

# Engenharia de Computação



## Arquitetura de Sistemas Operacionais

## Gerência de Entrada e Saída

**Prof. Anderson Luiz Fernandes Perez**

Universidade Federal de Santa Catarina

Campus Araranguá

Email: [anderson.perez@ufsc.br](mailto:anderson.perez@ufsc.br)

# Conteúdo

- Introdução
- Objetivos do Gerenciador de E/S
- Modelo Computacional de E/S
- Drivers de Dispositivo
- Escalonamento de Braço de Disco

# Dispositivos de Entrada/Saída

- Dispositivo de entrada: **hardware** que fornece de dados ao computador
- Dispositivo de saída: **hardware** que expõe dados fornecidos pelo computador
- Dispositivo de entrada/saída: combinação dos dois anteriores

# Tipos de dispositivos de E/S



- Dispositivos classificados em dois grupos
  - Dispositivo de bloco
    - Discos
  - Dispositivo de caracter
    - Mouse
    - Teclado
  - Dispositivos em nenhum dos grupos acima
    - Tela touch-screen
    - Relógios

# Introdução

- A camada de E/S em um SO é responsável por prover um meio de comunicação entre os processos e os dispositivos de Entrada e Saída de dados.
- Os dispositivos de E/S, também conhecidos como periféricos, proveem uma gama de funcionalidades ao sistema computacional.
- A camada de E/S deve ser capaz de interfacear com diferentes dispositivos com diferentes configurações.

# Introdução



# Objetivos do Gerenciador de E/S

- Tornar a programação independente de periféricos e de todos os detalhes de E/S.
- Permitir a fácil inclusão de novos periféricos, o que implica que é possível a diferentes fabricantes criarem o software de adaptação entre o núcleo do SO e as funções específicas de controle de periféricos.

# Objetivos do Gerenciador de E/S



- Identificar dispositivos deve uniformemente e sem depender do tipo de dispositivo



# Objetivos do Gerenciador de E/S

```

martin — -bash — 80x26

[macwork:~ martin$ diskutil list
/dev/disk0 (internal, physical):
#          TYPE NAME              SIZE          IDENTIFIER
0:        GUID_partition_scheme    *1.0 TB       disk0
1:          EFI EFI                209.7 MB      disk0s1
2:      Apple_CoreStorage Untitled      872.4 GB      disk0s2
3:      Apple_Boot Recovery HD      650.0 MB      disk0s3
4:      Apple_CoreStorage Untitled      126.3 GB      disk0s4
5:      Apple_Boot Recovery HD      650.0 MB      disk0s5
/dev/disk1 (internal, virtual):
#          TYPE NAME              SIZE          IDENTIFIER
0:      Apple_HFS MacOS              +126.0 GB     disk1
               Logical Volume on disk0s4
               54E95F6E-52A1-4A81-AF02-855418FCDB70
               Unencrypted
/dev/disk2 (internal, virtual):
#          TYPE NAME              SIZE          IDENTIFIER
0:      Apple_HFS Macintosh HD      +872.0 GB     disk2
               Logical Volume on disk0s2
               C34C2970-1A52-4E81-86DA-9AB3501A66A3
               Unlocked Encrypted
/dev/disk3 (external, physical):
#          TYPE NAME              SIZE          IDENTIFIER
0:      FDisk_partition_scheme      *4.0 GB       disk3
1:      DOS_FAT_32 UNTITLED         4.0 GB       disk3s1
macwork:~ martin$

```

# Objetivos do Gerenciador de E/S

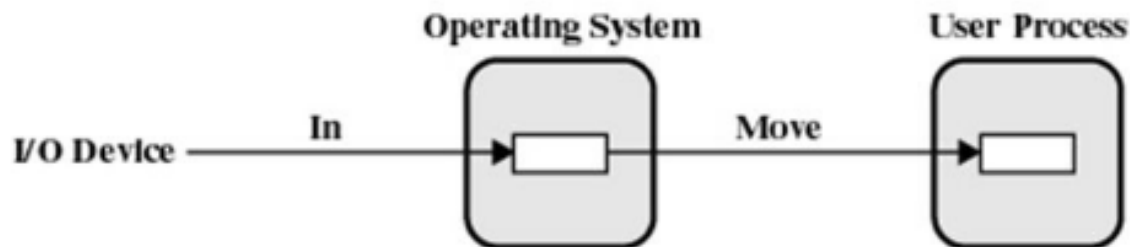
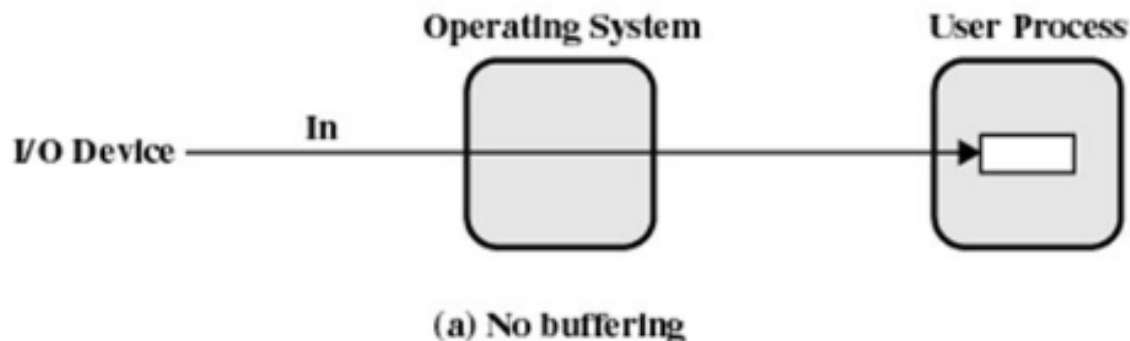
- Tratar erros o mais próximo do hardware (preferência de não delegar pro SO)
  - Exemplo: retransmitir dados pela rede

# Objetivos do Gerenciador de E/S

- Fornecer acesso síncrono (bloqueante) e assíncrono (dirigido a interrupção)
  - Síncrono: mais simples de criar aplicações
  - Assíncrono: maior desempenho

