Escalonamento de Discos

SISTEMAS OPERACIONAIS

AMANDA JULIA - 12211BSI225
ARTHUR CARVALHO - 12211BSI220
LAYZA NAUANE - 12211BSI251
ISABELLA PAULA - 12211BSI219
PAULA PRADO - 12211BSI267
JAQUELINE GONÇALVES- 12211BSI249

Tema 7

Gerenciamento de Disco

MAXIMIZA A EFICIÊNCIA E A SEGURANÇA DOS DADOS

A GESTÃO EFICIENTE E SEGURA DOS DADOS É ESSENCIAL PARA GARANTIR A INTEGRIDADE E A FUNCIONALIDADE DO DISCO RÍGIDO. ESTE PROCESSO ABRANGE DESDE A FORMATAÇÃO INICIAL ATÉ A RECUPERAÇÃO DE DADOS E A DESFRAGMENTAÇÃO, CADA ETAPA CONTRIBUINDO PARA A OTIMIZAÇÃO DO DESEMPENHO E A PROTEÇÃO DOS DADOS ARMAZENADOS.

Formatação:

1

FORMATAÇÃO FÍSICA

A FORMATAÇÃO FÍSICA ESTABELECE AS BASES FÍSICAS PARA O ARMAZENAMENTO DE DADOS, ORGANIZANDO OS SETORES DO DISCO EM TRILHAS E CILINDROS.

2

FORMATAÇÃO LÓGICA

A FORMATAÇÃO LÓGICA ADAPTA O DISCO PARA FUNCIONAR COM UM SISTEMA OPERACIONAL ESPECÍFICO, CRIANDO PARTIÇÕES E FORMATANDO COM UM SISTEMA DE ARQUIVOS COMPATÍVEL.

Blocos de Disco:

ORGANIZAÇÃO FLEXÍVEL

OS BLOCOS LÓGICOS PROPORCIONAM FLEXIBILIDADE EXCEPCIONAL NA GRAVAÇÃO E LEITURA DE INFORMAÇÕES, GARANTINDO UM ACESSO RÁPIDO E EFICIENTE AOS DADOS.

BASE PARA ORGANIZAÇÃO DE DADOS

CADA BLOCO, GERALMENTE COM TAMANHO FIXO DE 512 BYTES, SERVE COMO A BASE PARA A ORGANIZAÇÃO DE DADOS EM CILINDROS E CABEÇAS DO DISCO RÍGIDO.

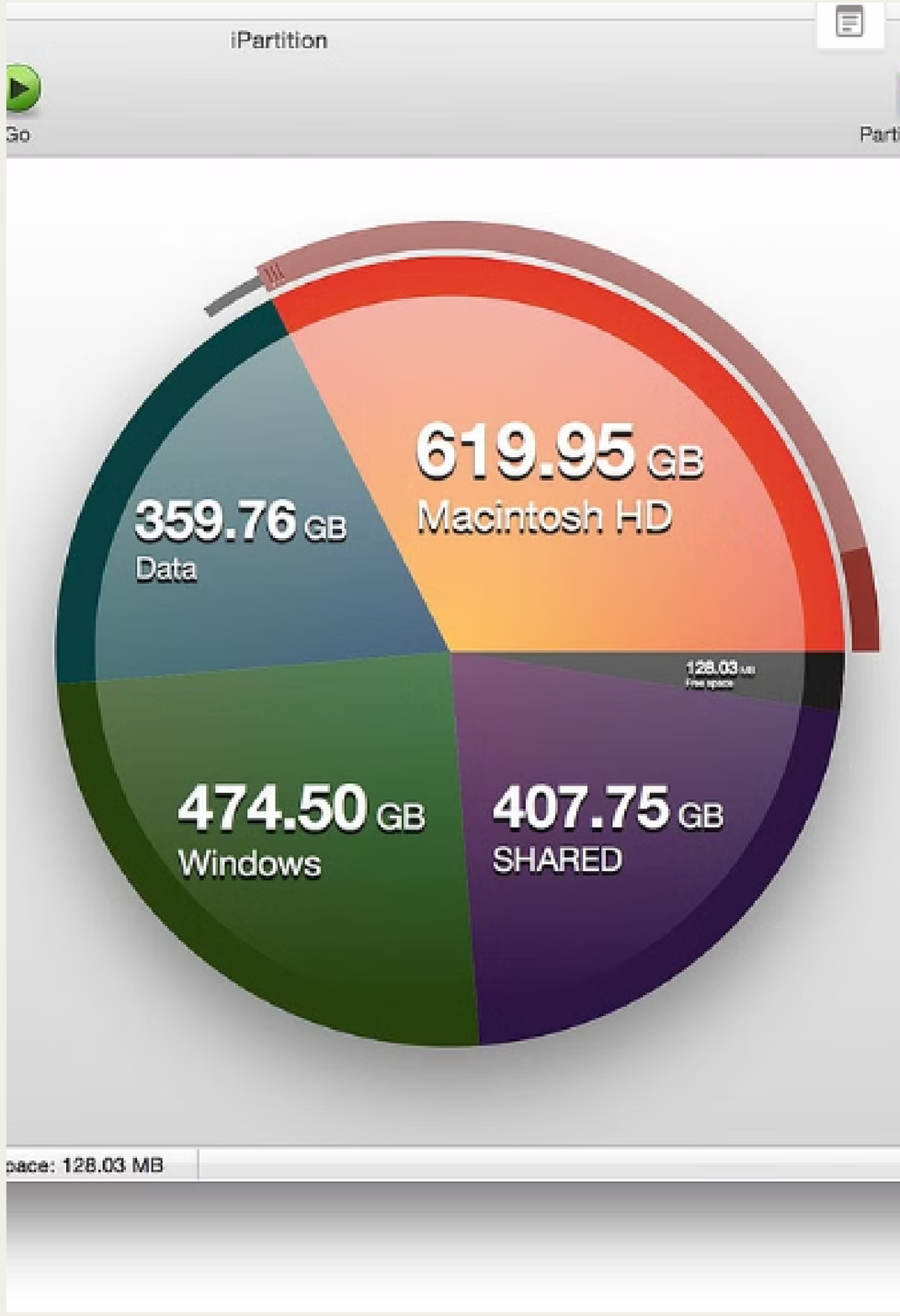
Gerenciamento de Partições:

1 **CRIAÇÃO INTELIGENTE**

A GESTÃO DE PARTIÇÕES PERMITE UMA ALOCAÇÃO INTELIGENTE DE RECURSOS, GARANTINDO UMA EXPERIÊNCIA DE USUÁRIO FLUIDA E EFICIENTE.

2 **COMPATIBILIDADE MULTI-SISTEMA**

PERMITE GARANTIR A COMPATIBILIDADE COM MÚLTIPLOS SISTEMAS OPERACIONAIS, OTIMIZANDO O USO DO ESPAÇO EM DISCO.



Sistemas de Arquivos:

FAT

SISTEMA DE ARQUIVOS COM AMPLA COMPATIBILIDADE, COMUMENTE USADO EM DISPOSITIVOS DE ARMAZENAMENTO REMOVÍVEIS E SISTEMAS OPERACIONAIS MAIS ANTIGOS.

EXT

SISTEMA DE ARQUIVOS TÍPICO EM AMBIENTES LINUX, OFERECENDO SUPORTE A PERMISSÕES DE ARQUIVO E EFICIÊNCIA DE ARMAZENAMENTO.

NTFS

SUPORTE A UMA AMPLA GAMA DE RECURSOS AVANÇADOS, INCLUINDO PERMISSÕES DE ARQUIVO, COMPRESSÃO E CRIPTOGRAFIA.

Recuperação de Dados:

MITIGAÇÃO DE FALHAS

A RECUPERAÇÃO DE DADOS É ESSENCIAL PARA MITIGAR OS IMPACTOS DE FALHAS DE HARDWARE, ERROS HUMANOS OU CORRUPÇÃO DE ARQUIVOS.

PROTEÇÃO CONTRA PERDAS

RESTAURAR DADOS PERDIDOS OU DANIFICADOS PROTEGE CONTRA PERDAS CATASTRÓFICAS, GARANTINDO A INTEGRIDADE DOS DADOS ARMAZENADOS.

