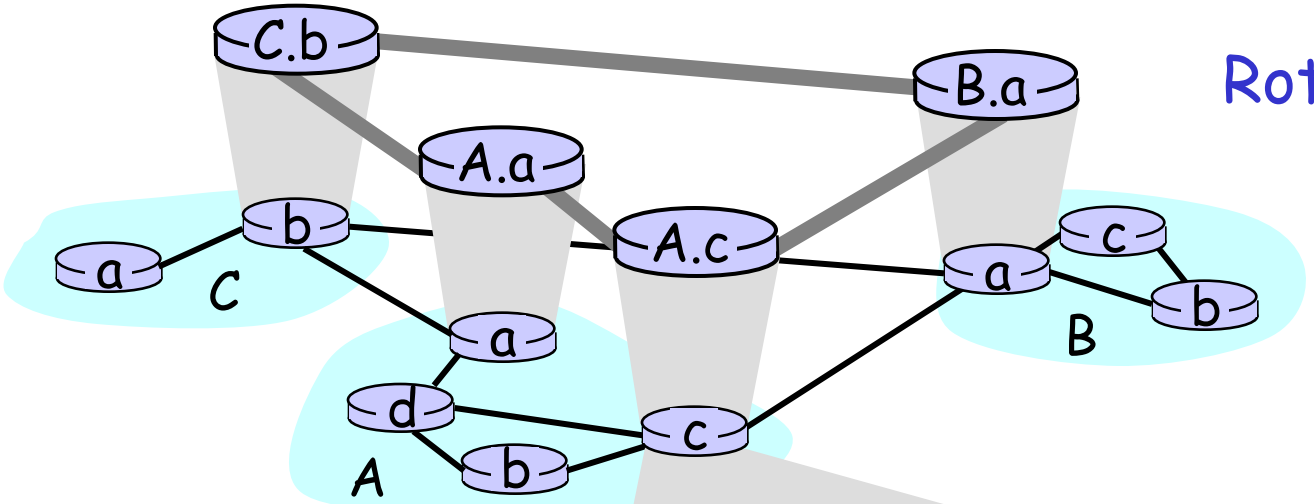
roteadores de borda

- roteadores especiais no SA
- usam protocolo de roteamento intra-SA com todos os demais roteadores no SA
- *também* responsáveis por rotear para destinos fora do SA
  - usam protocolo de roteamento "inter-SA" com outros roteadores de borda

Visão abstrata da Internet



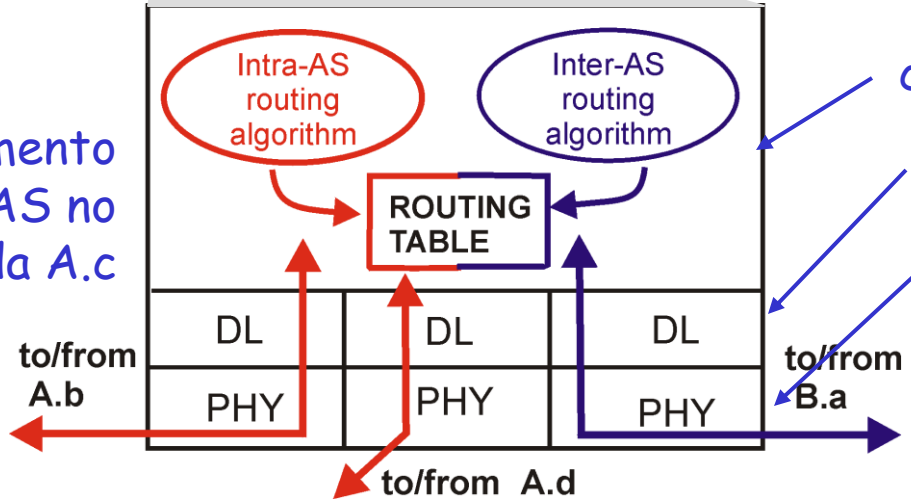
Roteamento Intra-SA e Inter-SA



Roteadores de borda:

- fazem roteamento inter-SA entre si
- fazem roteamento intra-SA com outros roteadores do seu próprio SA