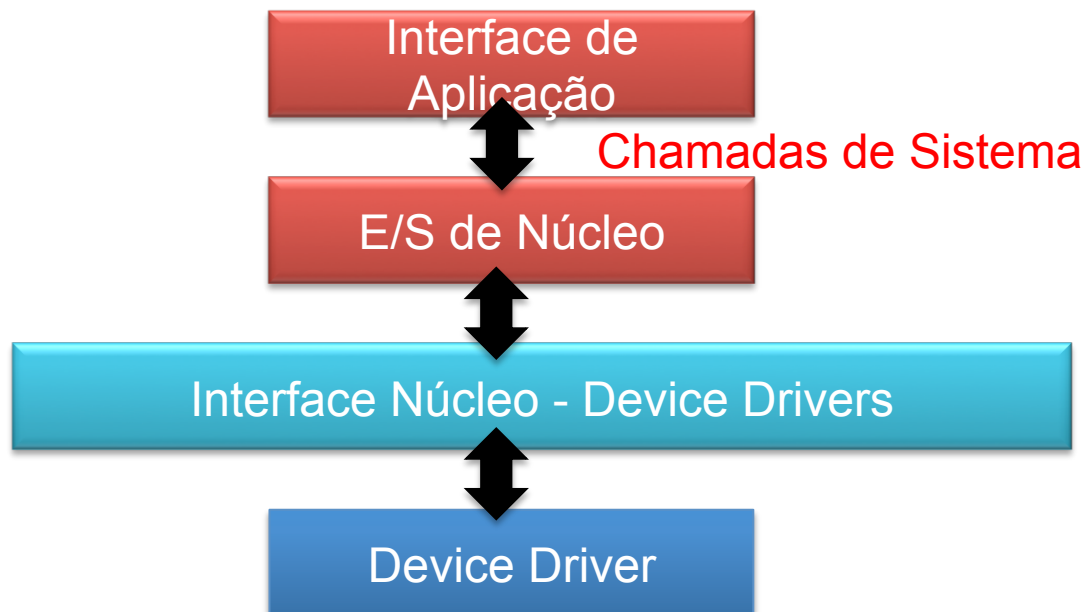
# Objetivos do Gerenciador de E/S

- Armazenamento temporário (buffering)
  - Pacotes rede: armazenar, analisar, entregar
  - Tempo real: tocar DVD sem cortes



# Modelo Computacional de E/S

- Arquitetura de um Sistema de E/S



# Interface Núcleo – Device Drivers



- Controladores de dispositivos têm
  - Registradores para programar dispositivo
  - Buffers para entrada/saída com dispositivo
- Acesso a registradores e buffers
  - Espaço de portas
  - Mapeamento em memória

# Interface Núcleo – Device Drivers



- Espaço de portas de E/S
  - SO atribui um porta de E/S ao dispositivo
  - Usam-se instruções assembly IN e OUT
- Mapeamento em memória
  - Registradores e buffers mapeados em memória
  - Permite escrever drivers em C

# Modelo Computacional de E/S

- O modelo computacional da camada de E/S é composto por duas entidades:
  1. **Periféricos Virtuais:** todas as operações de E/S se realizam sobre uma entidade abstrata, um periférico virtual.
  2. **Funções de E/S:** funções padronizadas associadas ao periféricos virtuais possibilitam a programação, assegurando a independência entre aplicações e periféricos, e simplificando o modelo de segurança.



# Modelo Computacional de E/S



- Para que um processo realize operações sobre um periférico, as seguintes operações devem ser disponibilizadas pela camada de E/S (1/3):
  - **Abertura e Fechamento do Periférico Virtual**
    - Fase de abertura de um canal virtual com o periférico e a criação de um conjunto de estruturas no núcleo para suporte à comunicação com esse periférico.

# Modelo Computacional de E/S

- Para que um processo realize operações sobre um periférico, as seguintes operações devem ser disponibilizadas pela camada de E/S (2/3):
  - Operações de Comunicação e Configuração
    - Os sistemas operacionais definem um conjunto de operações para manipulação dos dispositivos de E/S. As linguagens de programação permitem a interação com dispositivos de E/S a partir de bibliotecas que são ligadas ao programa.

# Modelo Computacional de E/S



- Para que um processo realize operações sobre um periférico, as seguintes operações devem ser disponibilizadas pela camada de E/S (2/3):
  - **Fechamento**
    - Quando o processo não necessitar mais do periférico, esse deverá executar a operação de fechamento explicitamente, indicando ao SO que o periférico não está mais em uso.

