



IMD0043

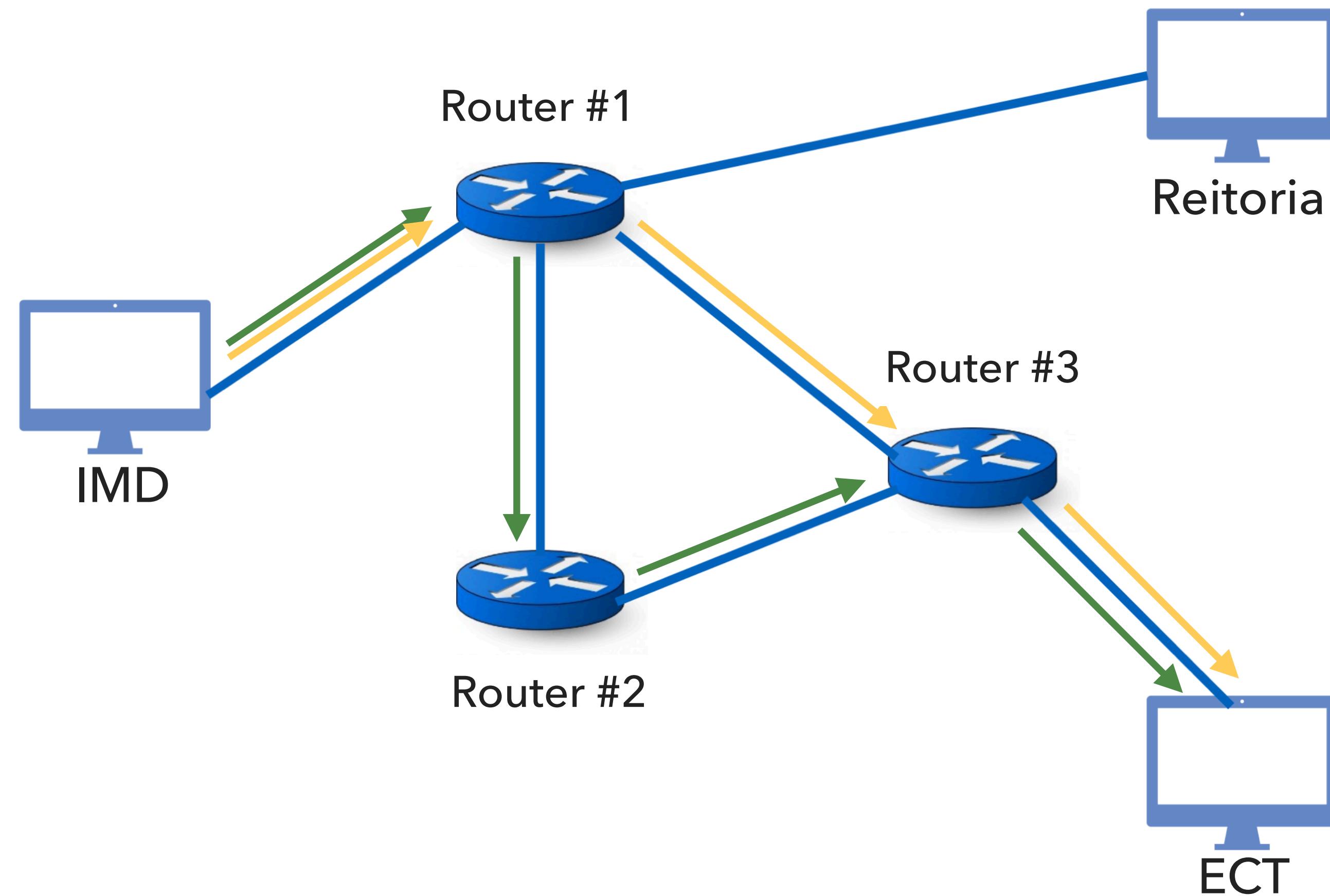
ARQUITETURA DO ROTEADOR

ROTEADORES IP (E SWITCHES L3...)

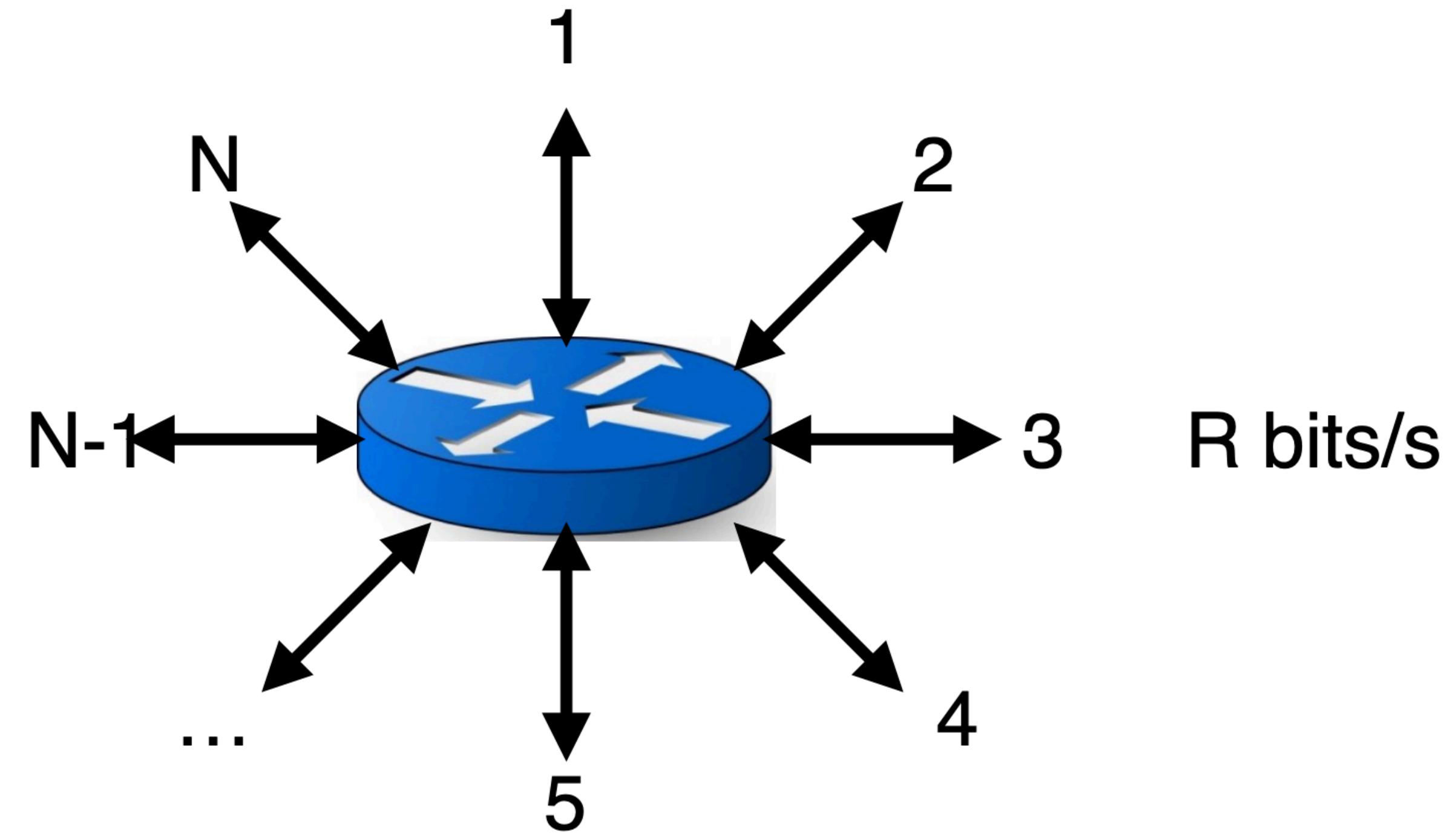
Peça fundamental da **infraestrutura da Internet**

Vendedores: Cisco, Huawei, Juniper, Arista, HPE ...