Roteamento inter-AS, intra-AS no roteador de borda A.c

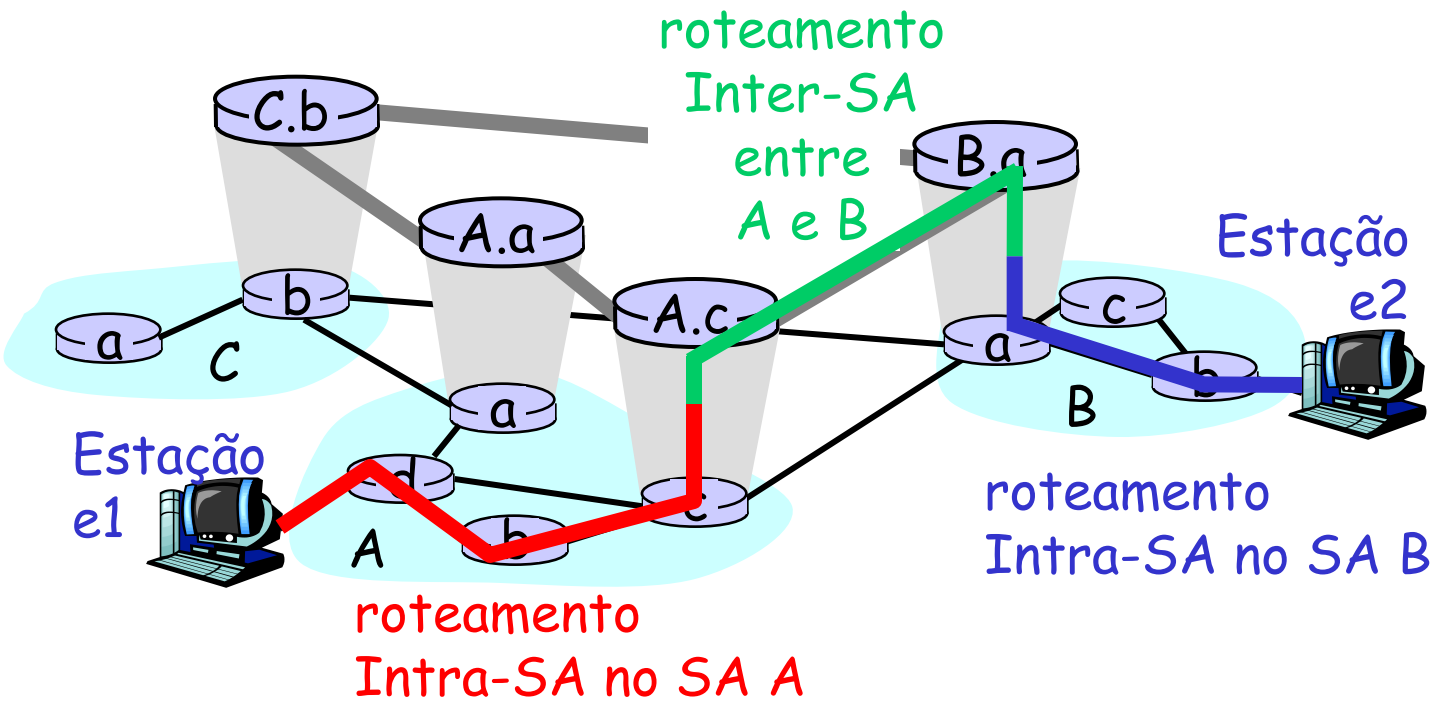


camada de rede

camada de enlace

camada física

Roteamento Intra-SA e Inter-SA



- Em breve veremos protocolos de roteamento inter-SA e intra-SA específicos da Internet

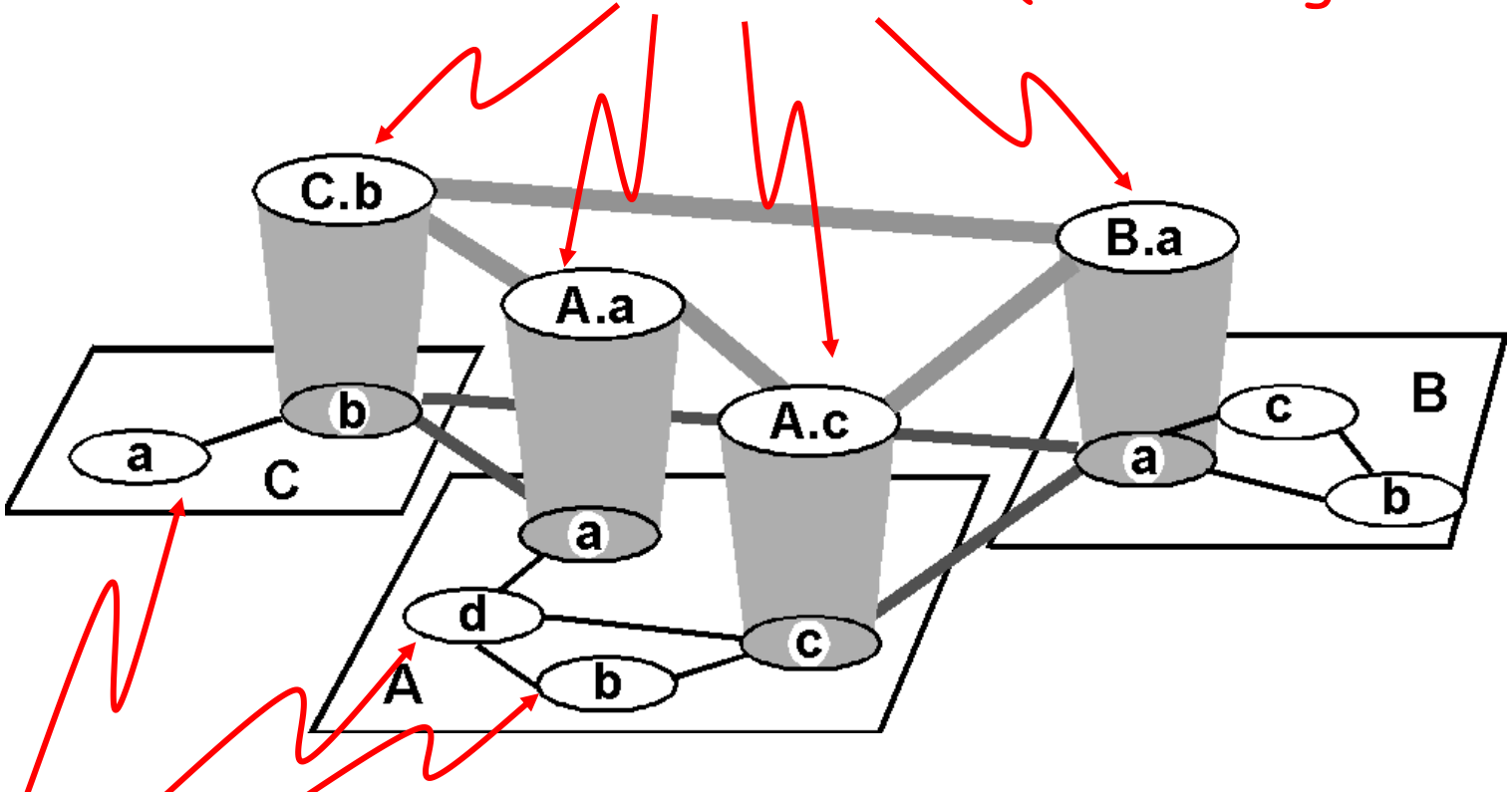
Roteamento na Internet

- A Internet Global consiste de Sistemas Autonômos (SAs) interligados entre si:
  - ✓ **SA Folha**: empresa pequena
  - ✓ **SA com Múltipla Conectividade**: empresa grande (sem trânsito)
  - ✓ **SA de Trânsito**: provedor

- Roteamento em dois níveis:
  - ✓ **Intra-SA**: administrador é responsável pela escolha
  - ✓ **Inter-SA**: padrão único

Hierarquia de SAs na Internet

Inter-SA: roteadores de fronteira (exterior gateways)



Intra-SA: roteadores internos (interior gateways)

Roteamento Intra-SA

- Também conhecido como Interior Gateway Protocols (IGP) (protocolos de roteamento interno)
- Os IGPs mais comuns são:
  - ✓ RIP: *Routing Information Protocol*
  - ✓ OSPF: *Open Shortest Path First*
  - ✓ IGRP: *Interior Gateway Routing Protocol* (proprietário da Cisco)

RIP (*Routing Information Protocol*)

- Algoritmo do tipo vetor de distâncias
- Incluído na distribuição do BSD-UNIX em 1982
- Métrica de distância: # de enlaces (máx = 15 enlaces)
- Vetores de distâncias: trocados a cada 30 seg via Mensagem de Resposta (tb chamada de **anúncio**)
- Cada anúncio: rotas para 25 redes destino

OSPF (*Open Shortest Path First*)