# Modelo Computacional de E/S

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 int main(){
6     FILE *arq;
7     arq = fopen("ArqGrav1.txt","wb");
8
9     char str[20] = "Hello World";
10    float x = 5;
11    int v[5] = {1,2,3,4,5};
12
13    //grava str[0..strlen(str)]
14    fwrite(str,sizeof(char),strlen(str),arq);
15
16    //grava str[0..4]
17    fwrite(str,sizeof(char),5,arq);
18
19    //grava x
20    fwrite(&x,sizeof(float),1,arq);
21
22    //grava v[0..4]
23    fwrite(v,sizeof(int),5,arq);
24
25    //grava v[0..1]
26    fwrite(v,sizeof(int),2,arq);
27    fclose(arq);
28
29    return 0;
30 }
```

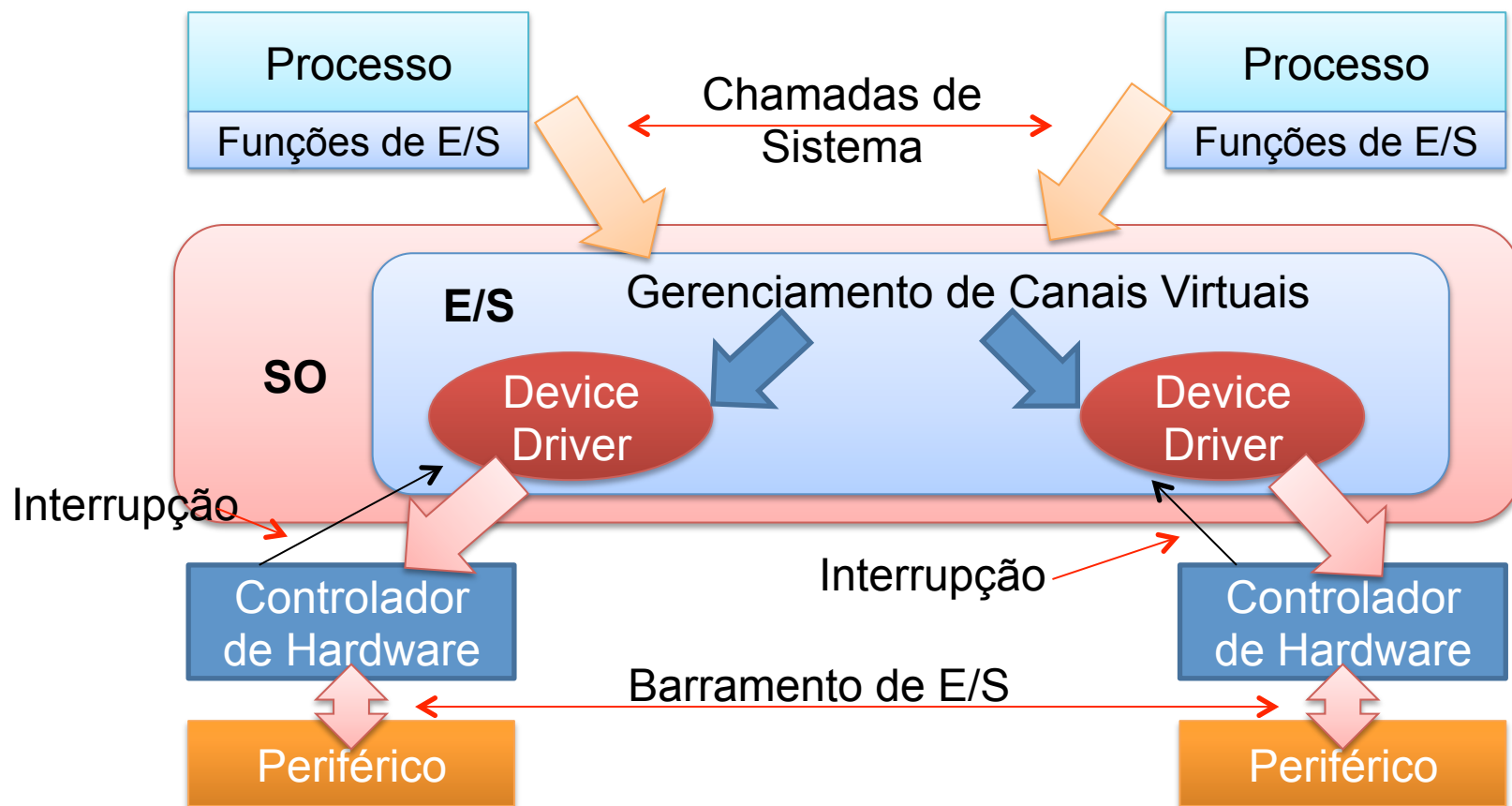
Abertura: cria *stream* (abstração de disp)

Operação de comunicação

Fechamento: fecha *stream*

# Modelo Computacional de E/S

- Ações de um Operação de E/S



# Drivers de Dispositivo

- Os *device drivers* podem ser de dois tipos com relação ao SO:
  1. **Um processo independente:** neste caso o device driver tem uma pilha de execução, cache, espaços de memória etc.
  2. **Integrado ao núcleo:** o *device driver* faz parte do núcleo do SO, evitando assim muitas troca de contexto na comunicação entre um processo e um dispositivo de E/S.

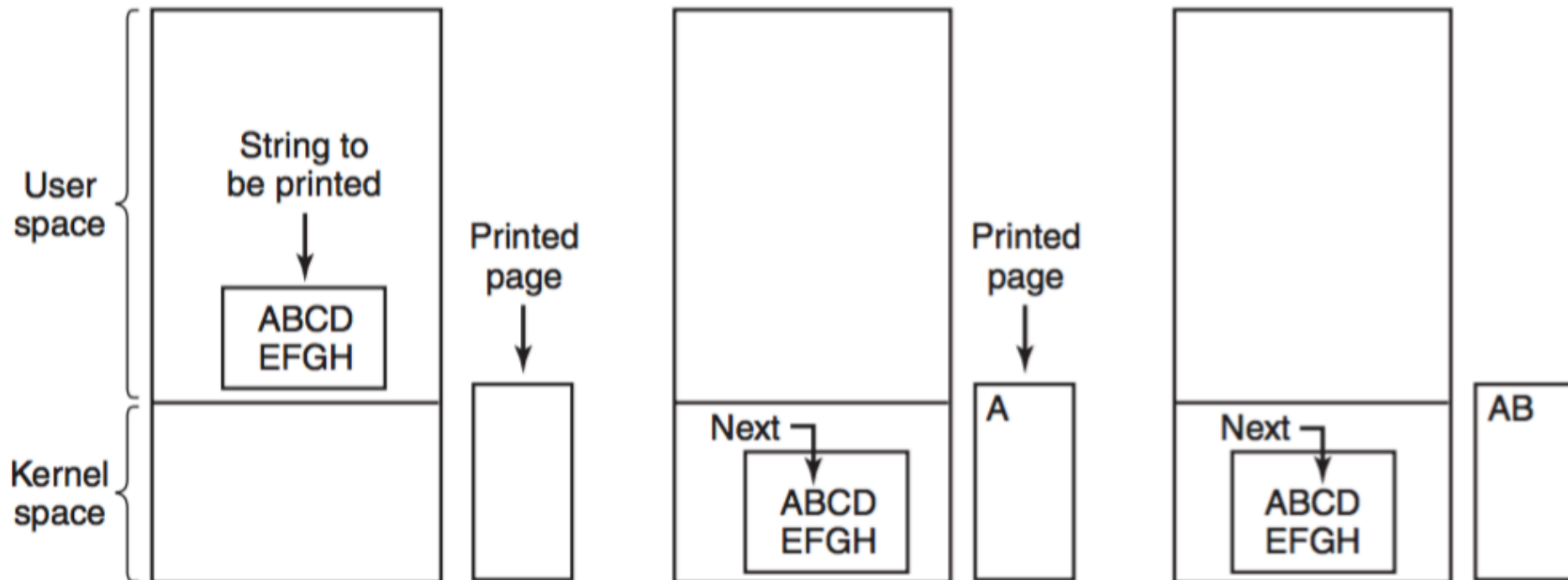
# Formas de E/S

- E/S Programada

- Leitura de dados diretamente do controlador. Por exemplo, atualização das informações sobre o posicionamento do mouse na interface gráfica.
- **Polling ou busy waiting:** SO testa periodicamente testa o estado do controlador.

# Formas de E/S

- Exemplo E/S programada: impressão





# Formas de E/S

- Exemplo E/S programada: impressão

```
copy_from_user(buffer, p, count);  
for (i = 0; i < count; i++) {  
    while (*printer_status_reg != READY) ;  
    *printer_data_register = p[i];  
}  
return_to_user();
```

/\* p is the kernel buffer \*/  
/\* loop on every character \*/  
/\* loop until ready \*/  
/\* output one character \*/

# Formas de E/S

- Dirigida a interrupção
  - Dispositivo avisa SO via interrupção quando a tarefa de E/S terminou
  - SO não fica ocioso esperando
  - Interrupção tratada através de um procedimento do SO (consome CPU)

# Formas de E/S

- Dirigida a interrupção

```
copy_from_user(buffer, p, count);  
enable_interrupts();  
while (*printer_status_reg != READY) ;  
*printer_data_register = p[0];  
scheduler();
```

(a)

Código executado quando  
usuário solicita impressão