REVISANDO: ROTEADORES ENCAMINHAM PACOTES

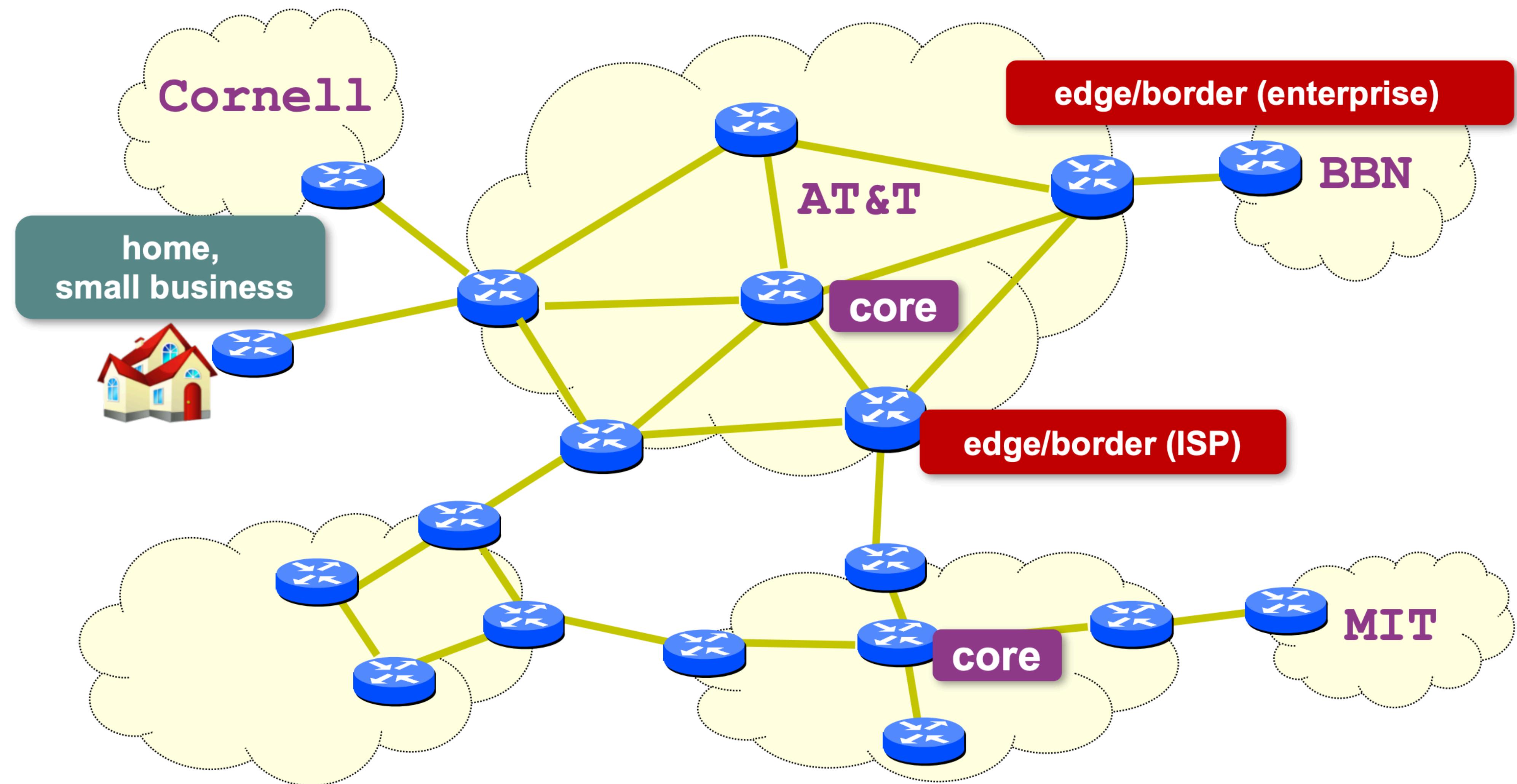


DEFINIÇÕES EM UM ROTEADOR



- ▶ N = No de portas
- ▶ R = velocidade (*line rate*) de uma porta
- ▶ Capacidade do roteador = $N \times R$

REDES E ROTEADORES



EXEMPLOS DE ROTEADORES

Core (ex: Cisco CRS-X)

- ▶ $R = 10/40/100 \text{ Gbps}$
- ▶ $NR = 922 \text{ Tbps}$
- ▶ Netflix: 3 GB/h ($\sim 6,5 \text{ Mb/s}$)
- ▶ ~ 140 milhões de usuários simultâneos

Edge (ISP) (ex: Cisco ASR)

- ▶ $R = 1/10/40 \text{ Gbps}$
- ▶ $NR = 120 \text{ Gbps}$

Edge (Enterprise) (ex: Cisco 3945E)

