

Aula 13 - Políticas de Enfileiramento, Protocolo IP

Diego Passos

Universidade Federal Fluminense

Redes de Computadores

Material adaptado a partir dos slides
originais de J.F Kurose and K.W. Ross.

Roteadores: Políticas de Enfileiramento

Políticas de Enfileiramento: Escalonamento e Descarte

- Também chamadas de **disciplinas**.
- Duas decisões importantes:
 - **Escalonamento**: em que ordem transmitir os pacotes.
 - *e.g.*, há pacotes mais importantes que outros?
 - *e.g.*, um pacote deve poder “passar a frente” dos demais?
 - Prioridades?
 - **Descarte**: quem (e quando) descartar.
 - Descartar o último pacote?
 - Descartar o primeiro?
 - Descartar aleatoriamente?
 - Descartar apenas quando fila está **completamente cheia**?

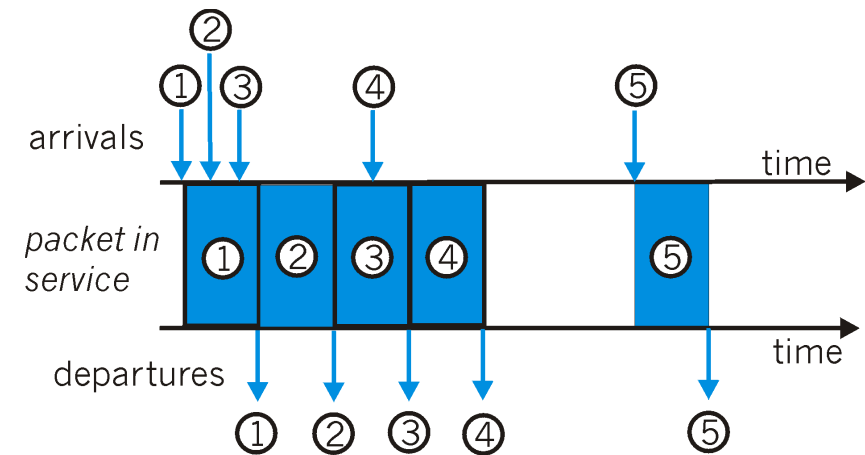
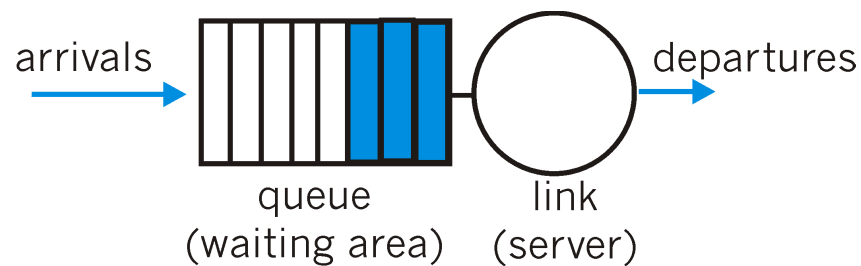
Políticas de Enfileiramento: Por Quê?

- **Solução óbvia:** respeitar ordem de chegada.
 - Transmitir pacotes na ordem em que são recebidos.
 - Descartar novos pacotes quando não há espaço em *buffer*.
- **Nem sempre é o ideal:**
 - Podemos querer **dar importância maior** a certos pacotes/fluxos.
 - *e.g.*, datagrama de um *download* pode esperar, de uma chamada VoIP não.
 - **Controle de congestionamento do TCP pode ser afetado.**
 - Infere congestionamento por perdas.
 - É possível **avisar mais rápido e a todos os fluxos**.

Políticas de Escalonamento: FIFO

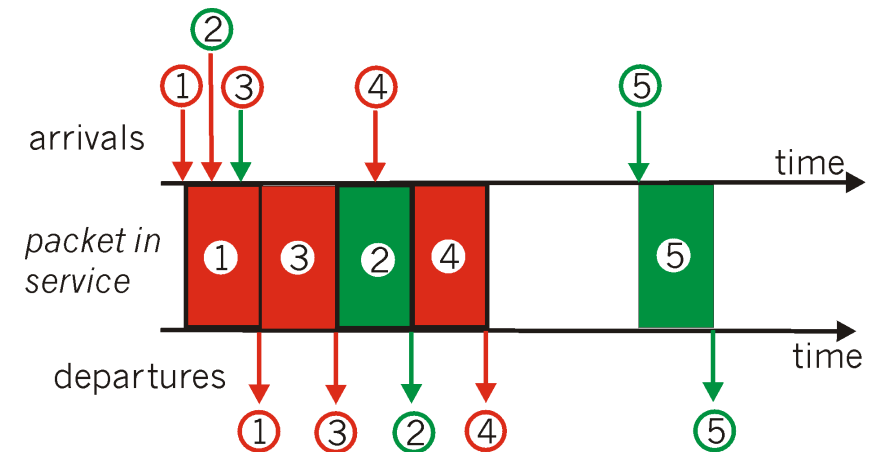
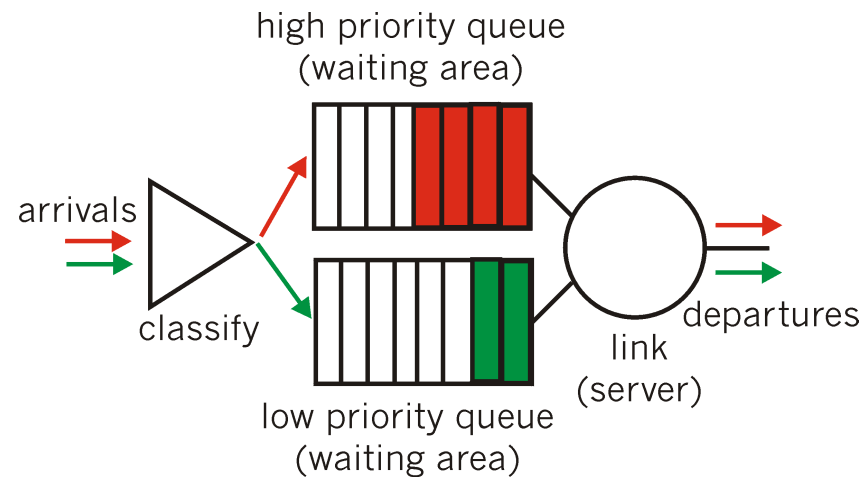
- **First-In, First-Out.**

- Também chamada de FCFS (*First-Come, First-Served*).
- “Solução óbvia”: pacotes servidos na ordem em que chegam.
- Não há prioridades.
 - Nenhum pacote “fura fila”.
- Simples, popular.



Políticas de Escalonamento: *Priority Queuing* (I)

- Divide pacotes em **classes**.
 - Uma fila separada para cada classe.
- Cada classe possui uma **prioridade diferente**.
- Pacotes de classes prioritárias **sempre** são transmitidos antes.
 - i.e., fila de uma classe só é atendida se filas de **todas as classes mais prioritárias estão vazias**.