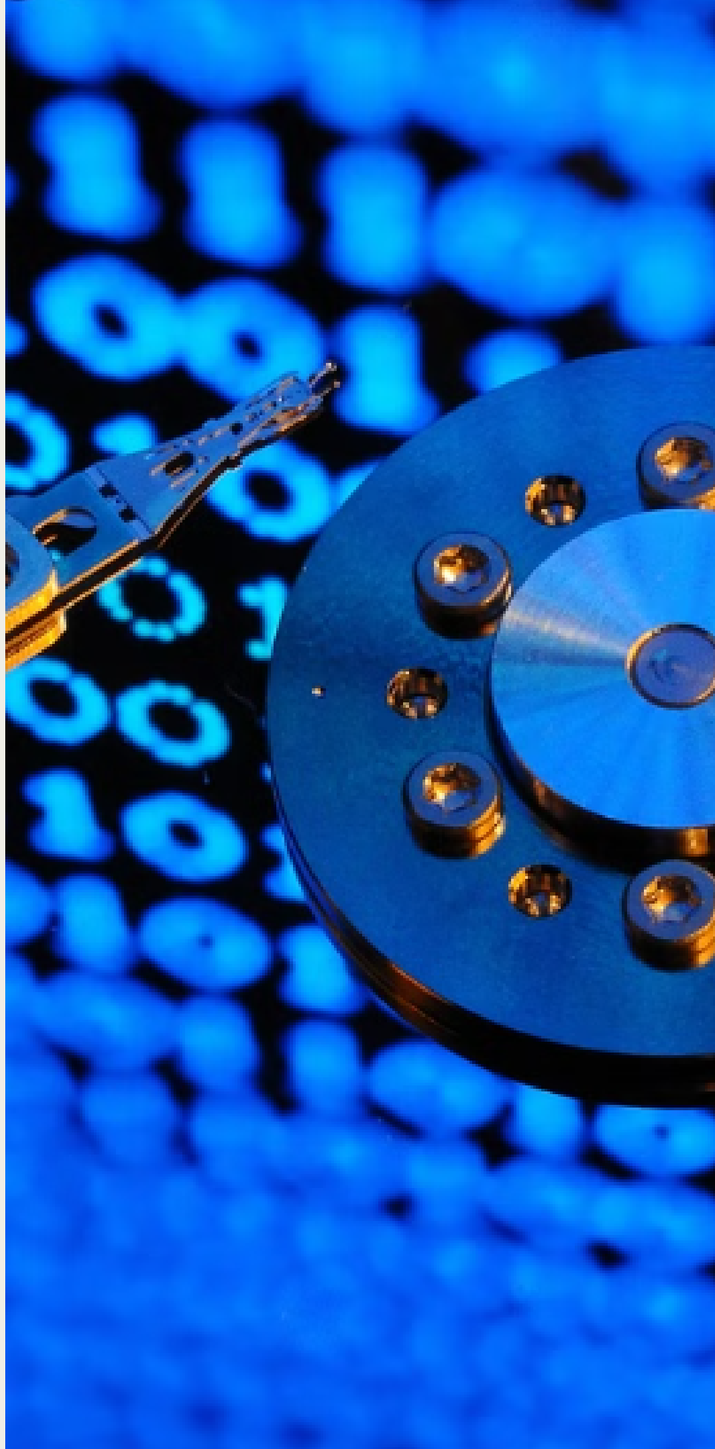
Desfragmentação:

REORGANIZAÇÃO EFICIENTE

A DESFRAGMENTAÇÃO REDUZ O TEMPO DE ACESSO AOS DADOS, MELHORANDO A EFICIÊNCIA GERAL DO SISTEMA.

PROTEÇÃO CONTRA PERDAS

RESTAURAR DADOS PERDIDOS OU DANIFICADOS PROTEGE CONTRA PERDAS CATASTRÓFICAS, GARANTINDO A INTEGRIDADE DOS DADOS ARMAZENADOS.