```
if (count == 0) {  
    unblock_user();  
} else {  
    *printer_data_register = p[i];  
    count = count - 1;  
    i = i + 1;  
}  
acknowledge_interrupt();  
return_from_interrupt();
```

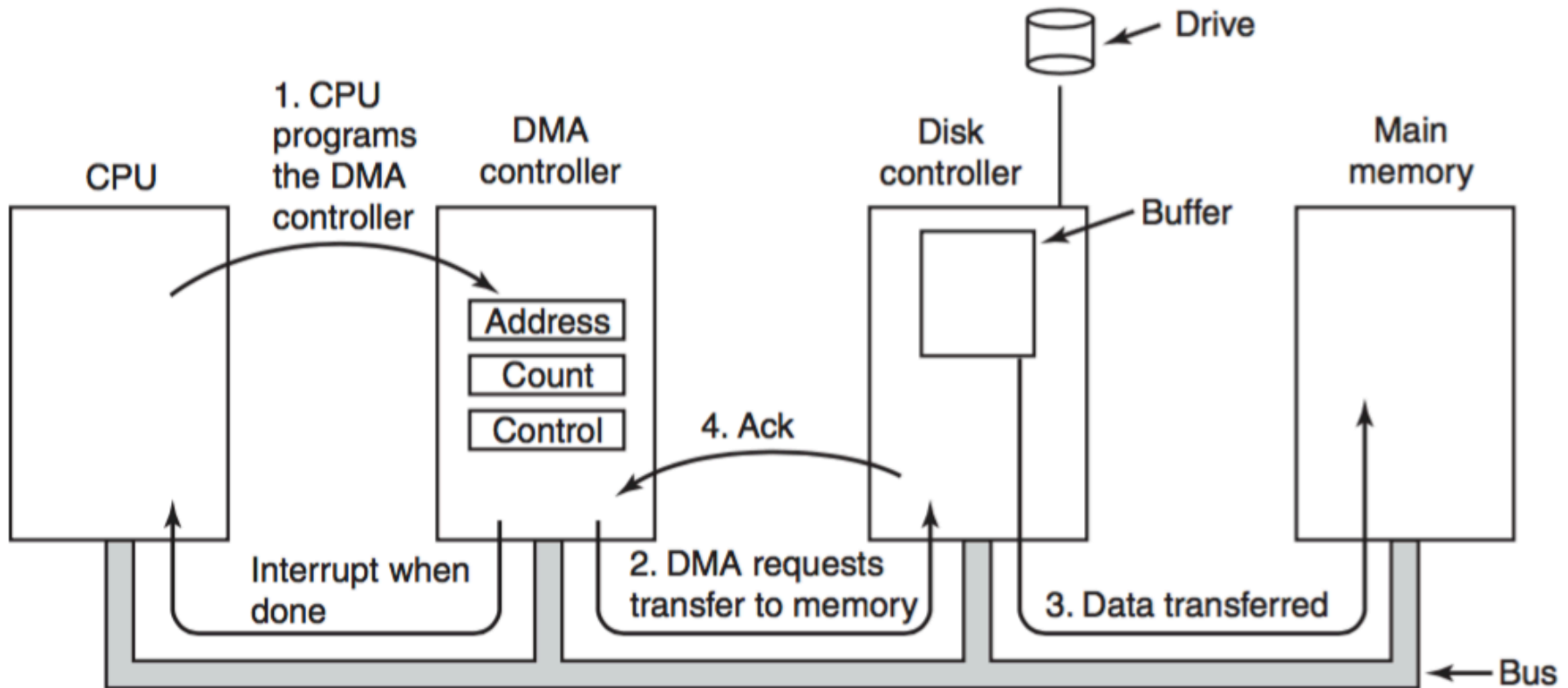
(b)

Procedimento executando  
quando ocorre interrupção

# Formas de E/S

- **Acesso Direto a Memória (*DMA – Direct Memory Access*)**
  - Controlador DMA transfere blocos de dados da memória principal para dispositivo e vice-versa
  - Reduz-se a quantidade de interrupções
    - Interrupção quando controlador termina a tarefa

# Formas de E/S: DMA



# Formas de E/S



```
copy_from_user(buffer, p, count);  
set_up_DMA_controller();  
scheduler();
```

(a)

Código executado quando  
usuário solicita impressão