Políticas de Escalonamento: *Priority Queuing* (II)

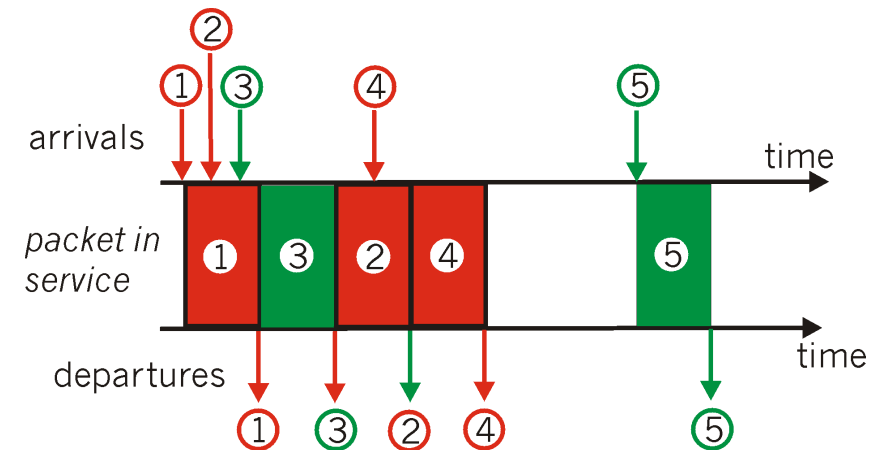
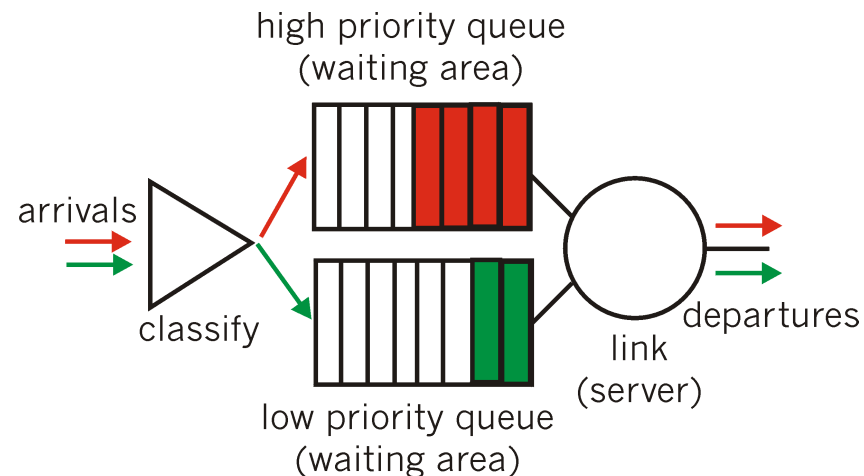
- Classe do pacote é definida de acordo com sua importância.
 - Termo relativo, definido pelo administrador do roteador.
- **Exemplo:**
 - Pacotes VoIP ficam na classe de mais alta prioridade.
 - Demais pacotes ficam na classe de prioridade mais baixa.
- **Problemas?**

Políticas de Escalonamento: *Priority Queuing* (III)

- Classe do pacote é definida de acordo com sua importância.
 - Termo relativo, definido pelo administrador do roteador.
- **Exemplo:**
 - Pacotes VoIP ficam na classe de mais alta prioridade.
 - Demais pacotes ficam na classe de prioridade mais baixa.
- **Problemas?**
 - Pode causar **esfomeação** (*starvation*).

Políticas de Escalonamento: Round-Robin

- Assim como a *Priority Queuing*, pacotes são divididos em classes.
- **Mas objetivo é diferente:**
 - Garantir uma divisão **justa** do uso dos recursos.
 - Todas as classes recebem **oportunidades iguais**.
- Filas das classes são servidas através de um *round-robin*, um pacote por vez.

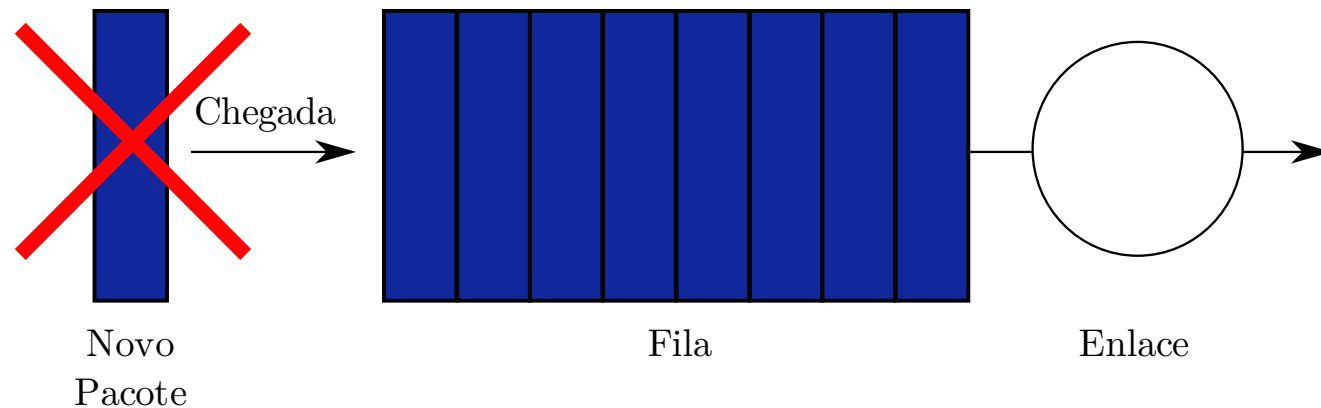


Políticas de Escalonamento: Round-Robin e Prioridades?

- Round-Robin não incorre em esfomeação.
- **Mas também não atribui prioridades diferentes!**
- É possível **combinar** as duas abordagens?
 - i.e., atribuir prioridades maiores a certas classes, **mas garantindo que todas as classes receberão** um certo grau de oportunidade?
- **Sim!**
 - Alcançado pela política *Weighted Fair Queueing*.
 - Mais detalhes em Redes II.

Políticas de Descarte: *Drop-tail* (I)

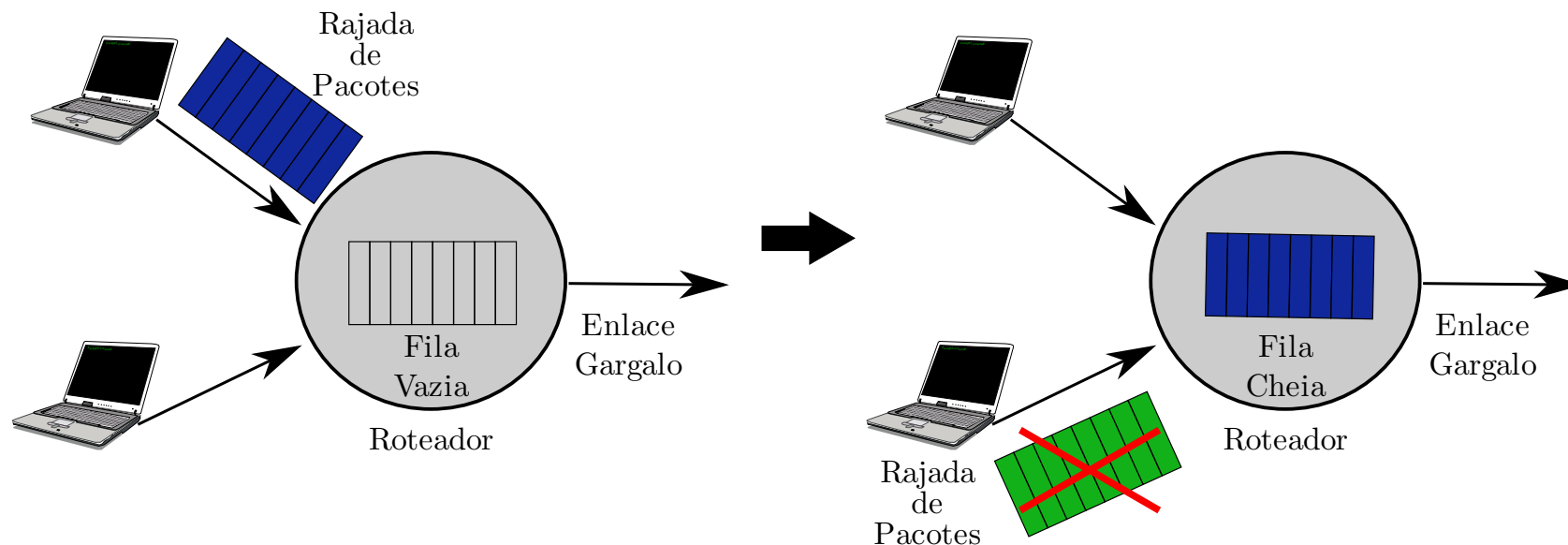
- Está para o descarte como a FIFO está para o escalonamento.
 - Ideia simples, imediata.
- Funcionamento:
 - Quando fila está cheia e novo pacote chega, **novo pacote sempre é descartado.**



- Política amplamente implementada e adotada.
 - Mas não necessariamente a melhor em todos os casos.