- ▶ $R = 10/100/1000 \text{ Mbps}$
- ▶ $NR < 10 \text{ Gbps}$



Cisco CRS

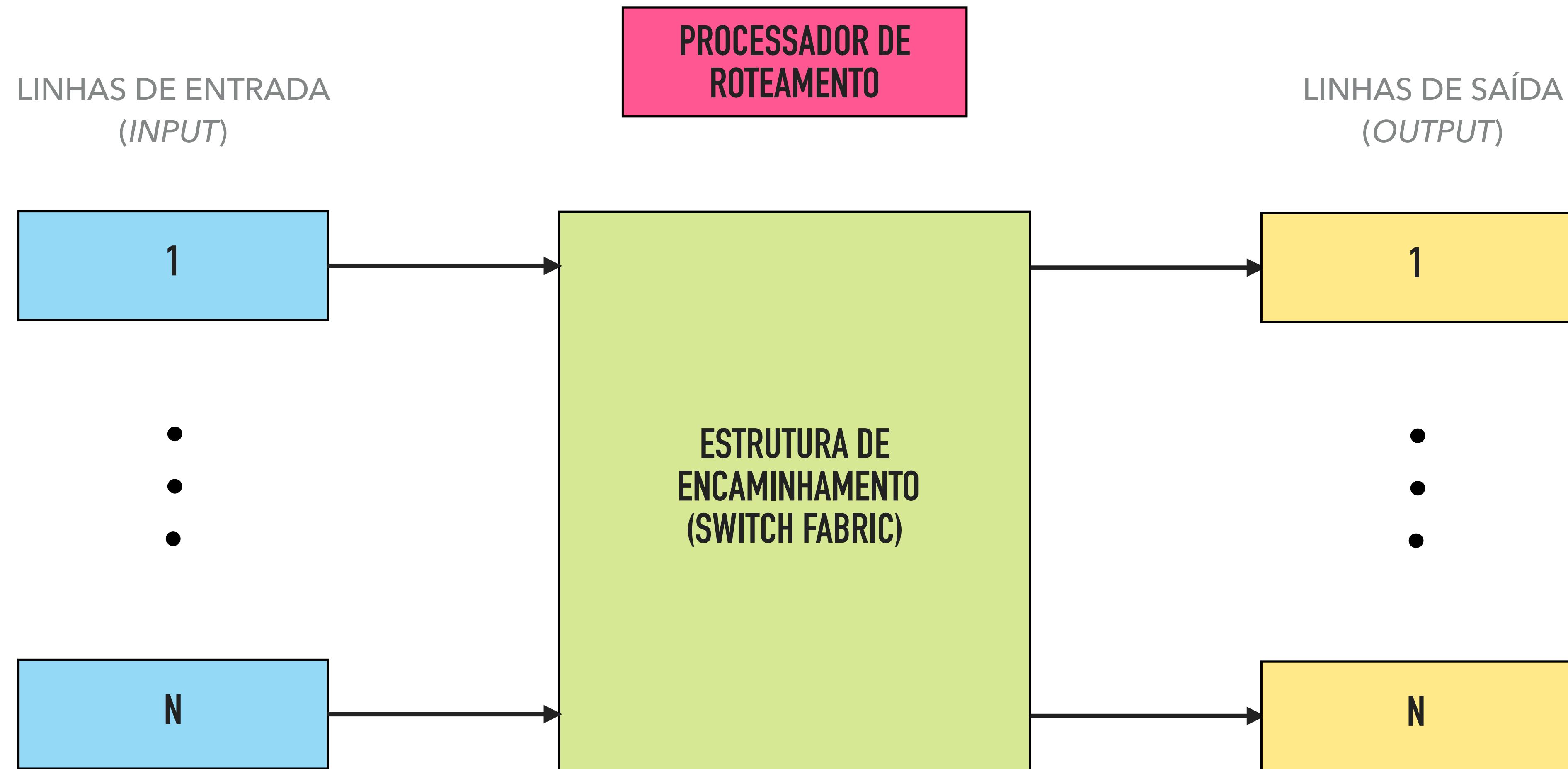


Cisco ASR

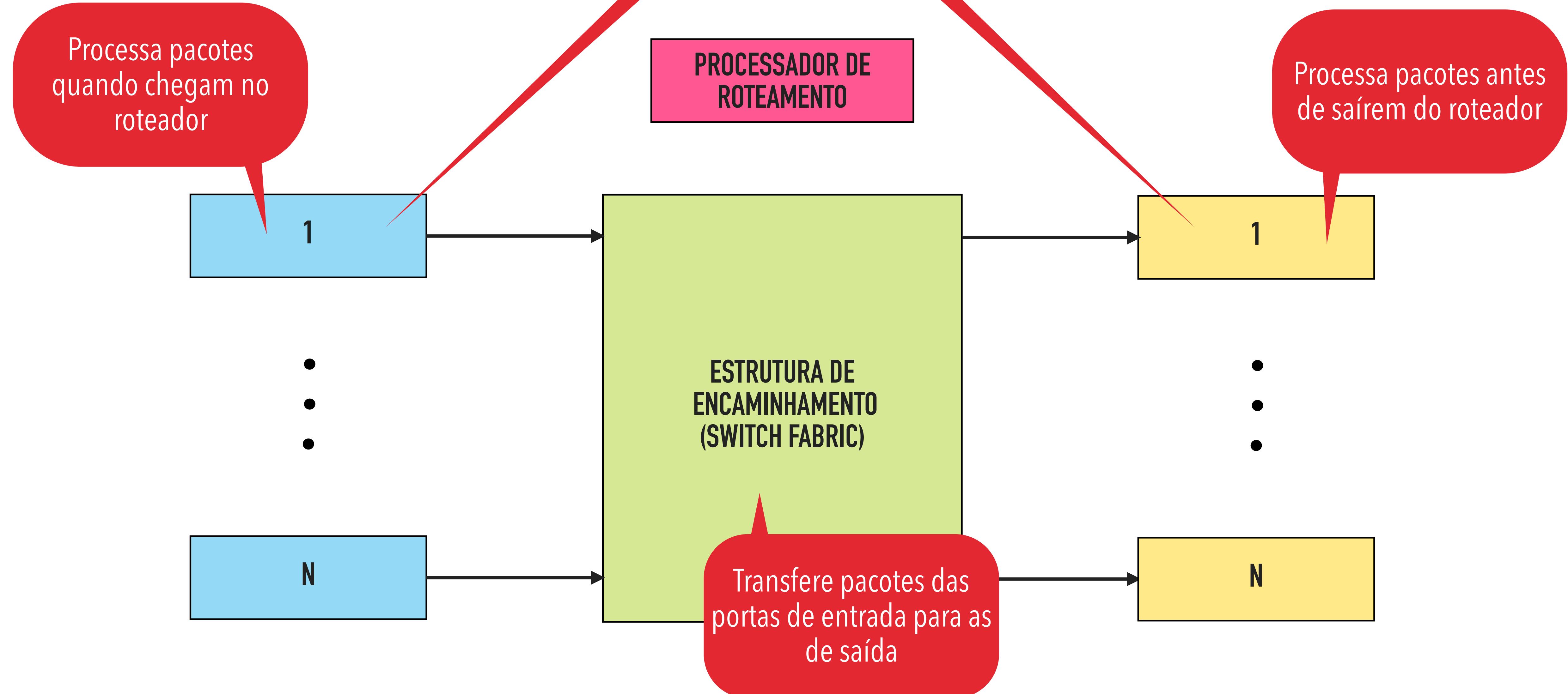


Cisco 3945E

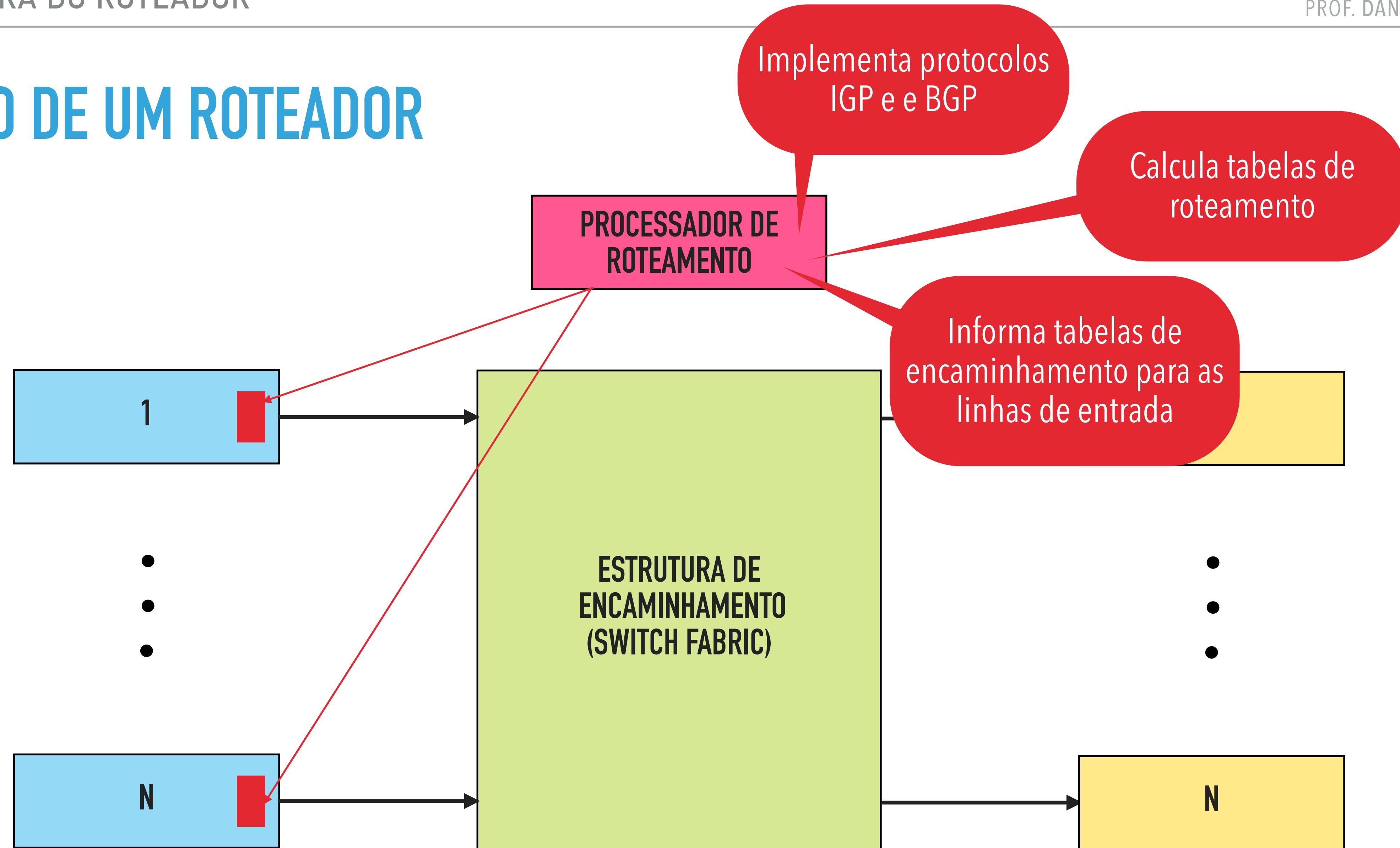
DENTRO DE UM ROTEADOR



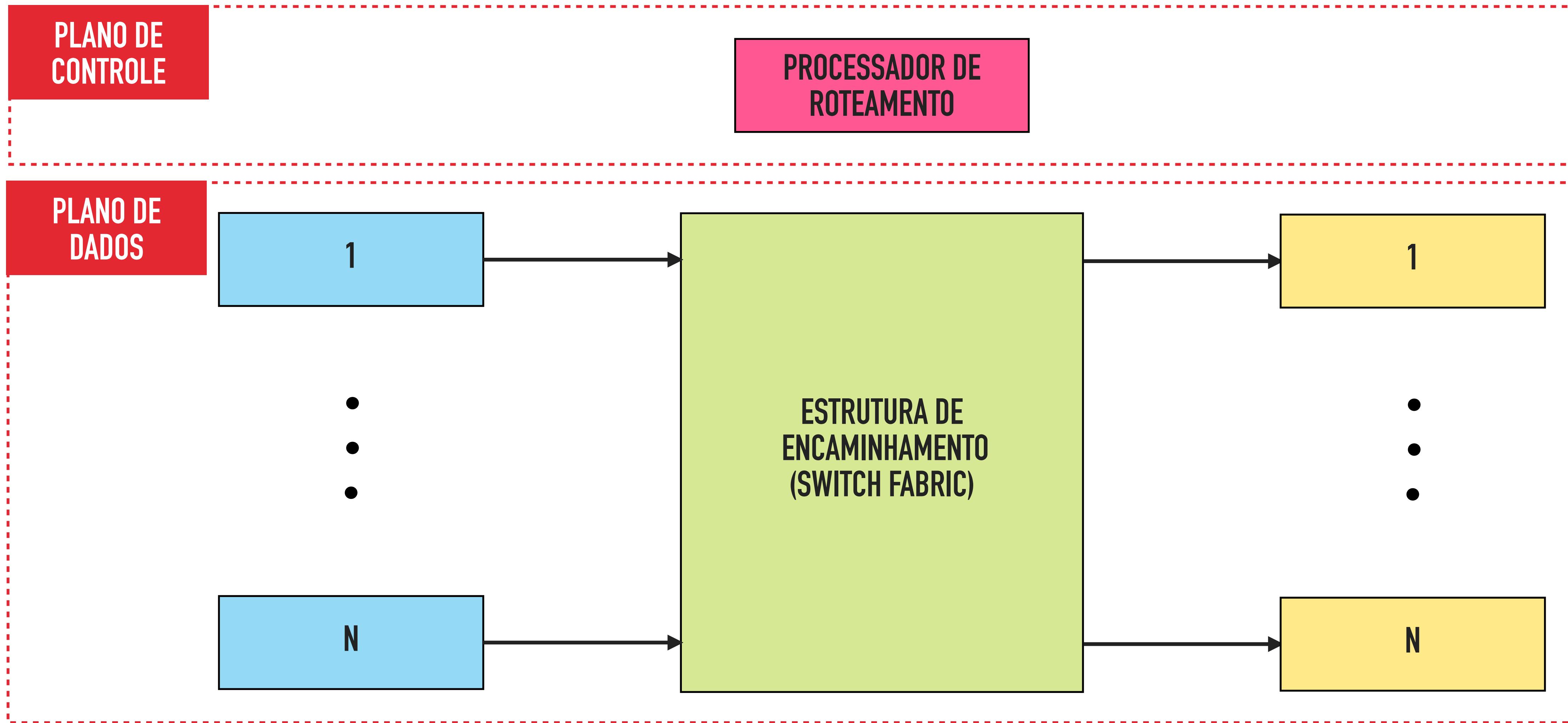
DENTRO DE UM ROTEADOR



DENTRO DE UM ROTEADOR



DENTRO DE UM ROTEADOR



TAREFAS DAS LINHAS DE ENTRADA

- ▶ Receber pacotes que chegam ao roteador (camada física)
- ▶ Atualizar o cabeçalho IP
 - ▶ TTL, checksum (talvez outros campos)
- ▶ Procurar a porta de saída para o IP de destino
- ▶ Enfileirar o pacote na *switch fabric*

DESAFIO: VELOCIDADE!

- ▶ Pacotes de 100 bytes a 40 Gbps = um pacote a cada 20ns
- ▶ Normalmente implementados em *hardware* específico/especializado
 - ▶ ASICs

PROCURANDO A PORTA DE SAÍDA

Ao receber o pacote:

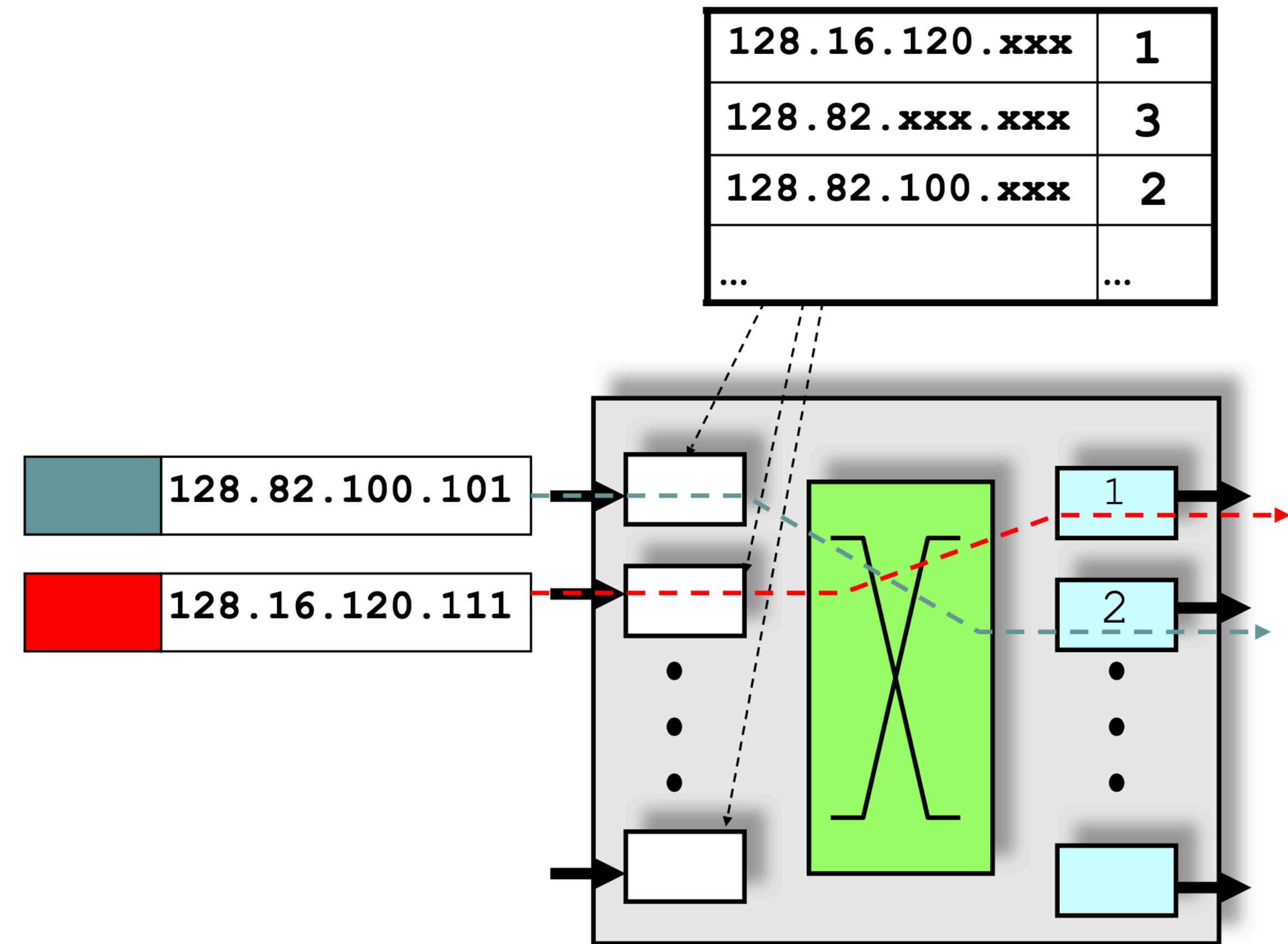
- ▶ Inspecionar o endereço IP de destino no cabeçalho
- ▶ Buscar na tabela de roteamento
- ▶ Se não encontrar uma correspondência (*match*), selecionar a rota padrão (*default*)
- ▶ Encaminhar o pacote para a interface apropriada

Rota padrão (*default*):

- ▶ Configurada para cobrir casos onde não há correspondências
- ▶ Permite diminuir as tabelas de roteamento nas bordas (*edges*)
- ▶ *Se não está na minha rede, envia para o meu ISP!*

ESCALABILIDADE NA BUSCA

- ▶ Relembrando: para escalabilidade, endereços são **agregados**
- ▶ Correspondência do prefixo mais longo (*longest prefix match*)
- ▶ Encontrar a linha que corresponde ao prefixo mais longo com aquele endereço de destino



ENCONTRANDO UMA CORRESPONDÊNCIA

Destino do pacote que chegou: 201.143.7.0

TABELA DE ROTEAMENTO

Prefixo	Porta
201.143.0.0/22	1
201.143.4.0/24	2
201.143.5.0/24	3
201.143.6.0/23	4

ENCONTRANDO UMA CORRESPONDÊNCIA: CONVERTENDO PARA BINÁRIO

Destino do pacote que chegou: **201.143.7.0**

11001001 10001111 00000111 11010010

TABELA DE ROTEAMENTO
201.143.0.0/22
11001001 10001111 000000-- -----
201.143.4.0/24
11001001 10001111 00000100 -----
201.143.5.0/24
11001001 10001111 00000101 -----
201.143.6.0/23
11001001 10001111 0000011- -----

ENCONTRANDO UMA CORRESPONDÊNCIA: CONVERTENDO PARA BINÁRIO

Destino do pacote que chegou: **201.143.7.0**

11001001 10001111 00000111 11010010

TABELA DE ROTEAMENTO
201.143.0.0/22
11001001 10001111 000000-- -----
201.143.4.0/24
11001001 10001111 00000100 -----
201.143.5.0/24
11001001 10001111 00000101 -----
201.143.6.0/23
11001001 10001111 0000011- -----

ENCONTRANDO UMA CORRESPONDÊNCIA: CONVERTENDO PARA BINÁRIO

Destino do pacote que chegou: **201.143.7.0**

11001001 10001111 00000**1**11 11010010

TABELA DE ROTEAMENTO
201.143.0.0/22
11001001 10001111 00000 0 -- ----- X
201.143.4.0/24
11001001 10001111 00000 1 00 -----
201.143.5.0/24
11001001 10001111 00000 1 01 -----
201.143.6.0/23
11001001 10001111 00000 1 1-----

ENCONTRANDO UMA CORRESPONDÊNCIA: CONVERTENDO PARA BINÁRIO

Destino do pacote que chegou: **201.143.7.0**

11001001 10001111 00000**111** 11010010

Verifica um endereço contra todos os prefixos de destino e seleciona o prefixo que combina com a maioria dos *bits*

TABELA DE ROTEAMENTO	Destino			Ação
	Prefixo	Mask	Next Hop	
201.143.0.0/22	11001001 10001111 00000 0 --	-----		X
201.143.4.0/24	11001001 10001111 00000 100	-----		X
201.143.5.0/24	11001001 10001111 00000 101	-----		X
201.143.6.0/23	11001001 10001111 00000 11-	-----		✓

ENCONTRANDO UMA CORRESPONDÊNCIA DE MANEIRA EFICIENTE

- ▶ Testando cada linha para encontrar uma correspondência escala de maneira não eficiente
 - ▶ Basicamente ($\text{número de linhas} \times \text{número de bits}$)
- ▶ Podemos aproveitar estruturas de dados como árvores
 - ▶ Árvore de prefixos (*trie*)