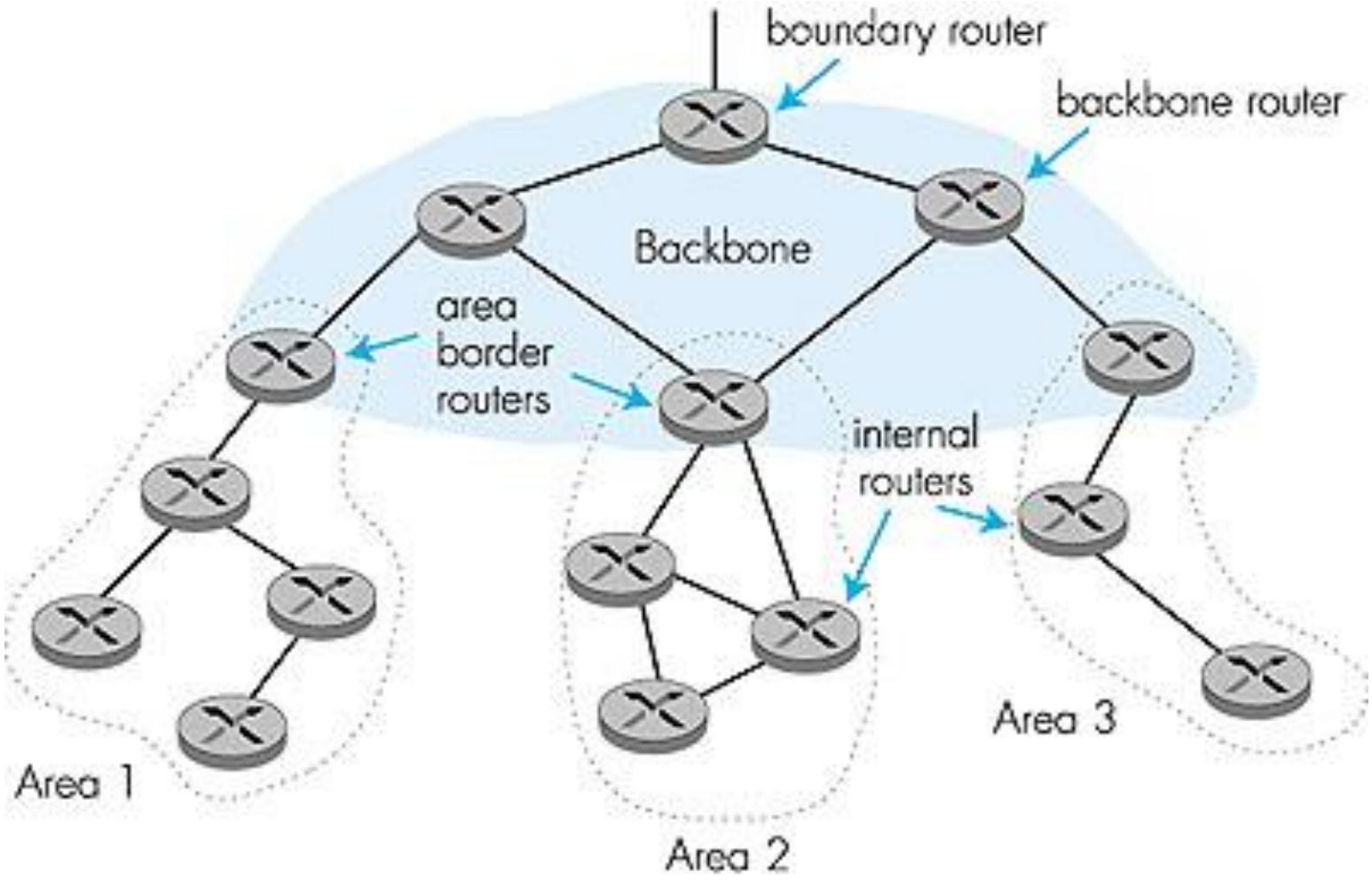
- "open" (aberto): publicamente disponível
- Usa algoritmo do Estado de Enlaces
  - ✓ disseminação de pacotes EE
  - ✓ Mapa da topologia a cada nó
  - ✓ Cálculo de rotas usando o algoritmo de Dijkstra

- Anúncio de OSPF inclui uma entrada por roteador vizinho
- Anúncios disseminados para SA inteiro (via inundação)

OSPF: características "avançadas" (não existentes no RIP)

- Segurança: todas mensagens OSPF autenticadas (para impedir intrusão maliciosa); conexões TCP usadas
- Caminhos Múltiplos de custos iguais permitidos (o RIP permite e usa apenas uma rota)
- Para cada enlace, múltiplas métricas de custo para TOS diferentes (p.ex, custo de enlace de satélite colocado como "baixo" para melhor esforço; "alto" para tempo real)
- Suporte integrado para ponto a ponto e multiponto:
  - ✓ OSPF multiponto (MOSPF) usa mesma base de dados de topologia usado por OSPF
- OSPF hierárquico em domínios grandes.