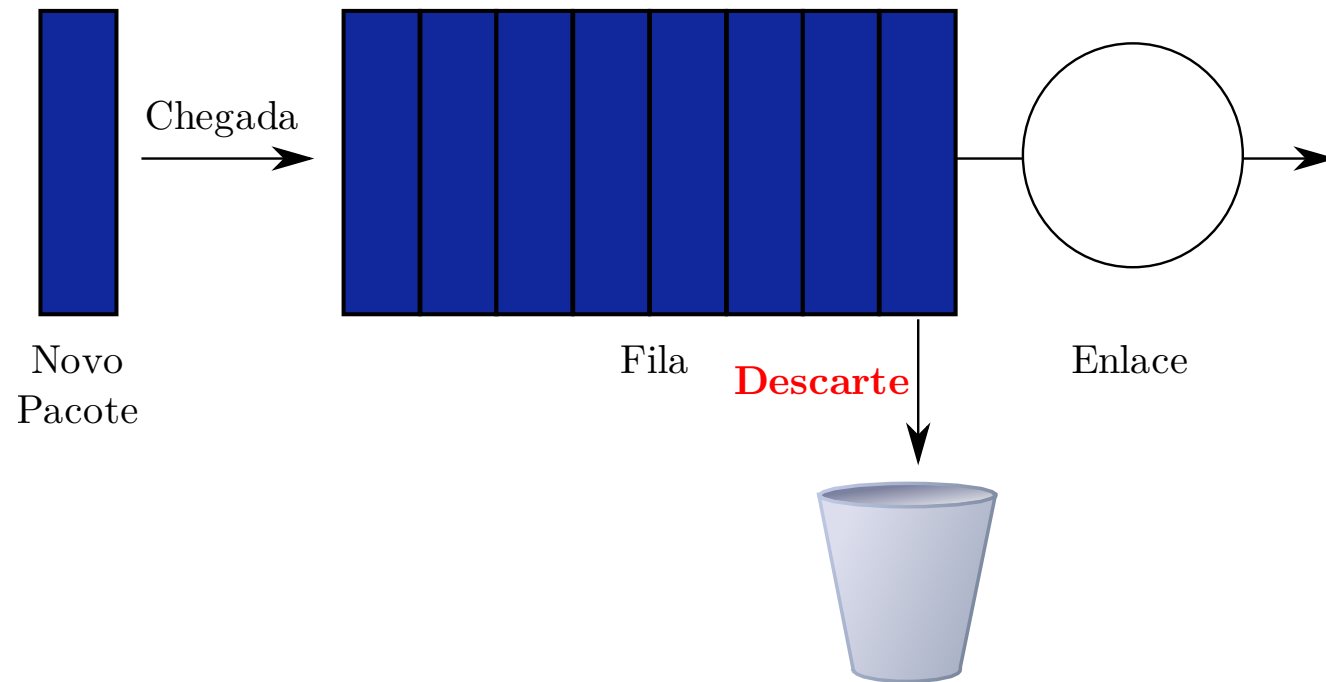
Políticas de Descarte: *Drop-tail* (II)

- Potencial problema: sincronização de fluxos.
 - Suponha dois *hosts* compartilhando um único enlace de saída de um roteador.
 - Assuma que ambos geram **tráfego em rajada**.
 - i.e., quando *host* transmite, envia **vários pacotes em sequência**.
 - Em seguida, passa algum sem geração de tráfego.
 - Dependendo da ordem dos envios, um dos *hosts* pode ser prejudicado.
 - i.e., seus pacotes comumente encontram a fila cheia e são descartados.



Políticas de Descarte: *Drop-head* (I)

- “Contrário” da *drop-tail*.
- Se novo pacote chega e fila está cheia, **primeiro pacote da fila é sempre descartado**.
 - *i.e.*, o pacote a mais tempo na fila.



- **Pergunta:** resolve o problema da injustiça na *Drop-tail*?

Políticas de Descarte: *Drop-head* (II)

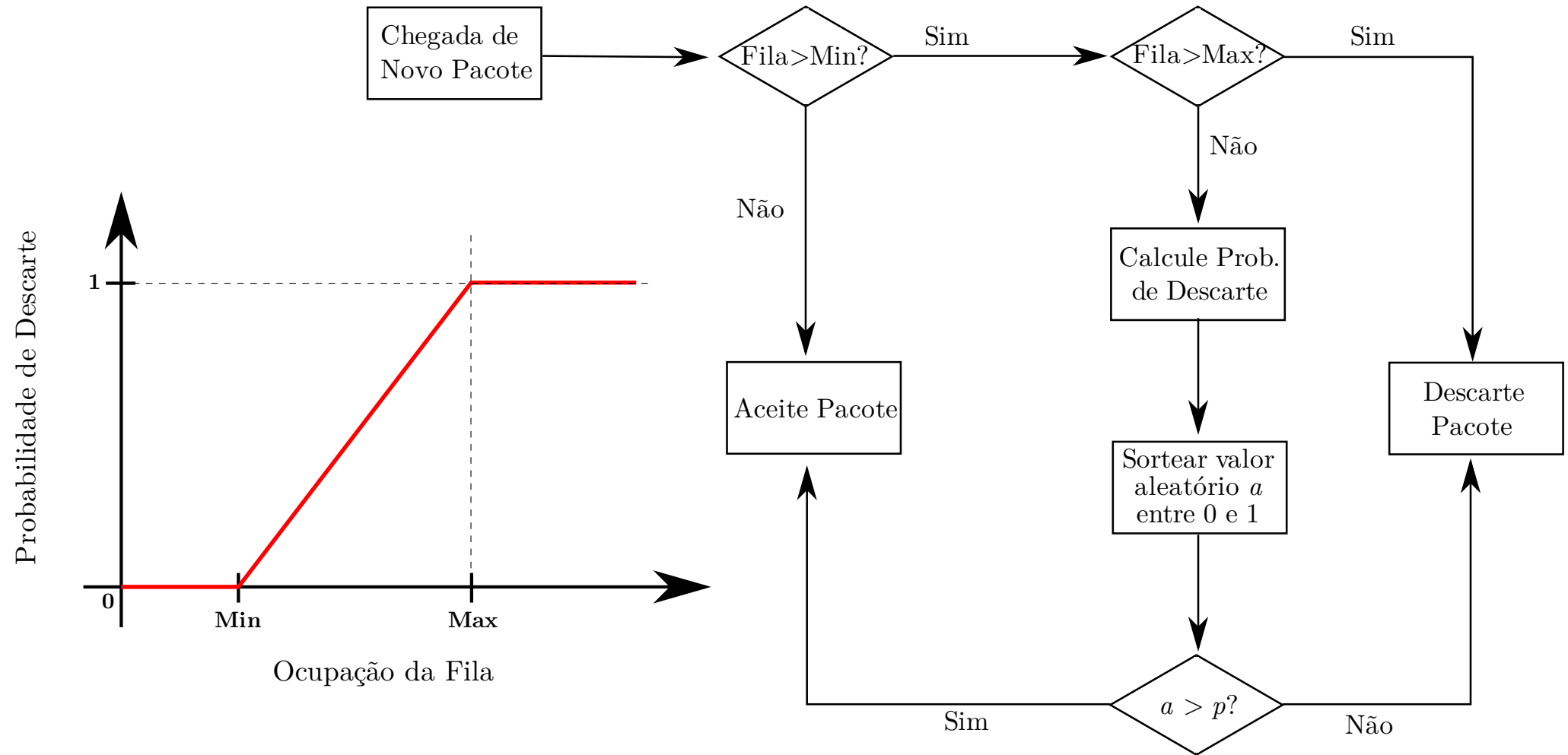
- Qual é a vantagem de se descartar o primeiro pacote da fila?
- Não é injusto descartar o pacote que espera há mais tempo?
- Talvez, mas se este pacote for um segmento TCP, há uma grande vantagem:
 - TCP **precisa ser avisado do congestionamento o mais rápido possível.**
 - Descartar primeiro pacote provavelmente gerará mais rapidamente:
 - Estouro do temporizador do TCP.
 - Ou ACKs duplicados.
 - **Resultado:** TCP reage mais rapidamente reduzindo janela.