CONSIDERE QUATRO PREFIXOS DE 3 BITS

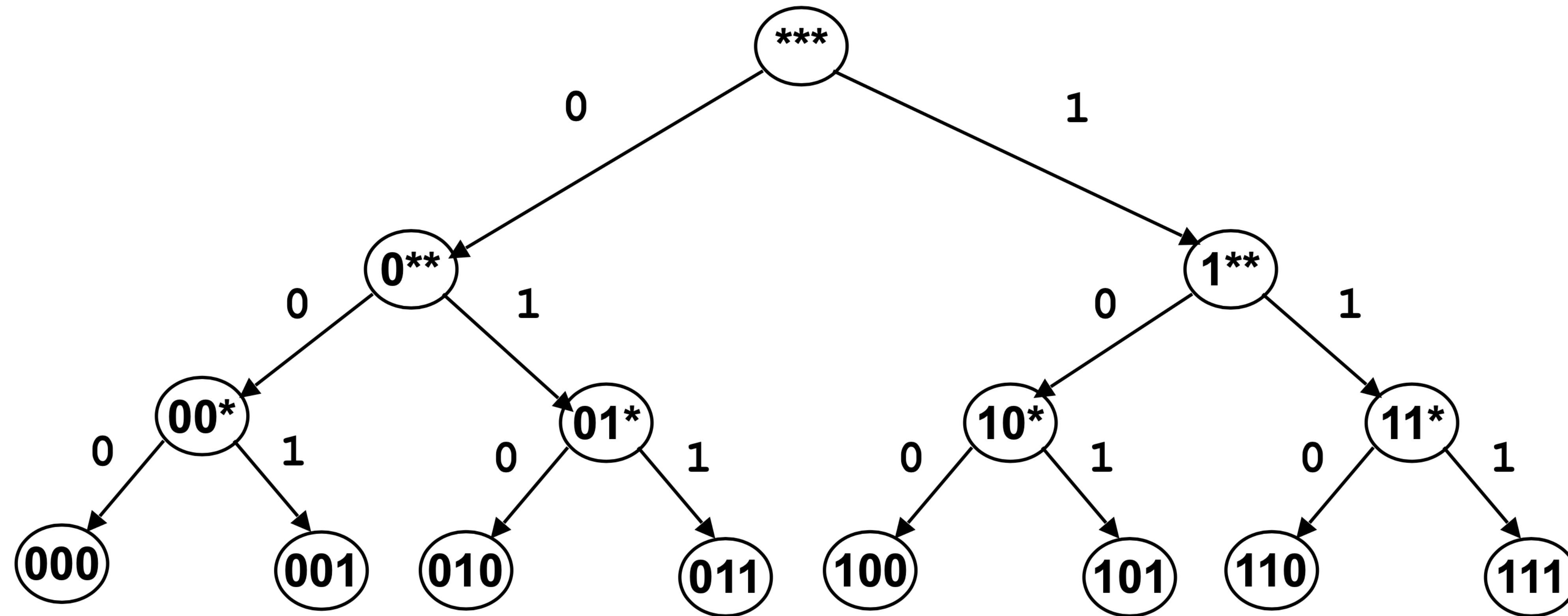
$0^{**} \rightarrow$ Porta 1

$100 \rightarrow$ Porta 2

$101 \rightarrow$ Porta 3

$11^* \rightarrow$ Porta 4

ESTRUTURA TRIE



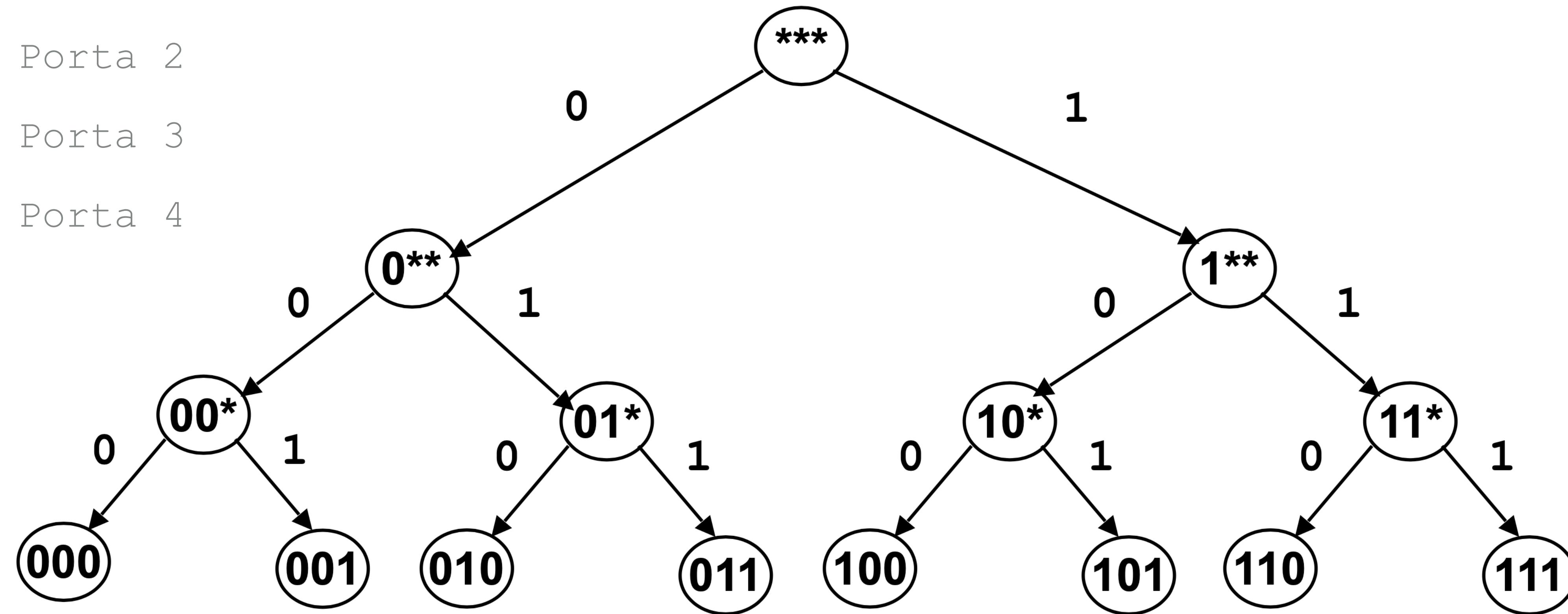
ESTRUTURA TRIE

$0^{**} \rightarrow$ Porta 1

$100 \rightarrow$ Porta 2

$101 \rightarrow$ Porta 3

$11^* \rightarrow$ Porta 4



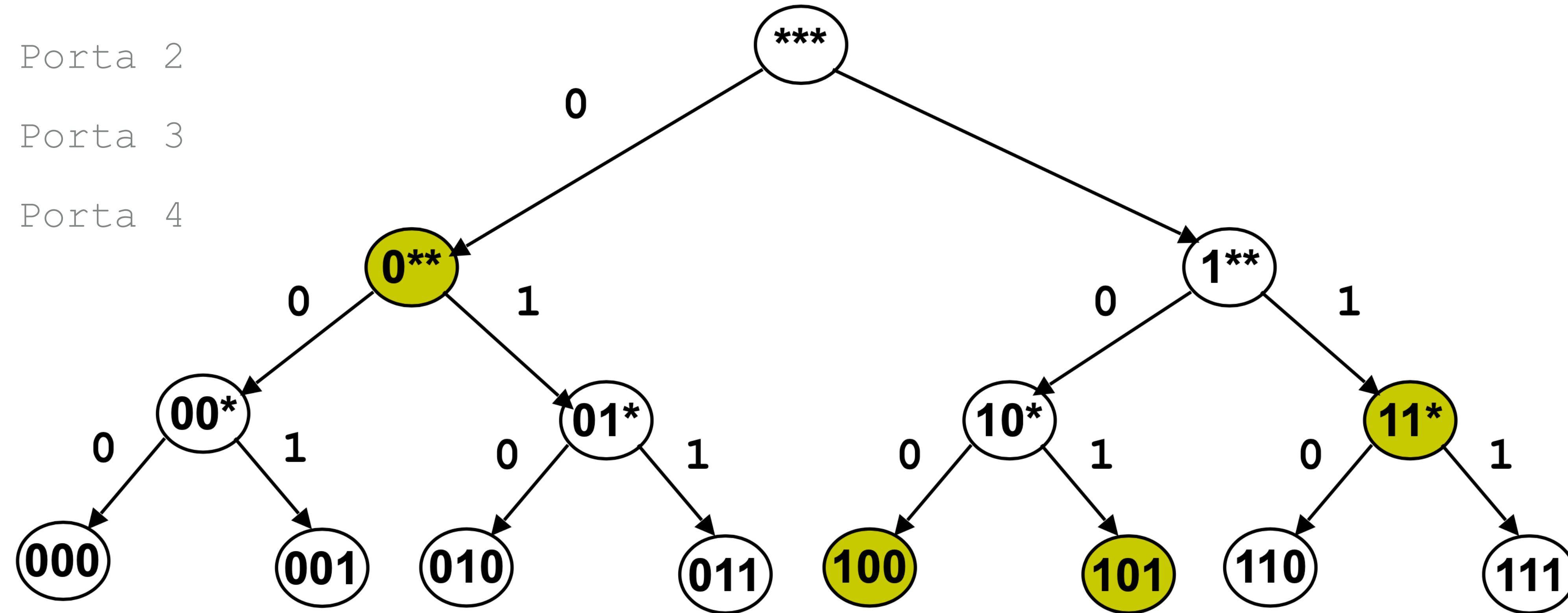
TRIE CORRESPONDENTE: PARAR NOS PREFIXOS

$0^{**} \rightarrow \text{Porta } 1$

$100 \rightarrow \text{Porta } 2$

$101 \rightarrow \text{Porta } 3$

$11^* \rightarrow \text{Porta } 4$



CORRESPONDÊNCIA DO PREFIXO MAIS LONGO EM ROTEADORES REAIS

- ▶ Roteadores reais utilizam soluções mais complexas e avançadas
 - ▶ Mas o discutido é o ponto de partida!
- ▶ Roteadores utilizam otimizações e heurísticas com padrões do mundo real
 - ▶ Alguns destinos são mais populares que outros
 - ▶ Algumas portas levam a mais destinos