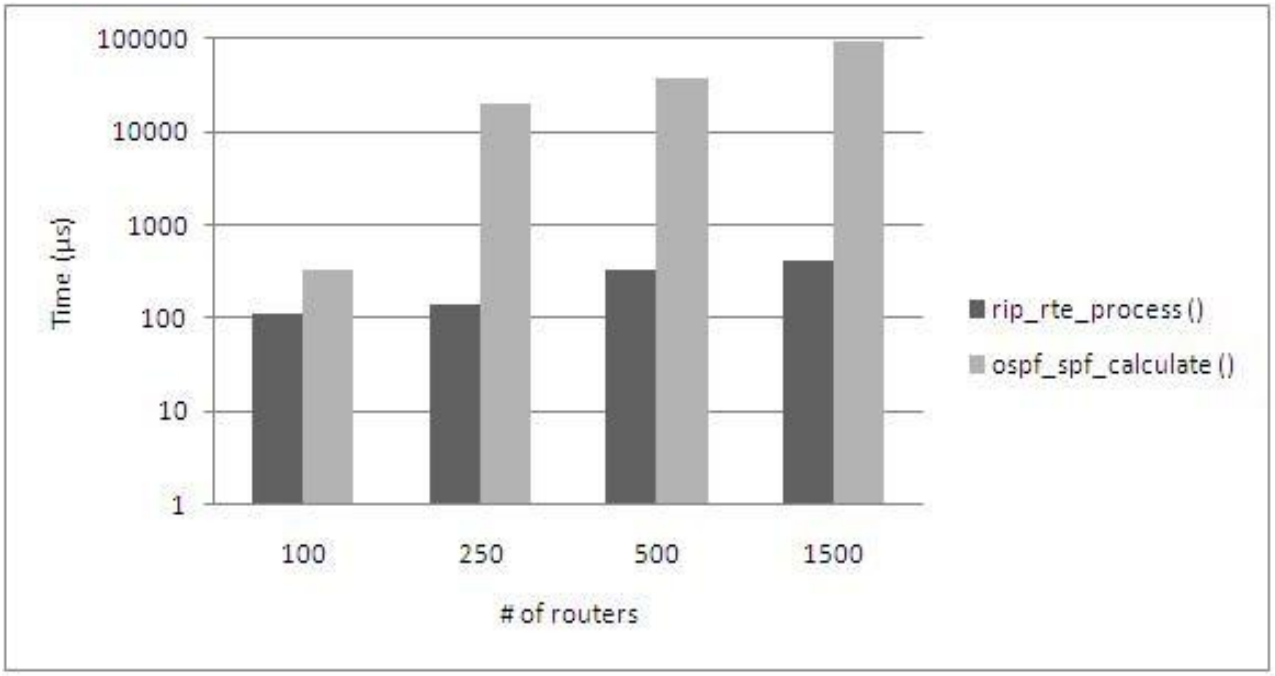
OSPF Hierárquico



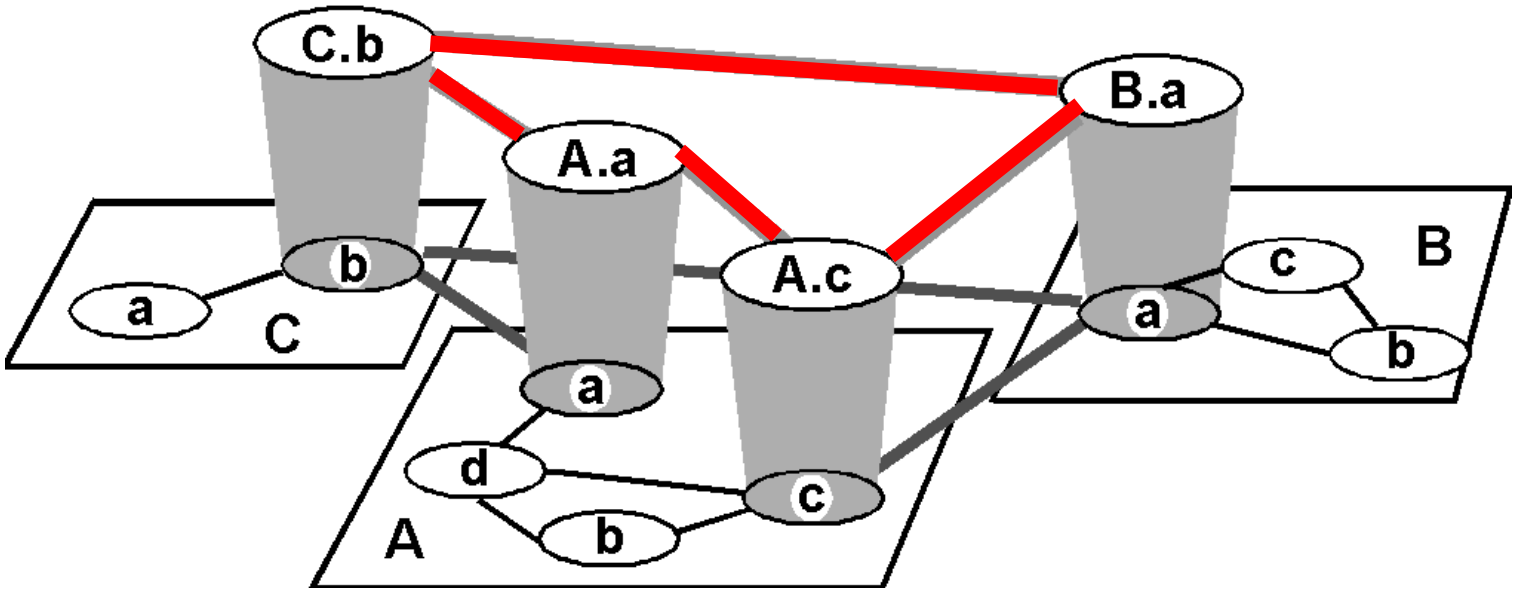
OSPF Hierárquico

- **Hierarquia de dois níveis:** área local, backbone.
  - ✓ Anúncios de EE disseminados apenas na mesma área
  - ✓ cada nó possui topologia detalhada da área; apenas sabe a direção (caminho mais curto) para redes em outras áreas (alcançadas através do backbone).
- **Roteador de fronteira de área:** "sumariza" distâncias às redes na sua própria área, anuncia a outros roteadores de fronteira de área.
- **Roteadores do backbone:** realizam roteamento OSPF limitado ao backbone.
- **Roteadores de fronteira:** ligam a outros SAs.

Comparativo Desempenho



Roteamento Inter-SA



Operação BGP

Q: O que um roteador BGP faz?

- Envia anúncio de rotas para seus vizinhos;
- Recebe e filtra anúncios de rotas dos seus vizinhos diretamente conectados
- Escolha da rota .
  - ✓ Para rotear para o destino X, qual caminho (entre tantos anunciados) deve ser seguindo?