```
acknowledge_interrupt();  
unblock_user();  
return_from_interrupt();
```

(b)

Procedimento executando  
quando ocorre interrupção

# Armazenamento persistente de alta capacidade



Disco magnético

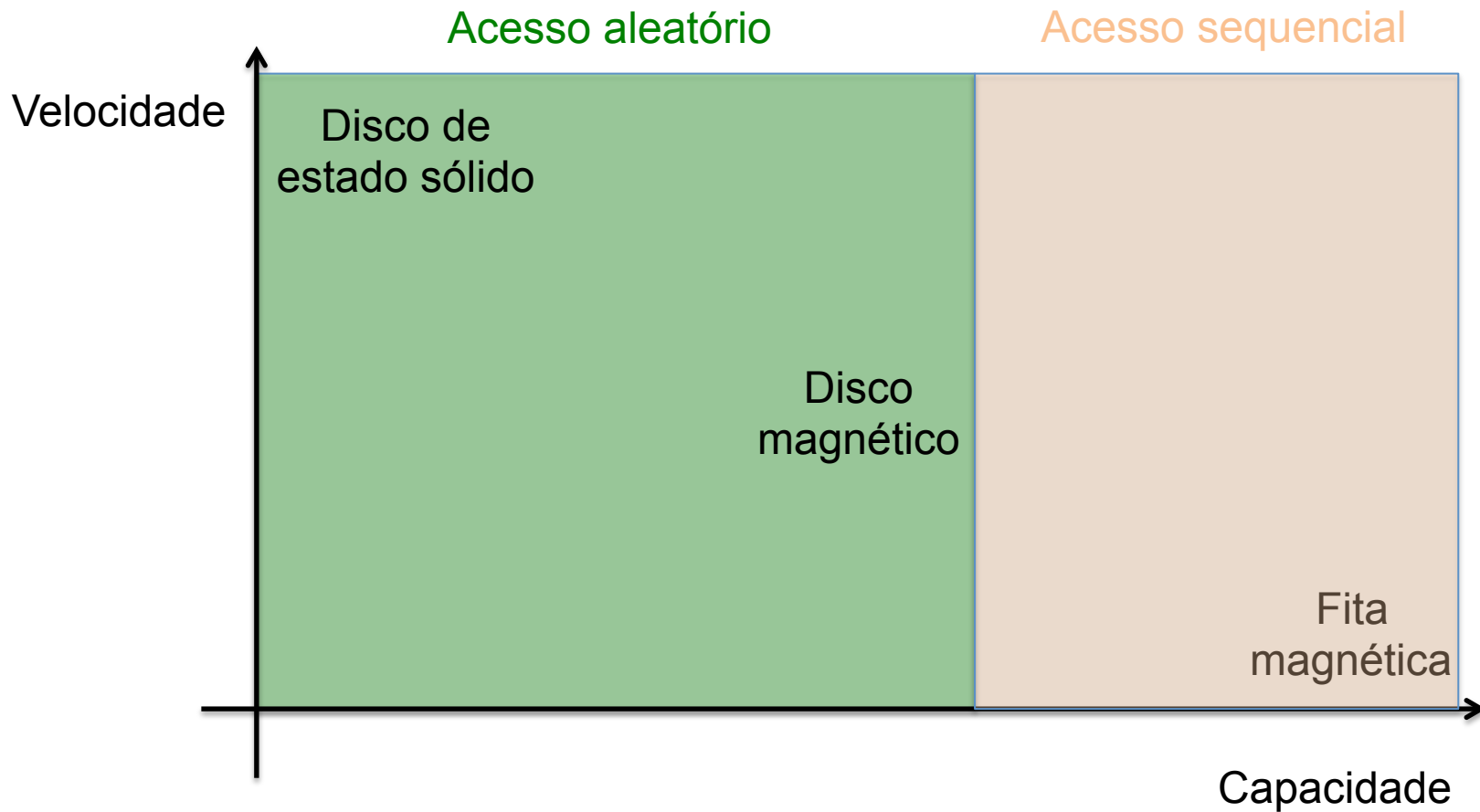


Disco de  
Estado sólido



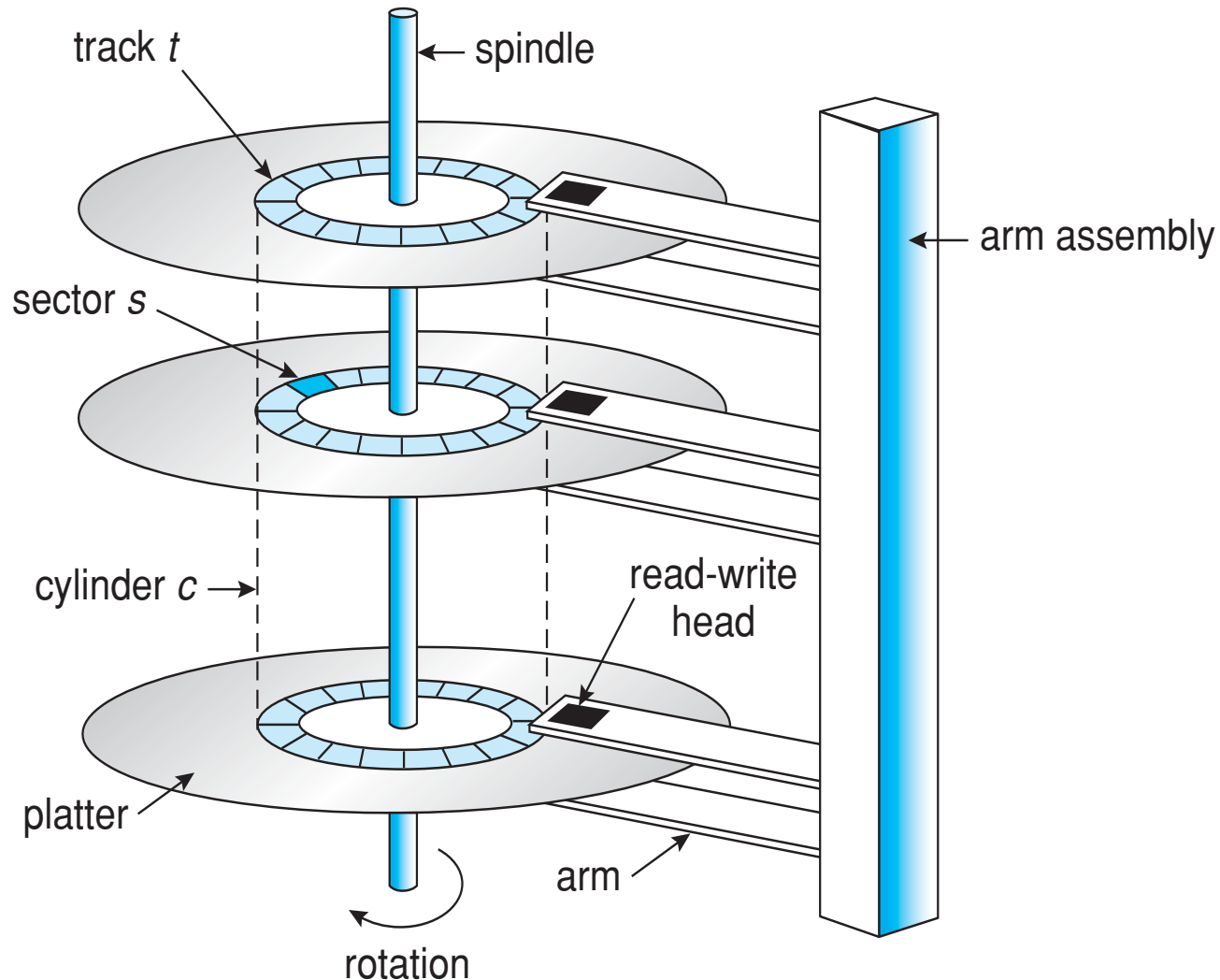
Fita magnética

# Armazenamento persistente de alta capacidade





# Estrutura do disco magnético



# Performance do disco magnético

- Taxa de transferência: bytes/segundo
- Tempo de posicionamento até o cilindro (seek time):  $\sim 9\text{ms}$  (desktop drives)
- Tempo de rotação até o setor (latência rotacional)
- Tempo de acesso: posicionamento + rotação

# Algoritmos de rotacionamento

- Pedidos de escrita/leitura do disco vindos do SO e usuários
- Atende pedidos de modo a minimizar tempo acesso
- Tempo de acesso  $\sim$  distância de acesso

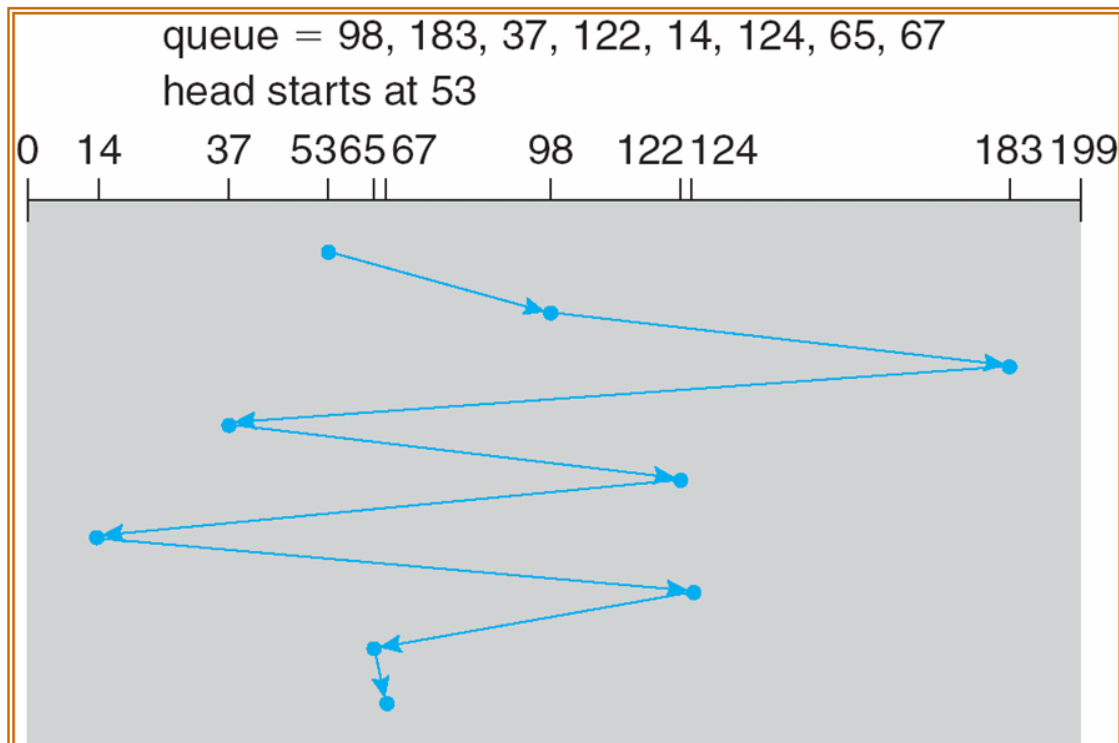
# Escalonamento de Braço de Disco



- FCFS (First Come First Served)
  - É o algoritmo mais simples.
  - As solicitações de acesso ao disco são realizadas na ordem em que os pedidos são feitos.
  - Nenhuma tentativa é feita para reorganizar a ordem dos pedidos visando otimizar os movimentos da cabeça de leitura/escrita entre os cilindros.

# Escalonamento de Braço de Disco

- Exemplo: FCFS – posição inicial 53



| Cilindro     | Distância  |
|--------------|------------|
| 53           | 0          |
| 98           | 45         |
| 183          | 85         |
| 37           | 146        |
| 122          | 85         |
| 14           | 108        |
| 124          | 110        |
| 65           | 59         |