Políticas de Descarte: RED (I)

- **Random Early Detection.**
- Começa (possivelmente) a descartar pacotes **antes que a fila esteja completamente cheia.**
- Funcionamento (Simplificado):
 - **Mínimo:** menor ocupação da fila para a qual pacotes podem ser descartados.
 - **Máximo:** tamanho máximo da fila.
 - Quando novo pacote chega:
 - Se fila está menor que Mínimo, nunca descarte.
 - Se fila está igual a Máximo, sempre descarte.
 - Caso contrário, **descarte último pacote com probabilidade p proporcional ao tamanho atual da fila.**

Políticas de Descarte: RED (II)

- Esquema simplificado de funcionamento do RED:



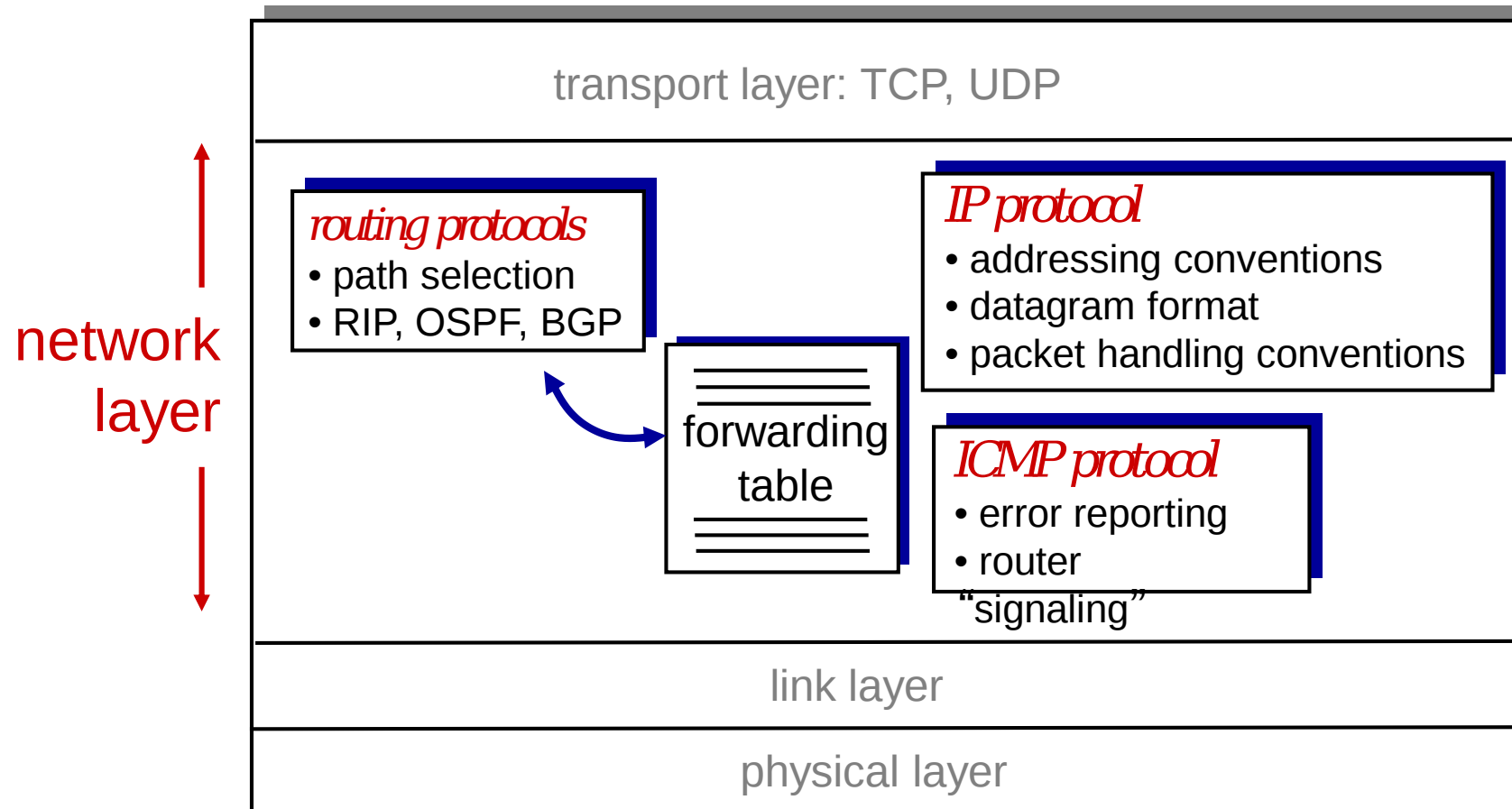
Políticas de Descarte: RED (III)

- Mar **por que descartar** pacotes se o *buffer* ainda não está totalmente cheio?
- Lembre-se:
 - Congestionamento se manifesta como um **aumento no nível de enfileiramento**.
 - *Overflow* é apenas uma **consequência** após um **período estendido de congestionamento**.
 - TCP identifica congestionamento por perdas.
- Ao descartar pacotes quando *buffer* está parcialmente cheio, **sinalizamos congestionamento antecipadamente**.
- **Efeito colateral:** evita problema da sincronização.
 - Descartes tendem a ser mais bem distribuídos entre fluxos.