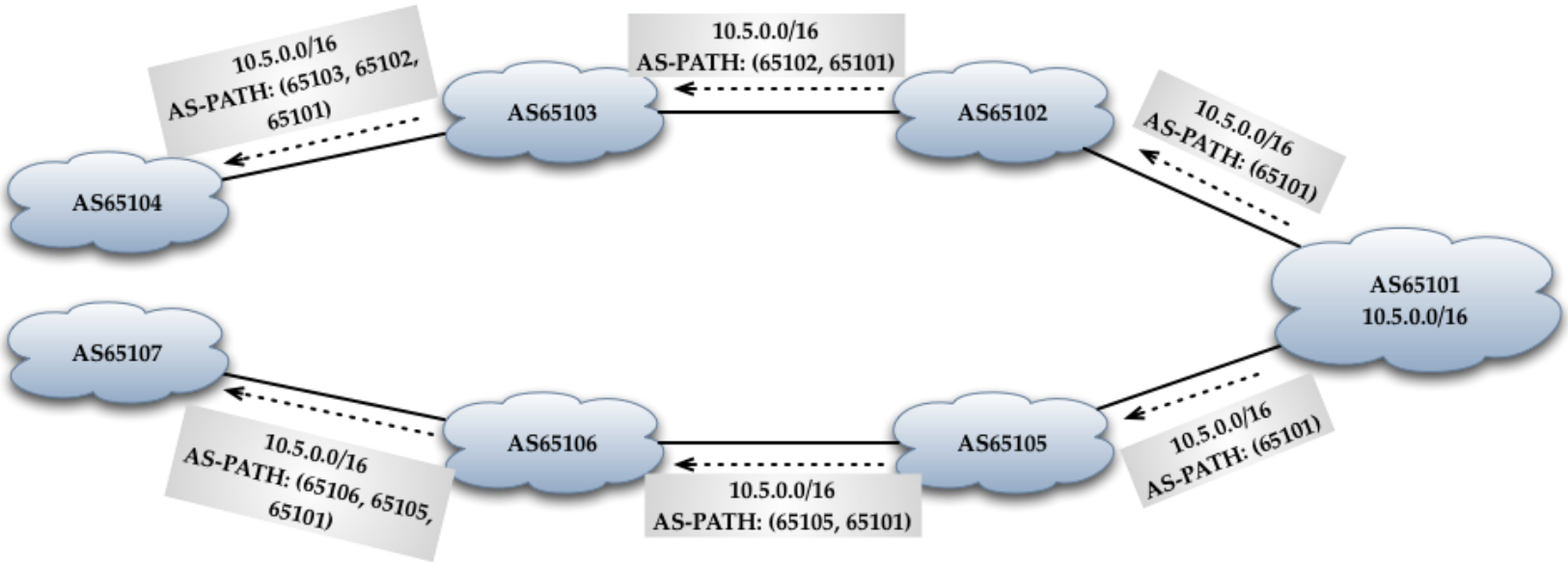
BGP

- BGP (Border Gateway Protocol): é o padrão de fato para uso na Internet
- BGP provê cada AS dos meios para:
  1. Obter informações de alcance de sub-rede dos Ass. Vizinhos
  2. Propagar informações de alcance para todos os roteadores internos ao AS
  3. Determinar "boas" rotas para as sub-redes baseado em informações de alcance e política
- Permite que uma subnet comunique sua existência para o resto da Internet: **"Estou aqui"**

Rotas BGP

- Quando se comunica um prefixo, o comunicado inclui os atributos do BGP.
  - Prefixo + atributos = "rota"
- Dois atributos importantes:
- AS-PATH: contém os ASs pelos quais o comunicado para o prefixo passou: AS 67 AS 17
  - NEXT-HOP: Indica o roteador específico interno ao AS para o AS do próximo salto (next-hop). (Pode haver múltiplos links do AS atual para o AS do próximo salto.
- Quando um roteador gateway recebe um comunicado de rota, ele usa política de importação para aceitar/rejeitar.

Atributo AS_PATH



Seleção de Rotas - BGP

Um roteador pode aprender mais do que 1 rota para o mesmo prefixo. O roteador deve selecionar uma rota

- Regras de eliminação:
  - Atributo de valor de preferência local: decisão de política
  - AS-PATH (caminho) mais curto
  - Roteador do NEXT-HOP (próximo salto) mais próximo: roteamento da "batata quente"
  - Critérios adicionais

Selação de Rotas

Network	Next Hop	LOCAL_PREF	Weight	Best?	PATH	Origin
61.13.0.0/16	139.175.56.165		0	N	4780,9739	IGP
	140.123.231.103		0	N	9918,4780,9739	IGP
	140.123.231.100	0	0	Y	9739	IGP
61.251.128.0/20	139.175.56.165		0	Y	4780,9277,17577	IGP
	140.123.231.103		0	N	9918,4780,9277,17577	IGP
211.73.128.0/19	210.241.222.62		0	Y	9674	IGP
218.32.0.0/17	139.175.56.165		0	N	4780,9919	IGP
	140.123.231.103		0	N	9918,4780,9919	IGP
	140.123.231.106		0	Y	9919	IGP
218.32.128.0/17	139.175.56.165		0	N	4780,9919	IGP
	140.123.231.103		0	N	9918,4780,9919	IGP
	140.123.231.106		0	Y	9919	IGP

Porque protocolos Intra- e Inter-AS diferentes ?

Políticas:

- Inter-SA: administração quer controle sobre como tráfego roteado, quem transita através da sua rede.
- Intra-AS: administração única, logo são desnecessárias decisões políticas

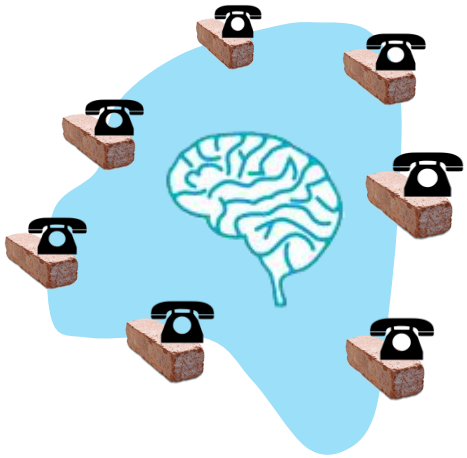
Escalabilidade:

- roteamento hierárquico economiza tamanho de tabela de rotas, reduz tráfego de atualização

Desempenho:

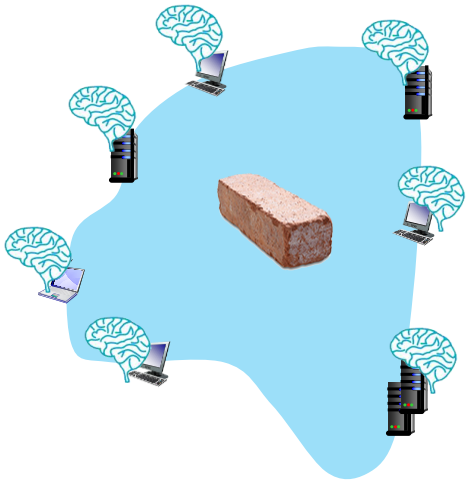
- Intra-AS: pode focar em desempenho
- Inter-AS: políticas podem ser mais importantes do que desempenho

Where's the intelligence?