REVISANDO: TAREFAS DAS LINHAS DE ENTRADA

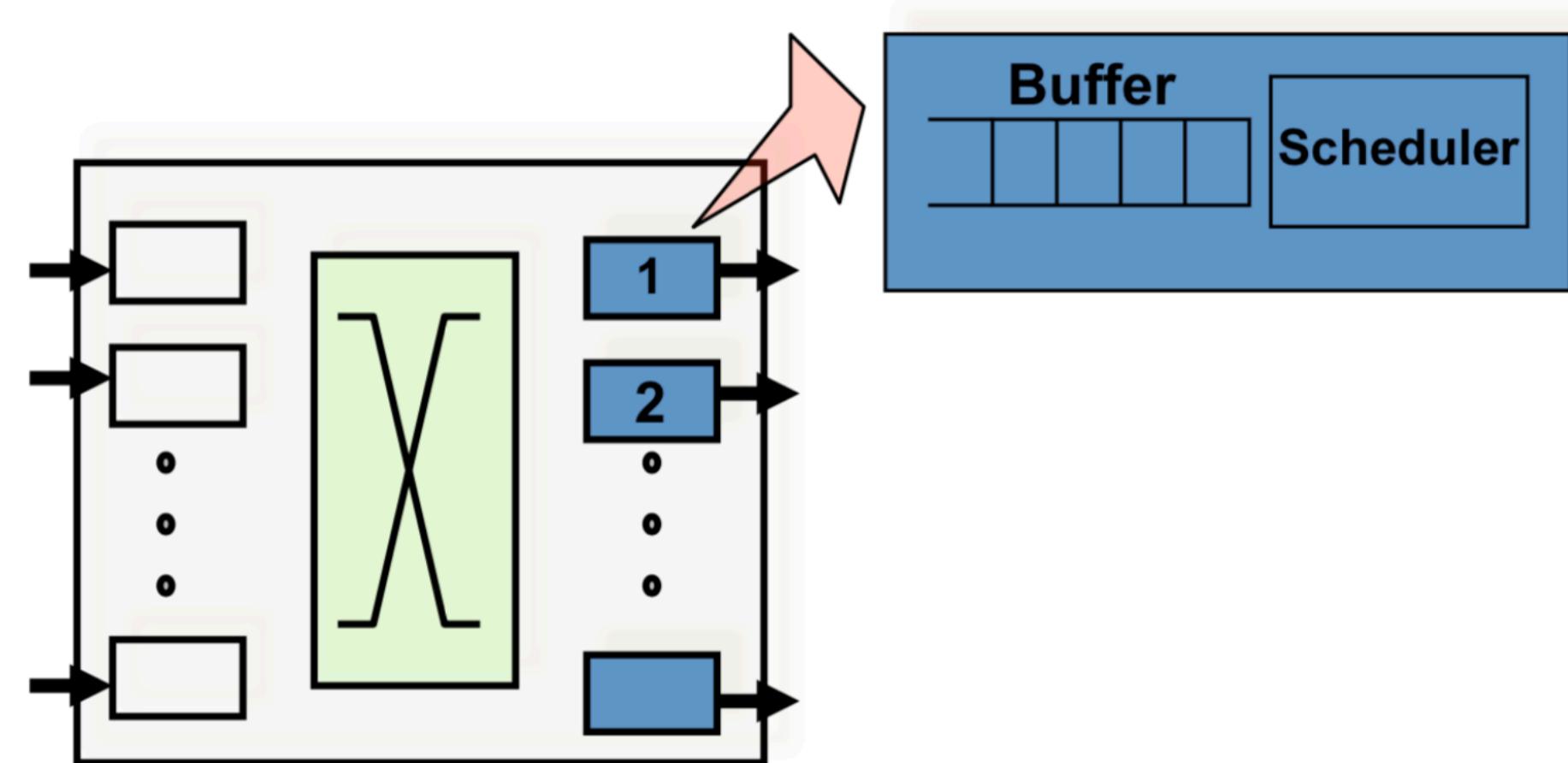
- ▶ Atualizar o cabeçalho IP (fácil)
- ▶ Correspondência do prefixo mais longo com o endereço de destino (difícil)
- ▶ Predominantemente implementado em *hardware* especializado

TAREFAS DAS LINHAS DE SAÍDA

- ▶ **Classificação do pacote:** mapeia cada pacote a um “fluxo”
- ▶ **Gerenciamento de *buffer*:** decide quando e qual pacote será descartado
- ▶ **Escalonador (*scheduler*):** decide quando e qual pacote será transmitido

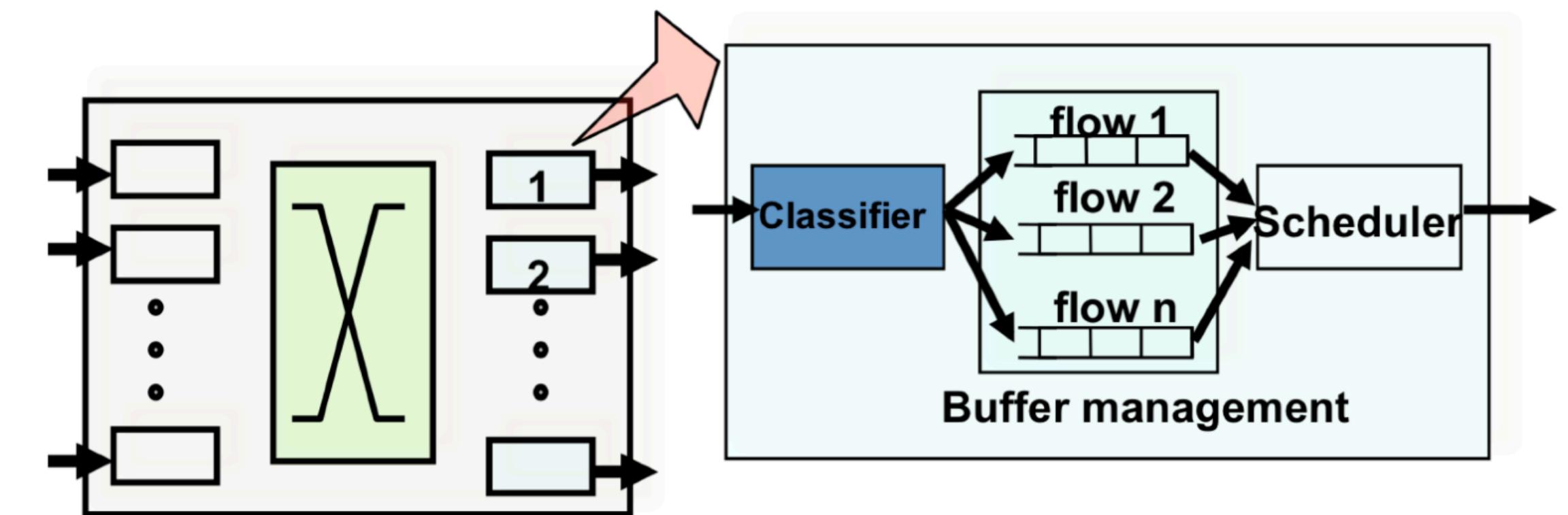
ROTEADOR FIFO - FIRST IN FIRST OUT

- ▶ Sem classificação
- ▶ Descarta (dropa) pacotes do que chegam quando o *buffer* está cheio
- ▶ Escalonador FIFO: escalona pacotes pela ordem de chegada