| 67           | 2          |
| <b>Total</b> | <b>640</b> |

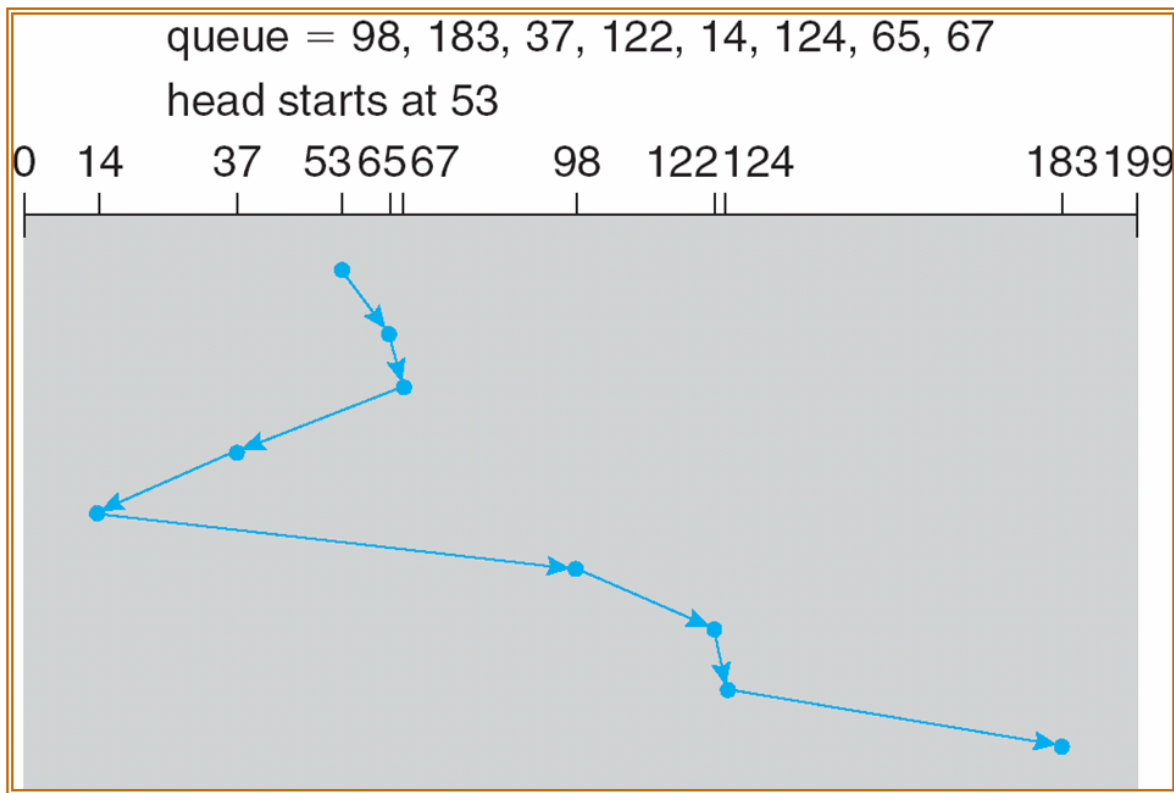
# Escalonamento de Braço de Disco



- SSTF (Shortest Seek Time First )
  - Novos pedidos são ordenados em relação à posição atual do cabeça de leitura/escrita, privilegiando assim o acesso aos cilindros que estão mais próximos a esta posição.
  - A desvantagem é que pode levar um pedido a postergação indefinida (starvation).

# Escalonamento de Braço de Disco

- Exemplo: SSTF – posição inicia: 53



| Cilindro     | Distância  |
|--------------|------------|
| 53           | 0          |
| 65           | 12         |
| 67           | 2          |
| 37           | 30         |
| 14           | 23         |
| 98           | 84         |
| 122          | 23         |
| 124          | 2          |
| 183          | 59         |
| <b>Total</b> | <b>236</b> |

# Escalonamento de Braço de Disco

- Nem sempre leva a solução ótima

| Cilindro     | Distância  |
|--------------|------------|
| 53           | 0          |
| 65           | 12         |
| 67           | 2          |
| 37           | 30         |
| 14           | 23         |
| 98           | 84         |
| 122          | 23         |
| 124          | 2          |
| 183          | 59         |
| <b>Total</b> | <b>236</b> |

SSTF



| Cilindro     | Distância  |
|--------------|------------|
| 53           | 0          |
| 37           | 16         |
| 14           | 23         |
| 65           | 51         |
| 67           | 2          |
| 98           | 31         |
| 122          | 24         |
| 124          | 2          |