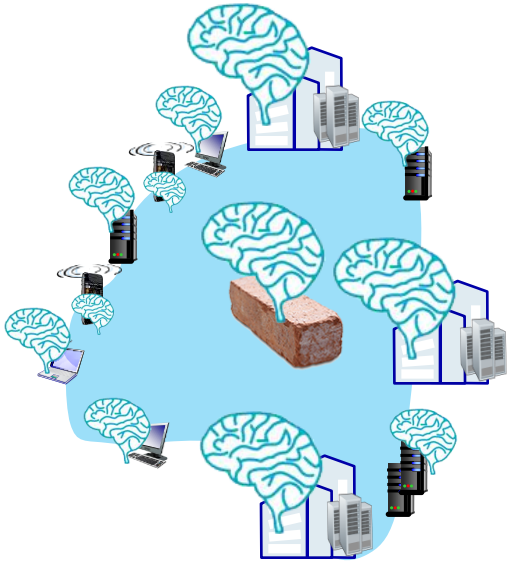
20th century phone net:

- intelligence/computing at network switches



Internet (pre-2005)

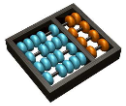
- intelligence, computing at edge



Internet (post-2005)

- programmable network devices
- intelligence, computing, massive application-level infrastructure at edge

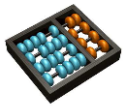
UNICAMP



Software defined networking (SDN)

- Internet network layer: historically implemented via distributed, per-router control approach:
  - ✓ *monolithic* router contains switching hardware, runs proprietary implementation of Internet standard protocols (IP, RIP, IS-IS, OSPF, BGP) in proprietary router OS (e.g., Cisco IOS)
  - ✓ different "middleboxes" for different network layer functions: firewalls, load balancers, NAT boxes, ..
- ~2005: renewed interest in rethinking network control plane

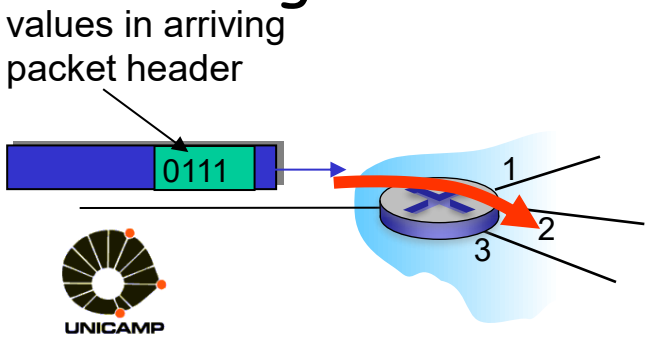
UNICAMP



Network layer: data plane, control plane

Data plane

- local, per-router function
- determines how datagram arriving on router input port is forwarded to router output port
- forwarding function

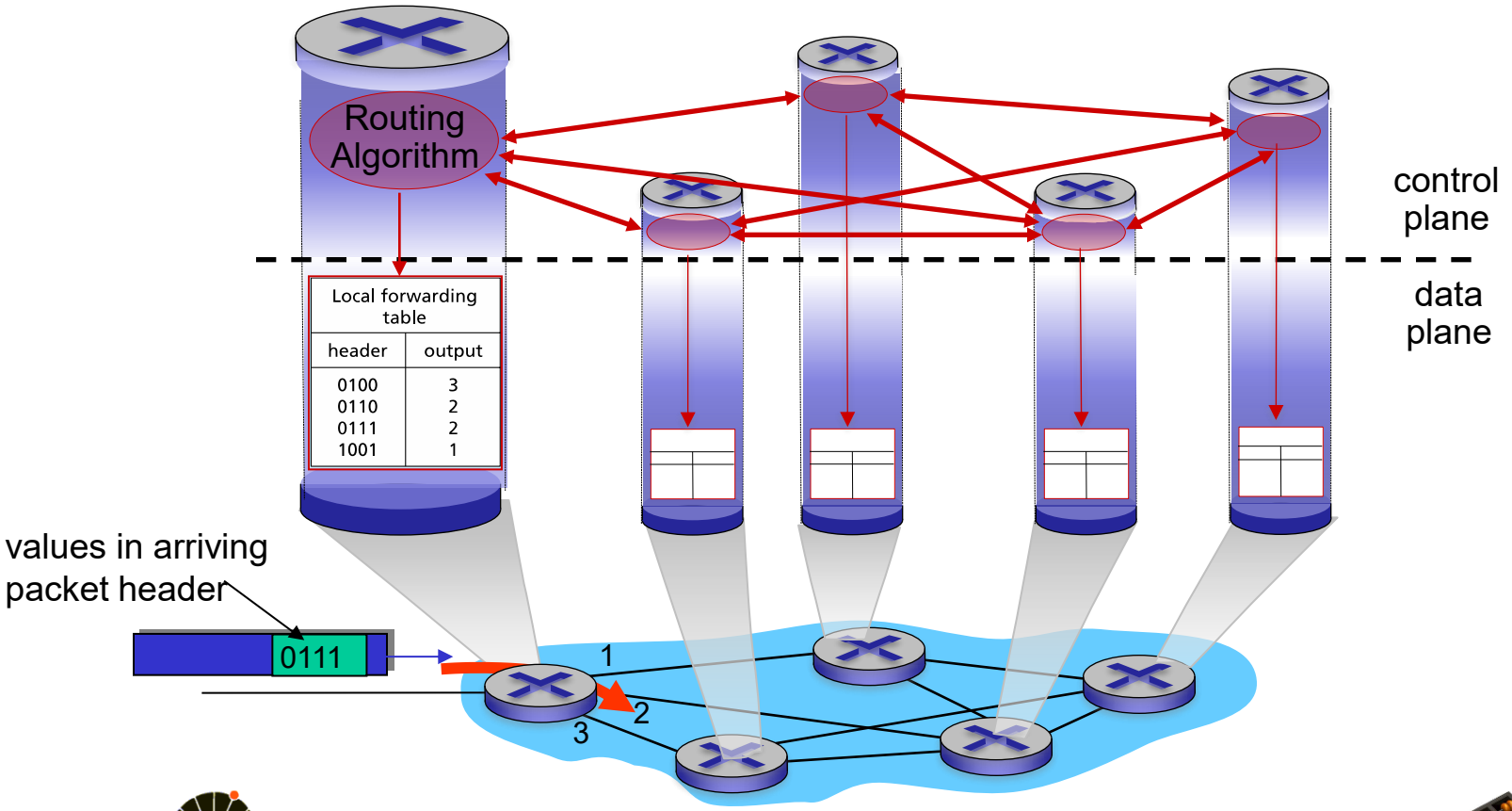


Control plane

- network-wide logic
- determines how datagram is routed among routers along end-end path from source host to destination host
- two control-plane approaches:
  - *traditional routing algorithms:* implemented in routers
  - *software-defined networking (SDN)*: implemented in (remote) servers