Protocolo IP: Datagramas

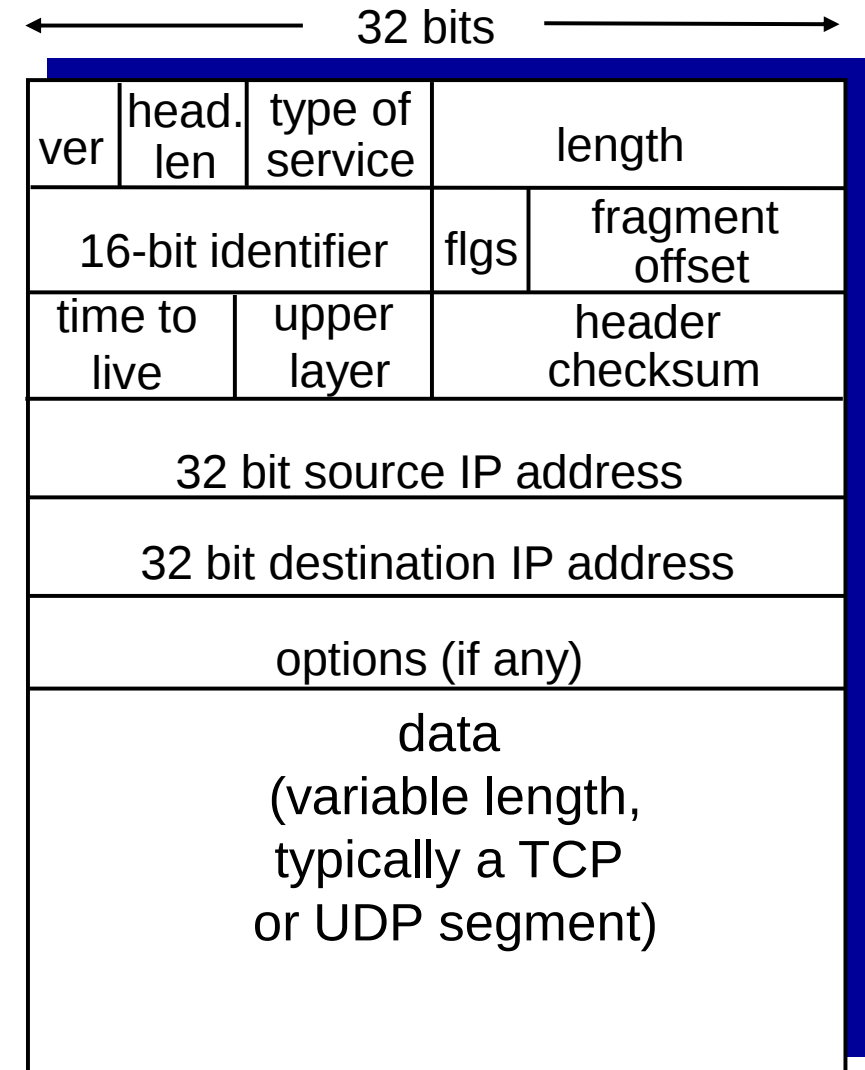
A Camada de Rede na Internet

- Funções da camada de rede de hosts e roteadores.



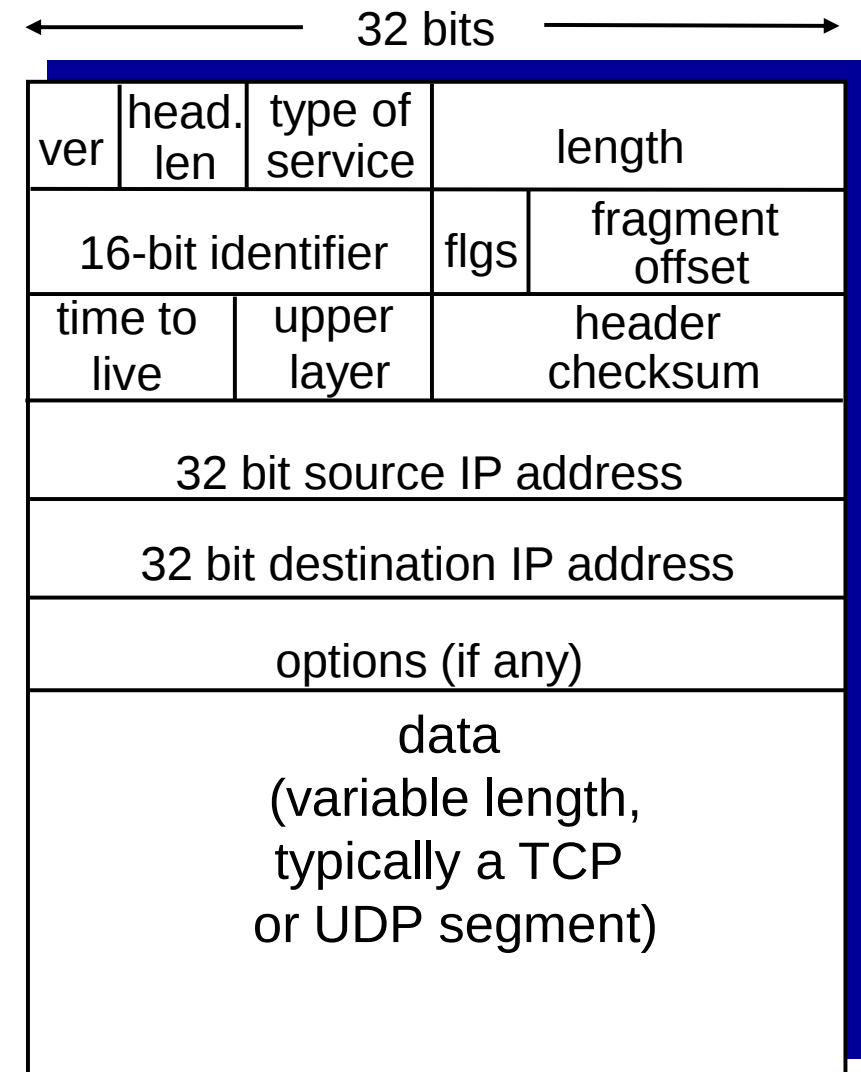
Formato do Datagrama (I)

- **version:** número de versão.
 - e.g., versão 4, versão 6.
- **header length:** comprimento do cabeçalho, em bytes.
 - Cabeçalho tem tamanho variável.
 - Vide campo **options**.
- **type of service:** “classe” do dado encapsulado.
 - e.g., tráfego de tempo real, melhor esforço.
- **length:** tamanho total do datagrama.
 - Cabeçalho + carga útil.
 - Tamanho máximo: 65535 bytes.
- **identifier, flags, fragment offset:** usados para fragmentação.
 - Mais em breve.



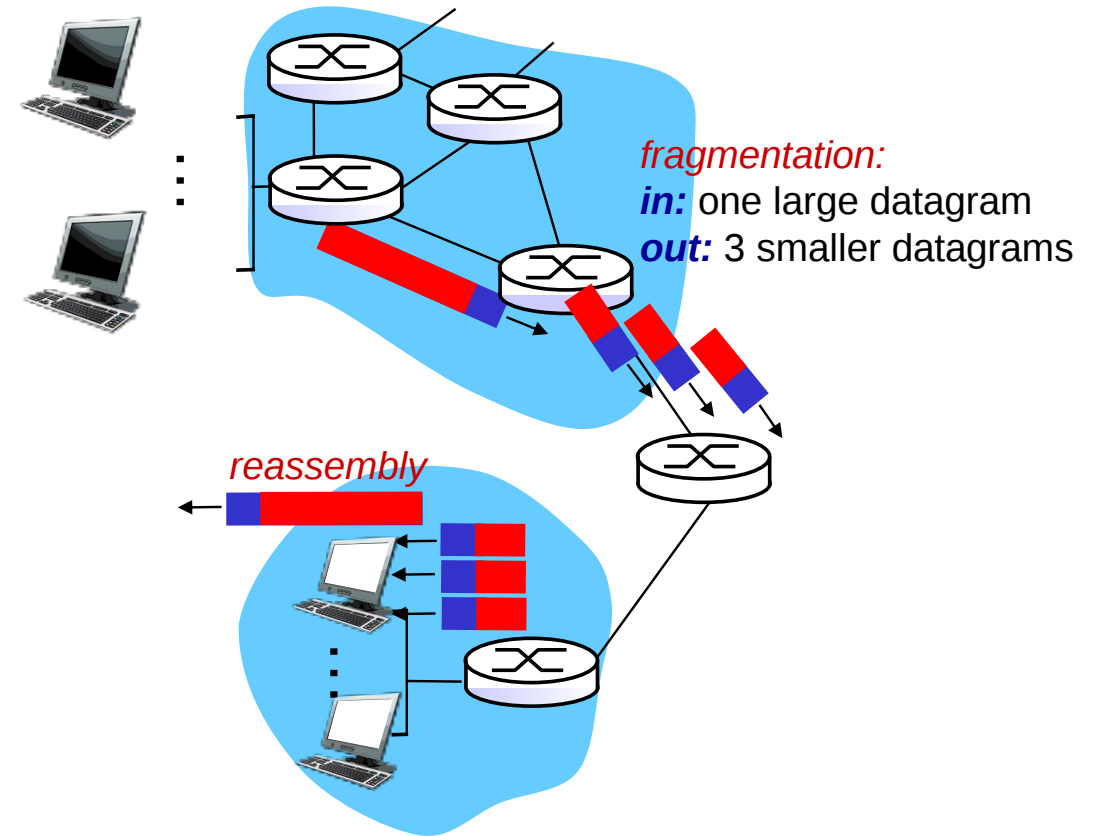
Formato do Datagrama (II)

- **time to live:** número máximo de saltos que datagrama pode percorrer.
 - Decrementado a cada roteador.
 - Usado, por exemplo, em caso de *loops* de roteamento.
- **upper layer:** indica protocolo responsável pela carga útil.
 - *e.g.*, TCP, UDP, IP.
- **header checksum:** verificação de integridade **apenas do cabeçalho**.
- **endereços de origem e destino:** 32 bits cada.
- **options:** opções de tratamento do datagrama.
 - *e.g.*, *timestamp*, gravar rota, especificação de caminho.