| 183          | 59         |
| <b>Total</b> | <b>208</b> |

SSTF modificado 40



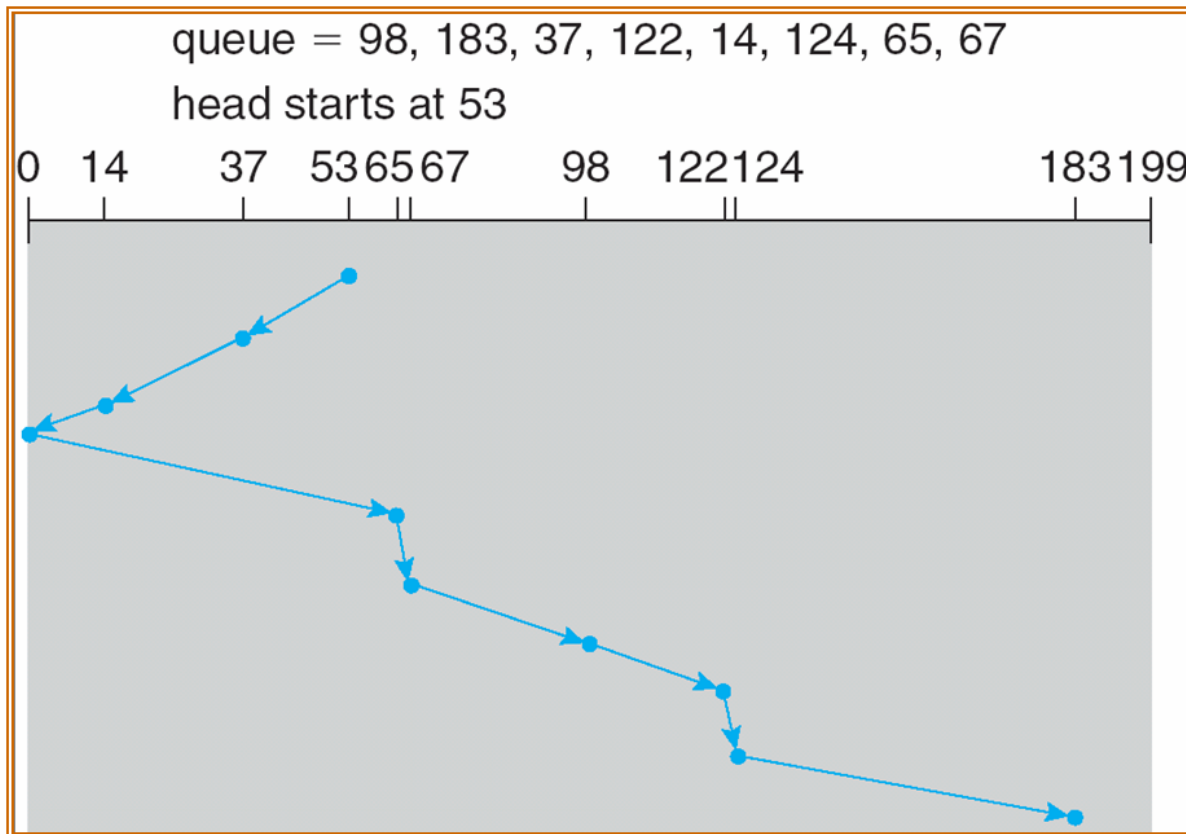
# Escalonamento de Braço de Disco



- SCAN
  - Uma variação do SSTF.
  - Visa atender os pedidos que estão mais próximos da cabeça de leitura/escrita.
  - Atende os pedidos em um sentido (cilindro mais externo ao mais interno). Logo depois inverte o sentido.
  - Também conhecido como algoritmo do elevador.

# Escalonamento de Braço de Disco

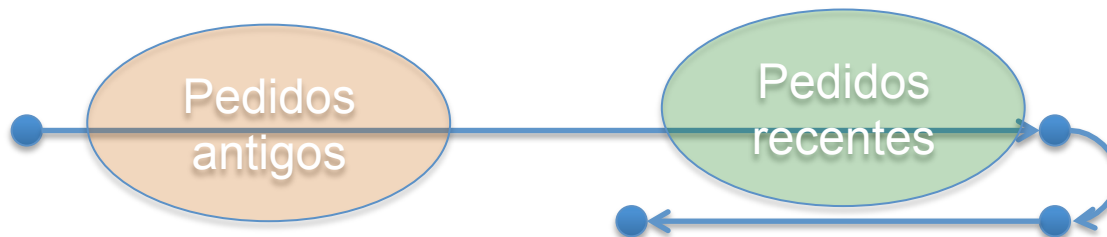
- Exemplo: SCAN – posição inicial 53



| Cilindro | Distância |
|----------|-----------|
| 53       | 0         |
| 37       | 16        |
| 14       | 23        |
| 0        | 14        |
| 65       | 65        |
| 67       | 2         |
| 98       | 31        |
| 122      | 24        |
| 124      | 2         |
| 183      | 59        |
| Total    | 236       |

# Escalonamento de Braço de Disco

- SCAN: tempo de espera dos pedidos



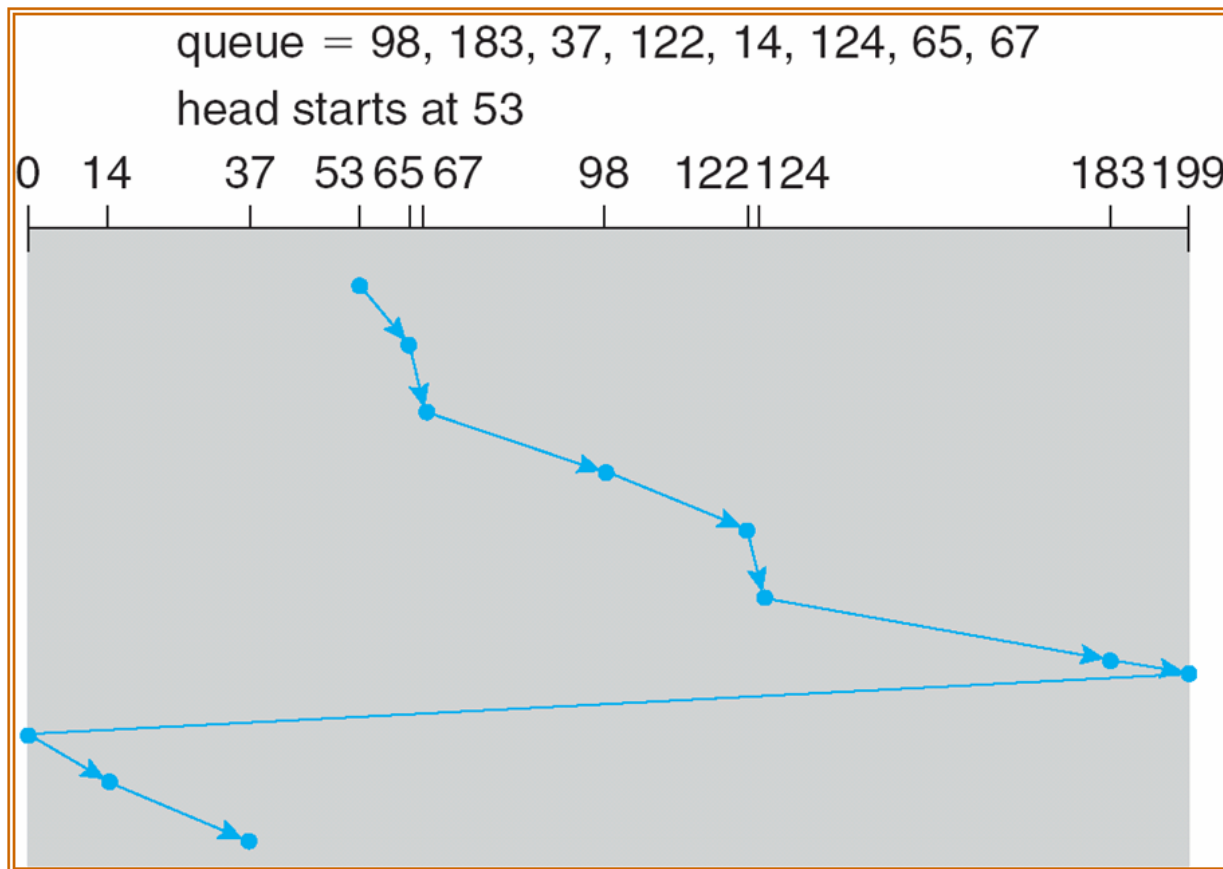
# Escalonamento de Braço de Disco



- C-SCAN
  - Variação do SCAN
  - Tempo de espera mais uniforme que SCAN
  - Os pedidos são atendidos em um único sentido

# Escalonamento de Braço de Disco

- Exemplo: C-SCAN – posição inicial: 53



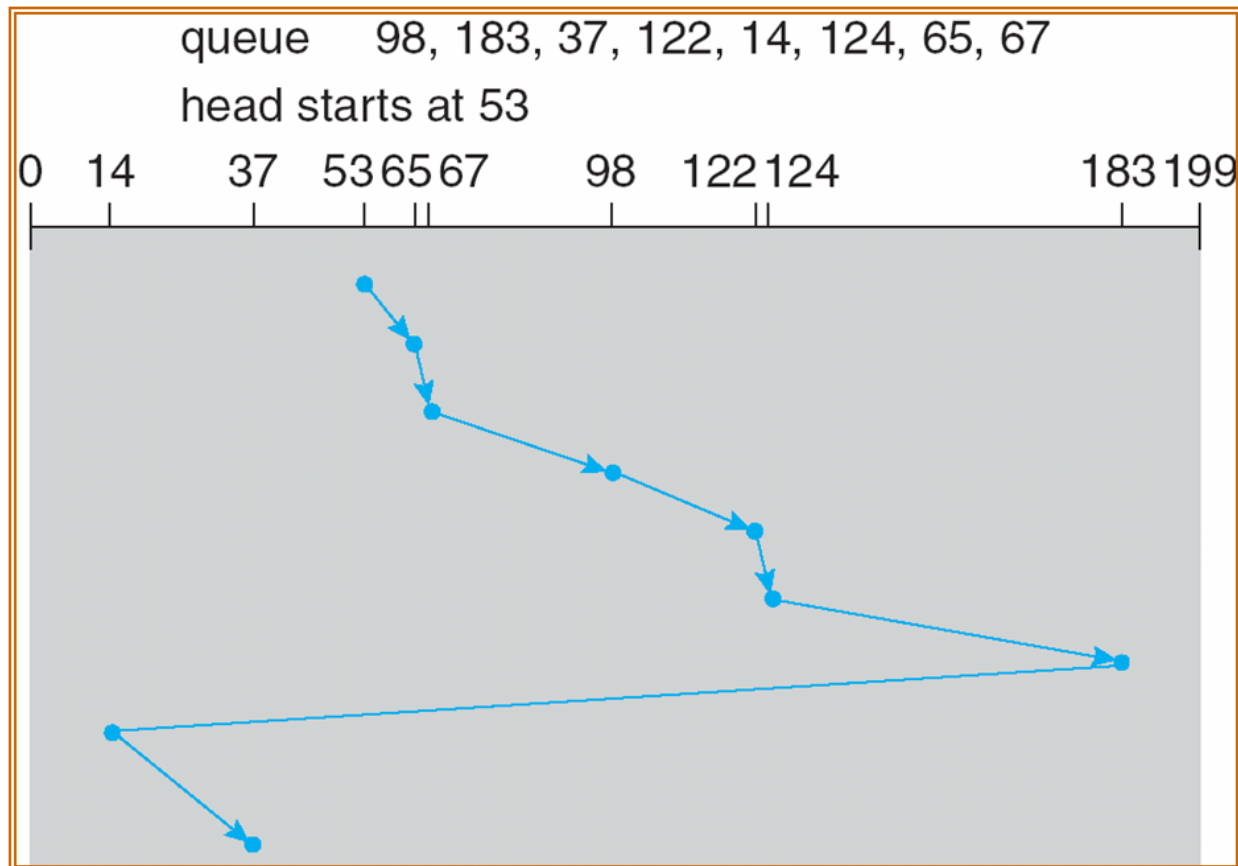
# Escalonamento de Braço de Disco



- C-LOOK
  - Variação do C-SCAN.
  - O braço do disco só vai até a distância da última solicitação em cada direção, depois reverte a direção imediatamente, sem primeiro ir até o final do disco.

# Escalonamento de Braço de Disco

- Exemplo: C-LOOK – Ponteiro da cabeça = 53



# Escolha do algoritmo

- SSTF ou LOOK são geralmente escolhidos