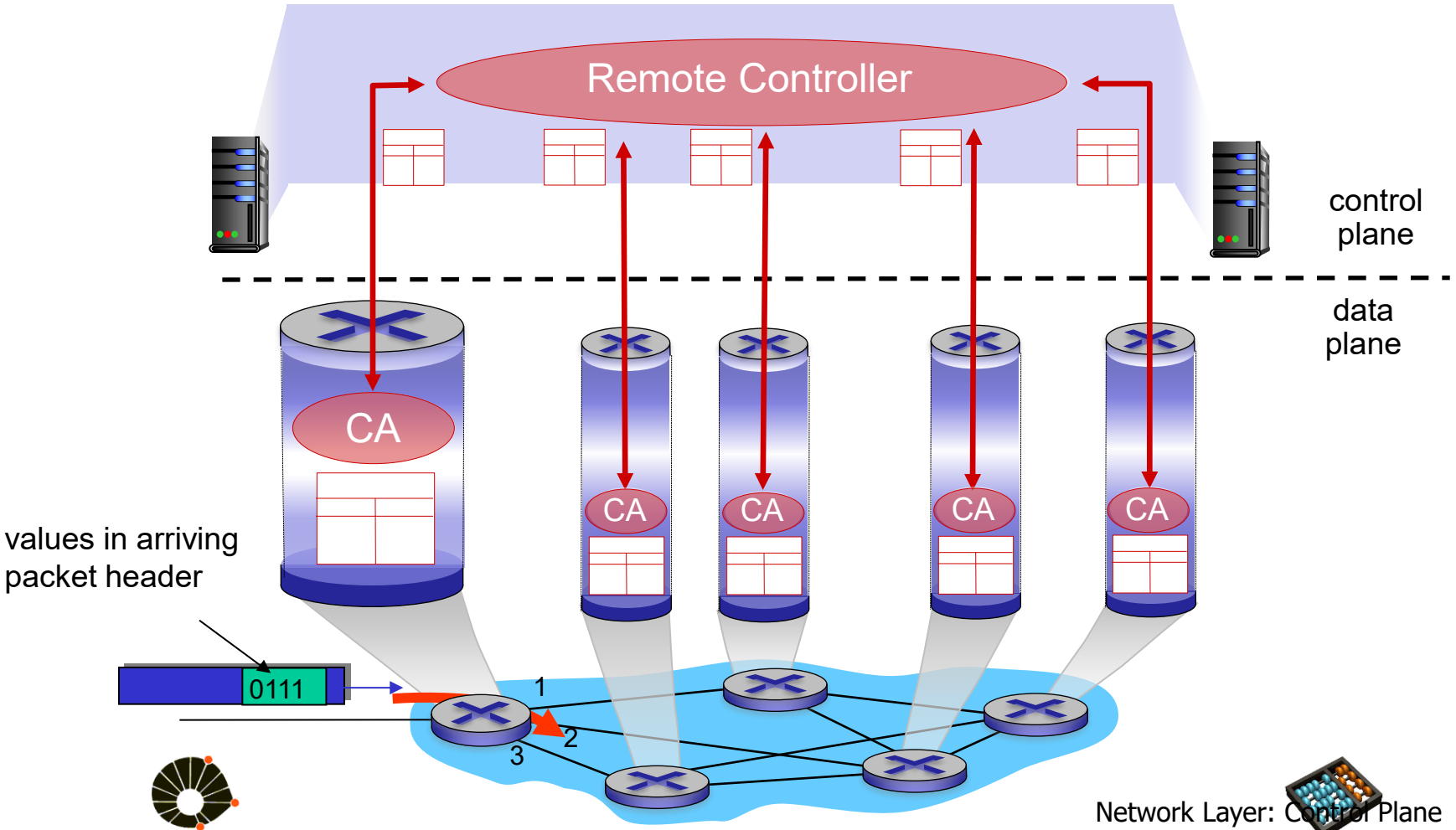
Per-router control plane

Individual routing algorithm components *in each and every router* interact in the control plane



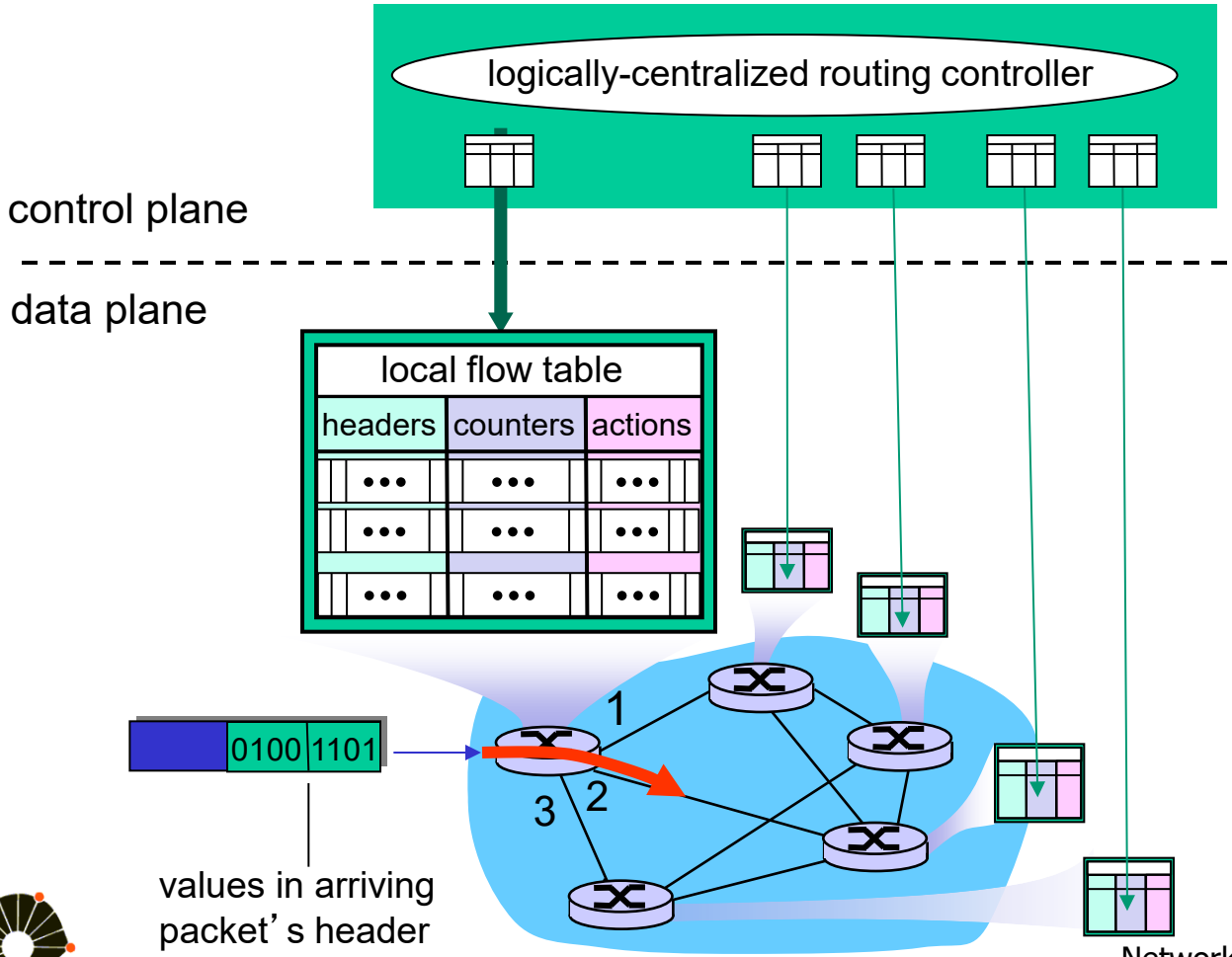
Logically centralized control plane

A distinct (typically remote) controller interacts with local control agents (CAs)