IP: Fragmentação, Remontagem de Datagramas (I)


- Enlaces de rede têm um MTU (*Maximum Transmission Unit*).
 - Maior quadro que pode ser transmitido pelo enlace.
 - Tecnologias diferentes têm MTUs diferentes.
- Datagramas IP grandes são divididos (“fragmentados”) na rede.
 - Um datagrama quebrado em vários datagramas.
 - “Remontados” **apenas no destinatário final**.
 - Bits do cabeçalho IP são usados para identificar e ordenar fragmentos de um mesmo datagrama original.



IP: Fragmentação, Remontagem de Datagramas (II)

- Exemplo:
 - Datagrama de 4000 bytes.
 - MTU = 1500 bytes.
- **Cuidado!** O campo *offset* da fragmentação **é contado em unidade de 8 bytes**.
- **Além disso:** MTU considera o tamanho do datagrama inteiro, **incluindo cabeçalho**.

	length	ID	fragflag	offset	
	=4000	=x	=0	=0	



	length	ID	fragflag	offset	
	=1500	=x	=1	=0	

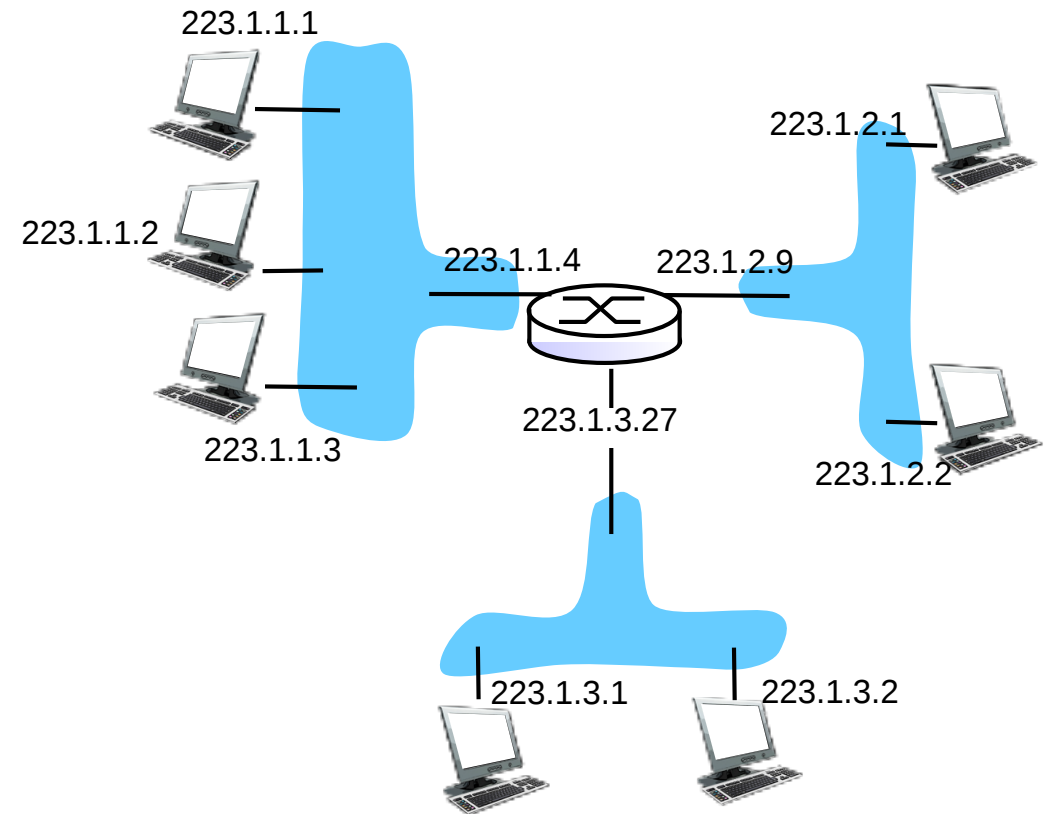
	length	ID	fragflag	offset	
	=1500	=x	=1	=185	

	length	ID	fragflag	offset	
	=1040	=x	=0	=370	

IP: Endereçamento (IPv4)

Endereçamento IP: Introdução (I)

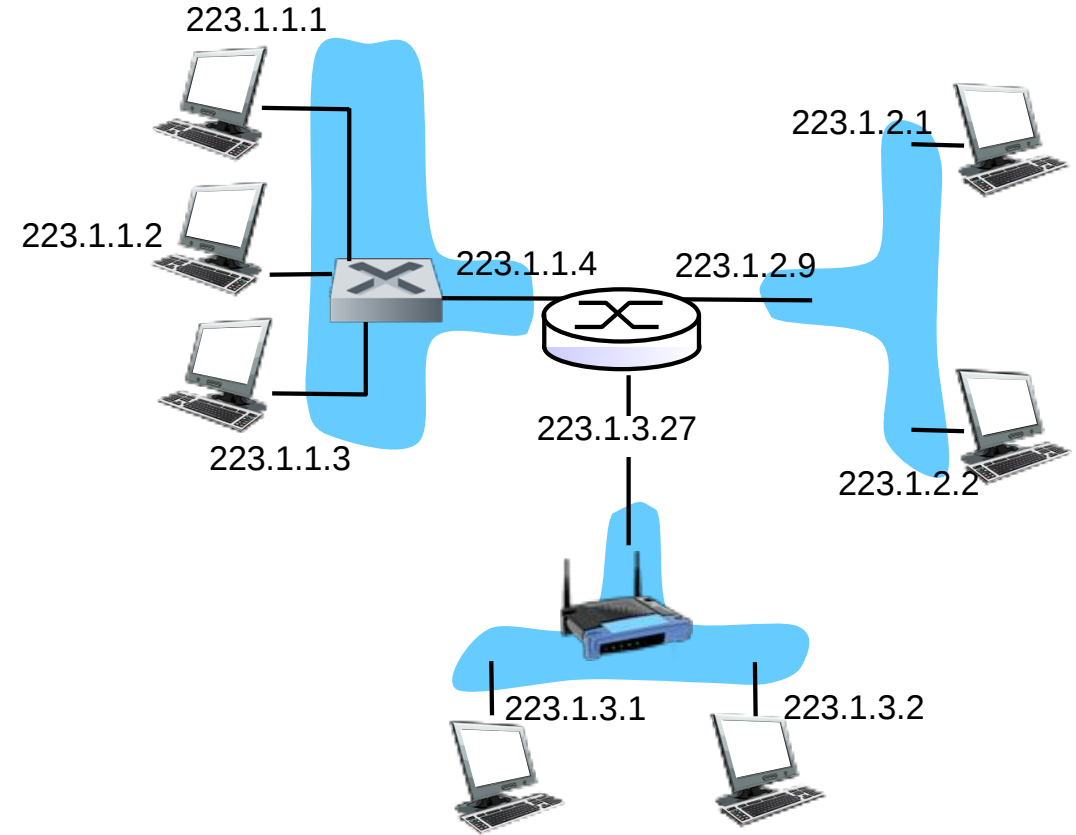
- **Endereço IP:** identificador de 32 bits para **interfaces** de *hosts*, roteadores.
- **Interface:** conexão entre *host*/roteador e enlace físico.
 - Roteadores tipicamente possuem múltiplas interfaces.
 - Host tipicamente possui uma ou duas interfaces (e.g., Ethernet cabeada e IEEE 802.11 sem fio).
- **Endereços IP associados a cada interface.**



$$223.1.1.1 = \underbrace{11011111}_{223} \underbrace{00000001}_1 \underbrace{00000001}_1 \underbrace{00000001}_1$$