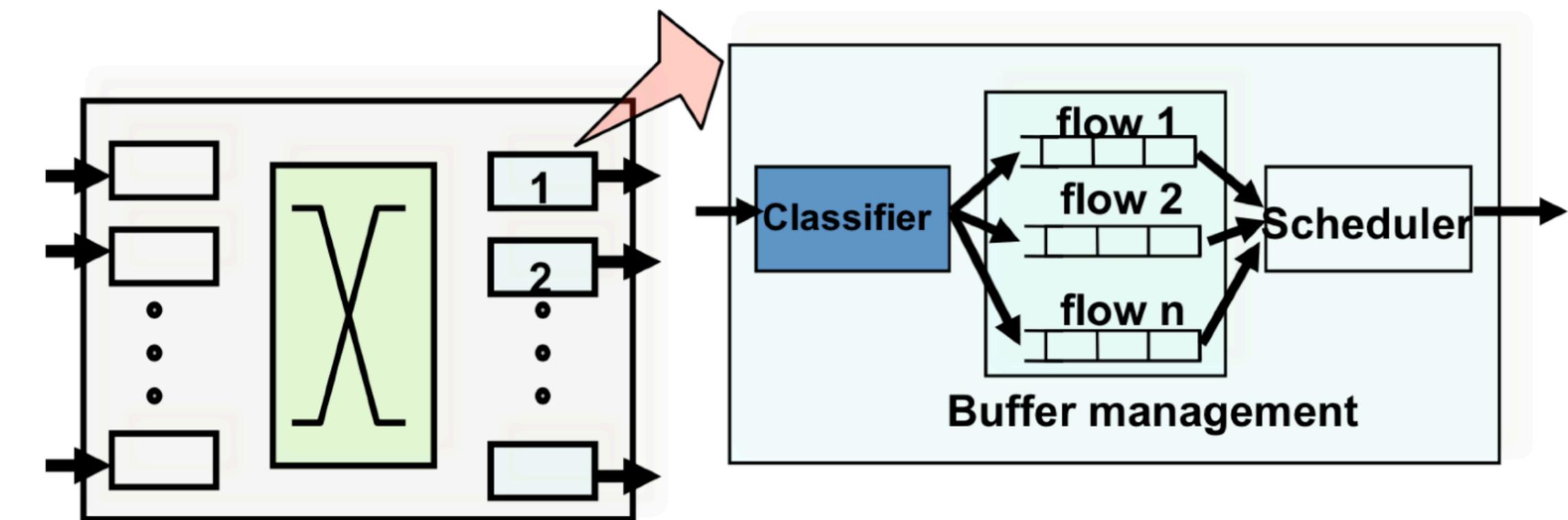
CLASSIFICAÇÃO DO PACOTE

- ▶ Baseado no valor de campos do cabeçalho:
 - ▶ Endereço IP de origem/destino
 - ▶ Porta TCP de origem/destino
 - ▶ Type of Service (ToS)
 - ▶ Protocolo



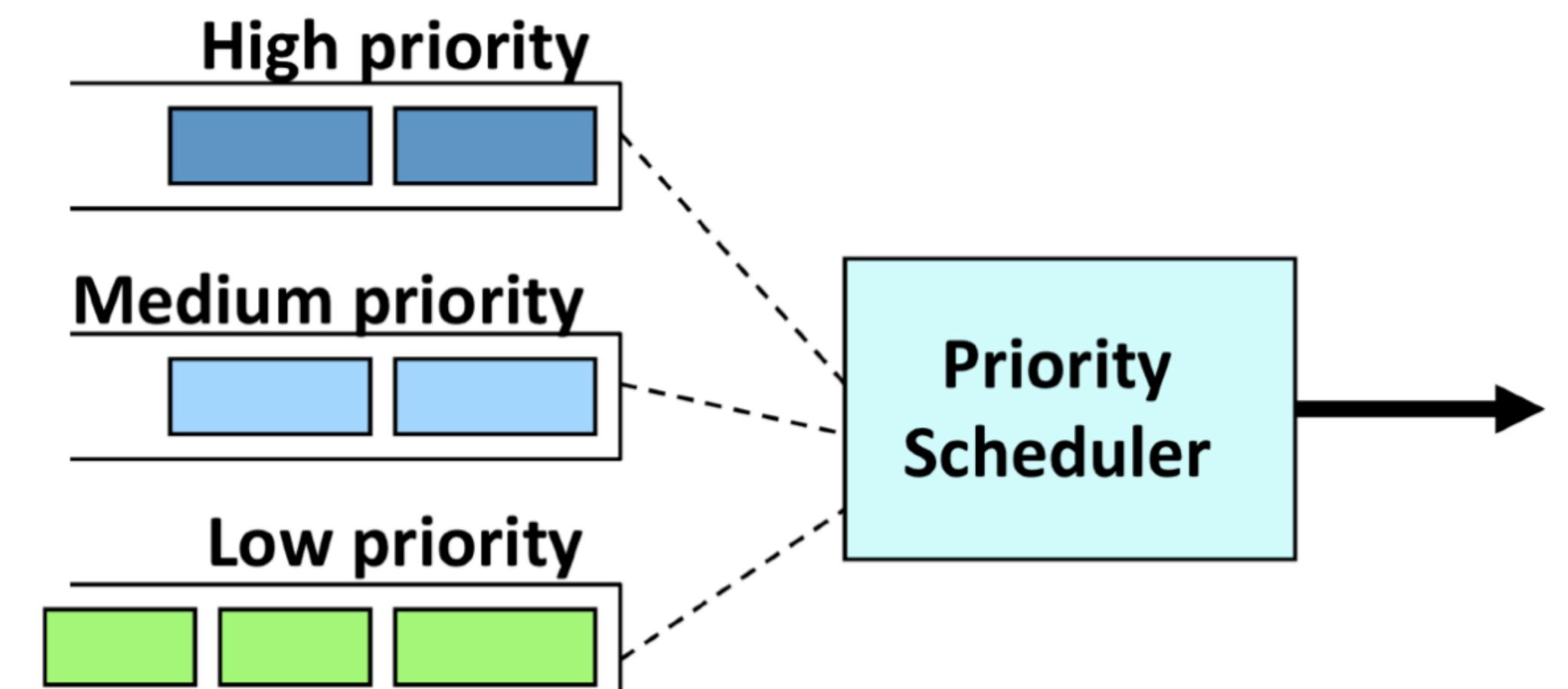
ESCALONADOR

- ▶ Uma fila por fluxo
- ▶ Escalonador decide a partir de qual fila será enviado um pacote
- ▶ Objetivos do algoritmo de escalonamento:
 - ▶ Rapidez
 - ▶ Dependência da política implementada (equidade, prioridade, etc.)



EXEMPLO: ESCALONAMENTO POR PRIORIDADE

- ▶ Pacotes com maior prioridade sempre são servidos antes de pacotes com prioridades menores



EXEMPLO: ESCALONAMENTO ROUND-ROBIN

- ▶ Pacotes são servidos um de cada fila por vez