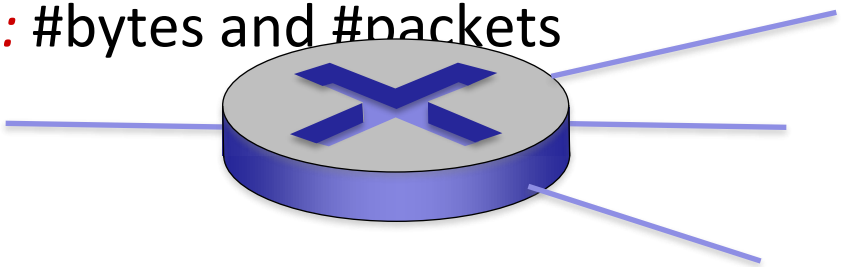
Generalized Forwarding and SDN

Each router contains a *flow table* that is computed and distributed by a *logically centralized* routing controller



OpenFlow data plane abstraction

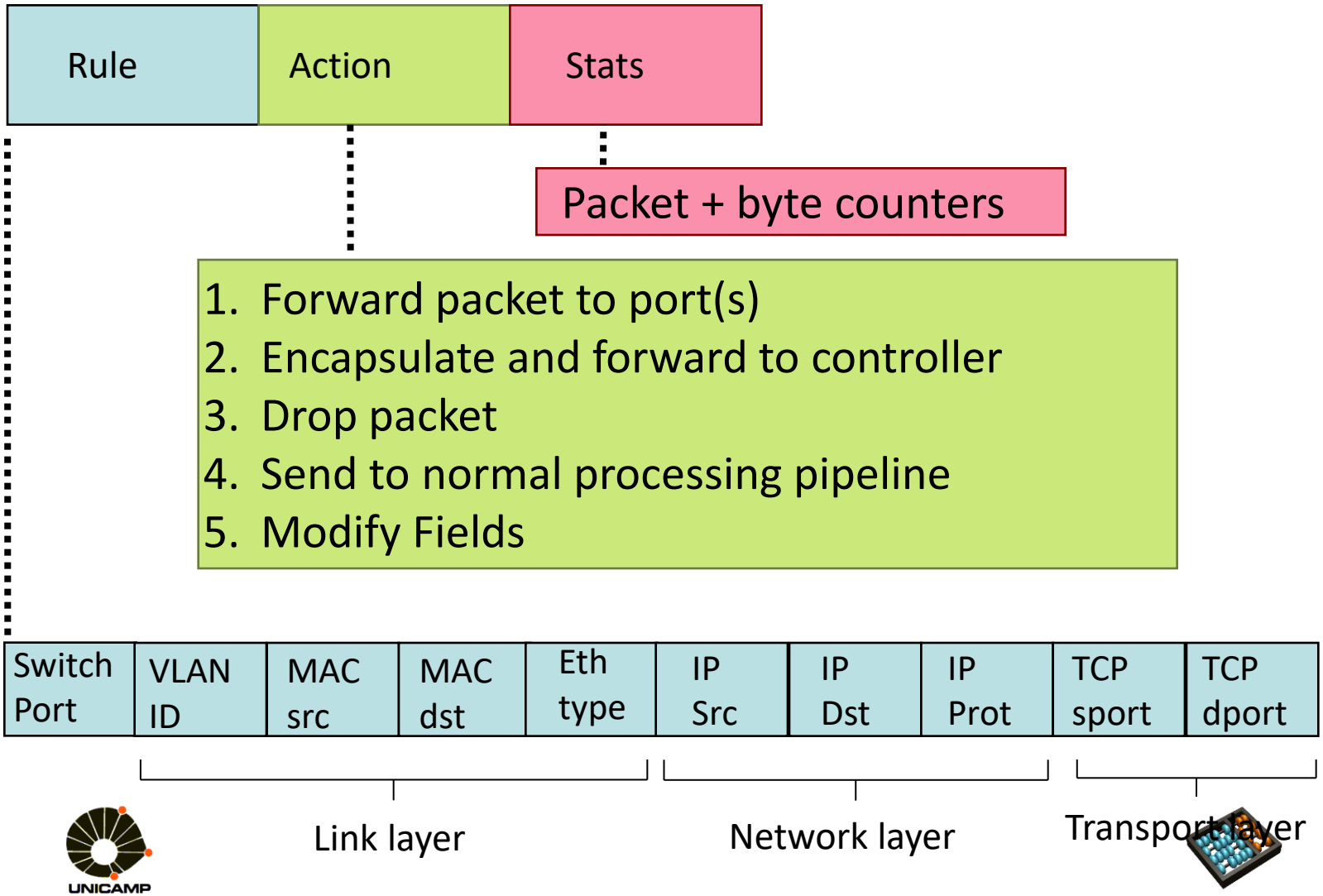
- *flow*: defined by header fields
- generalized forwarding: simple packet-handling rules
  - ✓ *Pattern:* match values in packet header fields
  - ✓ *Actions: for matched packet:* drop, forward, modify, matched packet or send matched packet to controller
  - ✓ *Priority*: disambiguate overlapping patterns
  - ✓ *Counters:* #bytes and #packets



* : wildcard

1. src=1.2.*.*, dest=3.4.5.* → drop
2. src = *.*.*.*, dest=3.4.*.* → forward(2)
3. src=10.1.2.3, dest=*.*.*.* → send to controller

OpenFlow: Flow Table Entries



OpenFlow abstraction

- *match+action:* unifies different kinds of devices

- Router
  - *match:* longest destination IP prefix
  - *action:* forward out a link
- Switch
  - *match:* destination MAC address
  - *action:* forward or flood
- Firewall
  - *match*: IP addresses and TCP/UDP port numbers
  - *action:* permit or deny
- NAT
  - *match:* IP address and port
  - *action:* rewrite address and port

Open Networking Foundation

- Open Networking Foundation (ONF) is a user-driven organization dedicated to the promotion and adoption of Software-Defined Networking (SDN) through open standards development.

- https://www.opennetworking.org
  - ✓ Technical library, codes, video

UNICAMP