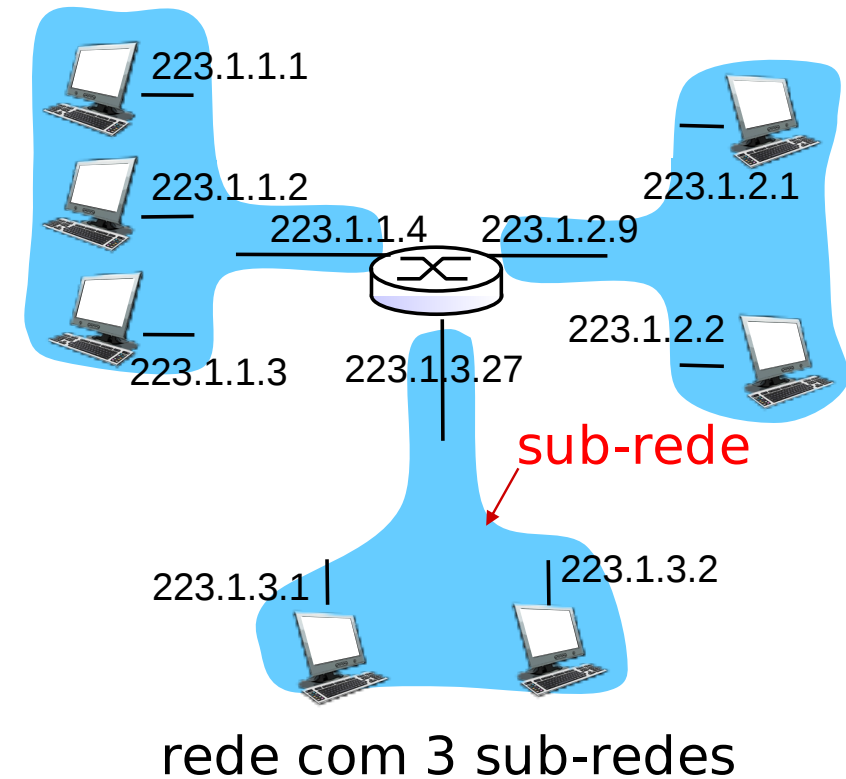
Endereçamento IP: Introdução (II)

- **Pergunta:** como as interfaces se conectam?
- **Resposta:** tópico dos capítulos 5 e 6.
 - Redes II.
- Interfaces cabeadas Ethernet se conectam através de *switches* Ethernet.
- Interfaces Wi-Fi se conectam através de estações base Wi-Fi.
- **Por enquanto:** não é preciso se preocupar com a interconexão de interfaces.



Sub-redes (I)

- **Endereço IP: duas partes.**
 - Porção da **sub-rede**: bits mais significativos (i.e., mais à esquerda).
 - Porção do *host*: bits menos significativos (i.e., mais à direita).
- **O que é uma sub-rede?**
 - Interfaces de dispositivos com a mesma porção da sub-rede nos seus endereços IP.
 - Podem se alcançar diretamente, **sem o intermédio de um roteador**.

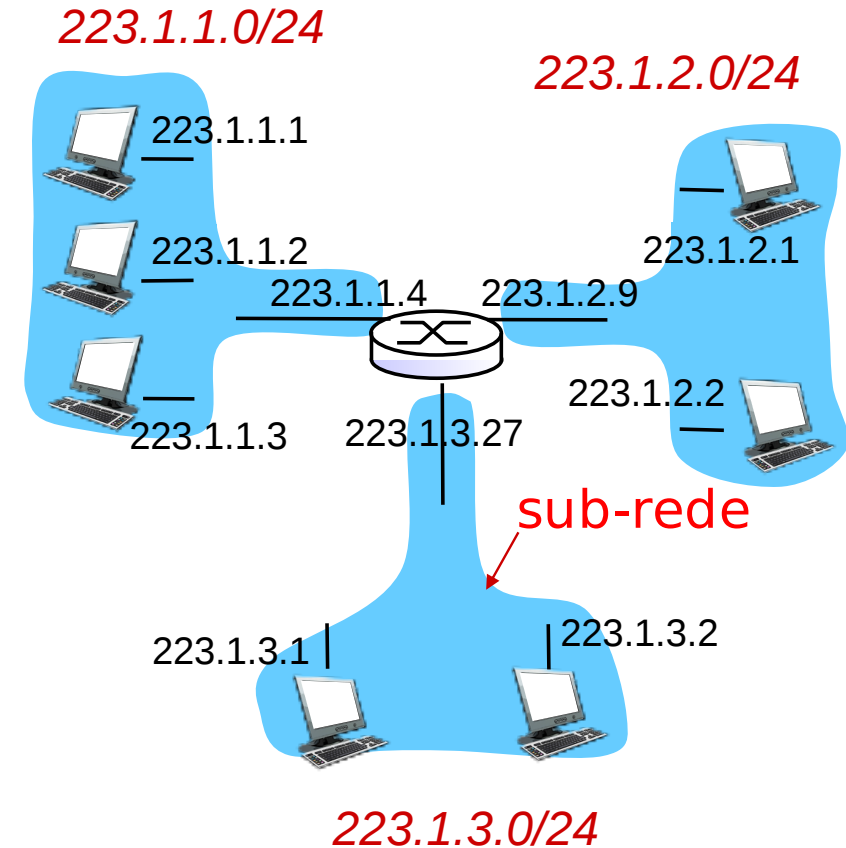


Sub-redes (II)

- **Receita:**

- Para determinar as sub-redes, desconecte as interfaces de seu *host* ou roteador, criando ilhas de redes isoladas.
- Cada rede isolada é uma **sub-rede**.