- SCAN e C-SCAN são indicados para sistemas muito acesso a disco
  - Evitam starvation
- Performance depende no número e tipo dos pedidos (e.g. alg. equivalentes se há somente um pedido na fila)
- Escolha do algoritmo depende do sistema de arquivos
  - Alocação contígua vs. alocação indexada
  - Onde alocar diretórios e arquivos?
- Difícil de estimar latência rotacional



# RAID



- Redundante Arrays of Independent Disks
- Modos de utilizar distintos discos em paralelo como uma única unidade de armazenamento
- Objetivos
  - Performance
  - Confiabilidade por redundância

# Confiabilidade via redundância

- Replicar dados entre discos distintos
- Técnica de espelhamento (mirroring)
- 1 volume lógico : n discos redundantes

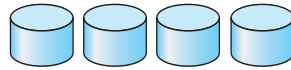
# Performance via paralelismo

- Cada dado é dividido e as partes são escritas em distintos discos (data stripping)
- Em nível de bits ou blocos
- Permite
  - Balanceamento de carga entre discos
  - Reduzir tempo de resposta

# Níveis de RAID

- Podem combinar redundância com paralelismo
- RAID 0: apenas paralelismo em bloco
- RAID 1: paralelismo + redundância (espelhamento)
- RAID 2--6: paralelismo + redundância (bits de paridade, menos redundância que espelhamento)

# RAID Levels



(a) RAID 0: non-redundant striping.



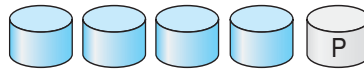
(b) RAID 1: mirrored disks.



(c) RAID 2: memory-style error-correcting codes.



(d) RAID 3: bit-interleaved parity.



(e) RAID 4: block-interleaved parity.



(f) RAID 5: block-interleaved distributed parity.



(g) RAID 6: P + Q redundancy.