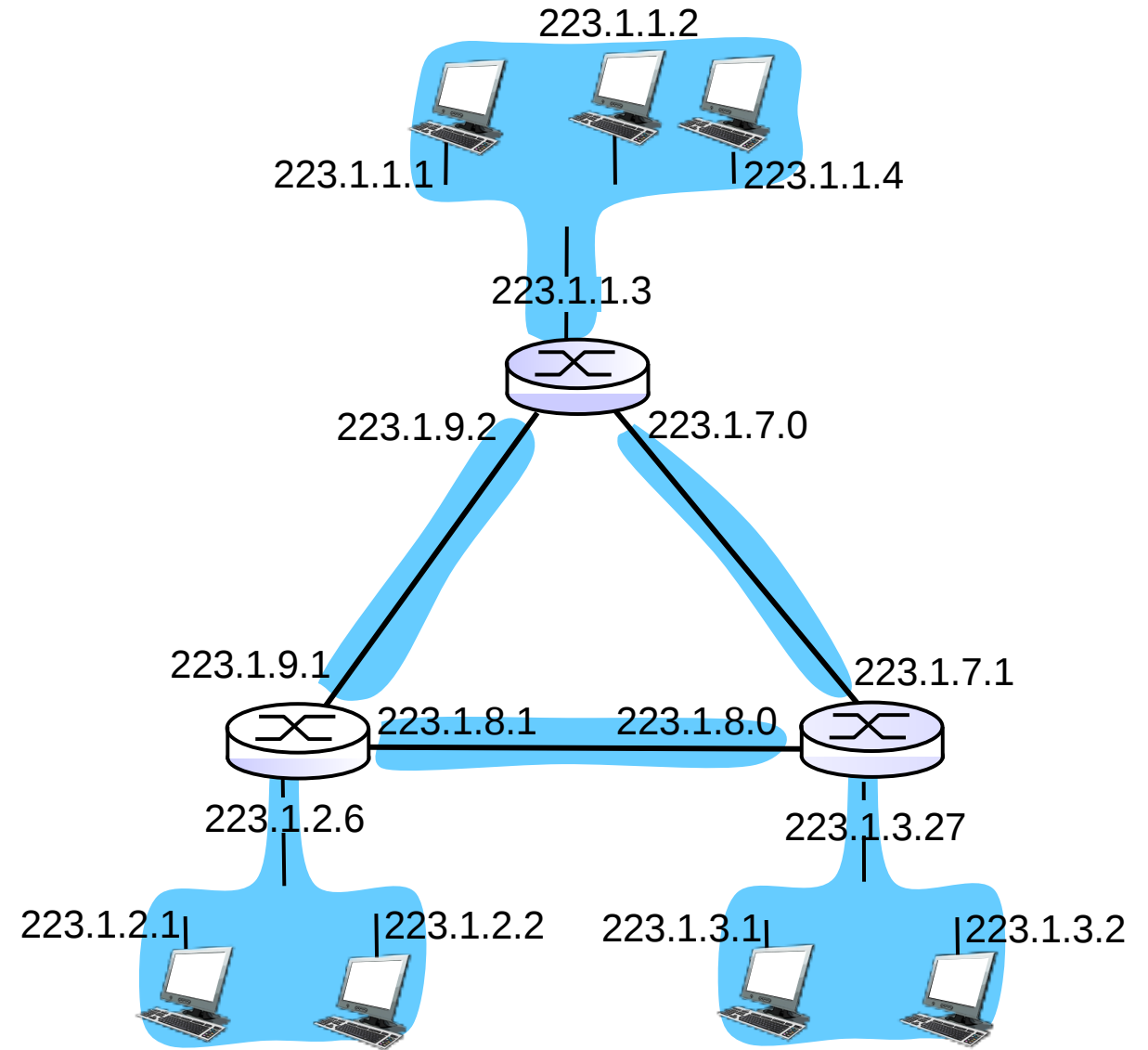
- **Corolário:** roteadores interconectam sub-redes.



máscara de sub-rede: /24

Sub-redes (III)

- Quantas sub-redes?



Quantos Bits Identificam a Sub-rede?

- Recapitulando: endereço IP é dividido em duas partes.
 - Porção da sub-rede (bits mais significativos).
 - Porção do *host* (bits menos significativos).
- **Pergunta:** quantos bits em cada porção?
 - Um endereço IP tem 32 bits **no total**.
 - Mas como eles são divididos nas duas porções?
 - Note que **nem toda sub-rede precisa ser do mesmo tamanho**.
- Duas alternativas:
 - Endereçamento baseado em classes (usado originalmente).
 - CIDR (usado atualmente).

Endereçamento Baseado em Classes

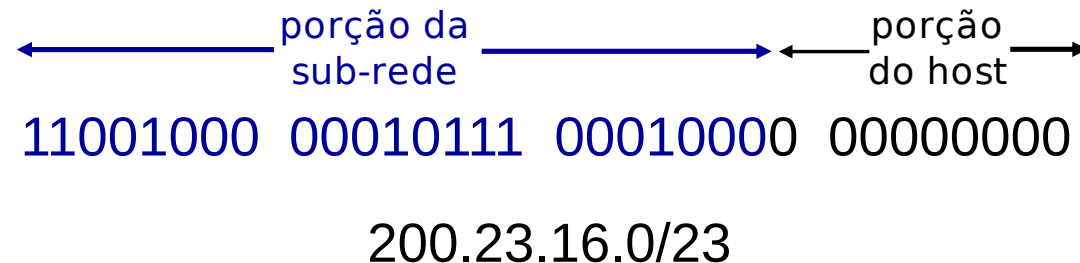
- Ideia básica: definimos sub-redes com propósitos diferentes.
 - **Classe A:** redes “grandes”.
 - Porção da sub-rede com 8 bits, sempre iniciados por 0.
 - $2^{24} = 16777216$ endereços.
 - **Classe B:** redes “médias”.
 - Porção da sub-rede com 16 bits, sempre iniciados por 10.
 - $2^{16} = 65536$ endereços.
 - **Classe C:** redes “pequenas”.
 - Porção da sub-rede com 24 bits, sempre iniciados por 110.
 - $2^8 = 256$ endereços.
 - **Classes D (Multicast) e E (Reservado):** outros usos.
 - Endereços iniciados por 1110 e 1111, respectivamente.
- Exemplos:
 - 10.151.0.1 é de classe A (começa por 0), porção da sub-rede é 00001010.
 - 200.20.15.156 é de classe C (começa por 110), porção da sub-rede é 11001000 00010100 00001111.

Endereçamento Baseado em Classes: Problemas

- Três tamanhos não são suficientes.
 - Pouca granularidade.
- Grande sub-utilização do espaço de endereçamento.
- Por exemplo: suponha que a UFF precise conectar 1000 dispositivos em uma sub-rede.
 - Sub-rede de classe C é muito pequena: apenas 256 endereços.
 - Solução: atribuir uma sub-rede de classe B.
 - Resultado: $65536 - 1000 = 64536$ endereços ociosos.
 - Mas **alocados**!

CIDR

- **CIDR: Classless InterDomain Routing.**
 - Porção de sub-rede de tamanho arbitrário.
 - Formato de endereço: **a.b.c.d/x**, onde x é o # de bits na porção da sub-rede.



- Convenciona-se que o endereço IP com todos os bits da porção do *host* iguais a zero seja denominado o “**endereço da rede**”.
- Se porção do *host* só contém 1's, trata-se do **endereço de broadcast**.
- Porção da sub-rede é comumente chamada de **prefixo**.
- O x, portanto, é o **tamanho do prefixo**.

CIDR: Máscara de Sub-rede

- Notação alternativa:
 - Número de 32 bits tal que um *and* bit-a-bit entre o endereço de um *host* e a máscara resulte no endereço da rede.
 - Exemplo:
 - Host tem endereço 200.20.10.89.
 - Máscara de sub-rede é 255.255.255.224.
 - Endereço da rede: 200.20.10.64.

	200.20.10.89	=	11001000	00010100	00001010	01011001
\wedge	255.255.255.224	=	11111111	11111111	11111111	11100000
=	200.20.10.64	=	11001000	00010100	00001010	01000000

- Na prática, máscaras são uma sequência de 1's seguida de uma sequência de 0's.
 - Tamanho da sequência de 1's é o tamanho do prefixo na notação CIDR.

CIRD e Tamanhos de Sub-Redes

- O CIDR permite sub-redes de tamanhos arbitrários?
 - **Não, ainda há restrições de granularidade.**
- Exemplos:
 - Prefixo de 27 bits — $2^5 = 32$ endereços.
 - Prefixo de 26 bits — $2^6 = 64$ endereços.
 - Prefixo de 25 bits — $2^7 = 128$ endereços.
 - Prefixo de 24 bits — $2^8 = 256$ endereços.
 - ...
- Mas há bem mais **opções**.