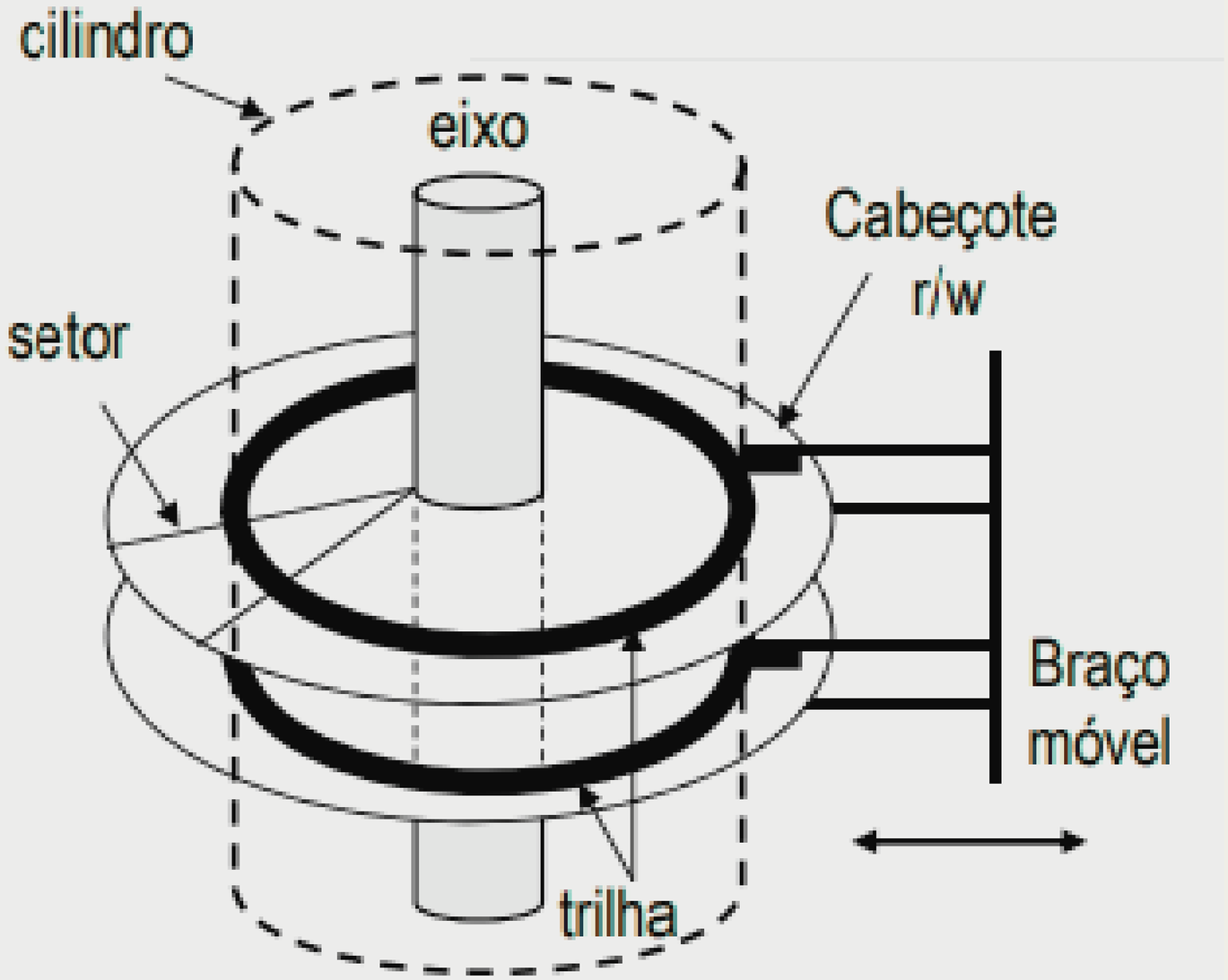
PARÂMETROS DE DESEMPENHO

FUNCIONAMENTO

Em sistemas de armazenamento de dados, como HDDs, o desempenho do escalonamento do disco é avaliado por:

- Tempo de busca
- Tempo de rotação
- Tempo de acesso
- Tempo de transferência

FUNCIONAMENTO



TEMPO DE BUSCA

O tempo de busca é o tempo que o braço de leitura/gravação do disco leva para se mover de onde está agora até onde os dados que precisamos estão guardados.

Para tornar esse tempo mais eficiente, existem algoritmos de escalonamento de disco:

- FCFS
- SSTF
- SCAN
- C-SCAN

TEMPO DE ROTAÇÃO

É o tempo necessário para o disco girar e trazer o setor desejado sob o cabeçote de leitura/gravação após o cabeçote ter sido posicionado na trilha correta.

- Um tempo de rotação mais curto significa que o disco pode encontrar e acessar os dados mais rapidamente, contribuindo para tempos de acesso mais rápidos.

TEMPO DE TRANSFERÊNCIA

Este é o tempo necessário para transferir os dados entre o disco e a memória principal do sistema, uma vez que o cabeçote de leitura/gravação está no local correto.

- Um tempo de transferência menor significa que os dados podem ser lidos ou gravados mais rapidamente

TEMPO DE ACESSO

O tempo de acesso ao disco refere-se ao tempo total necessário para acessar uma determinada informação armazenada em um disco rígido.

- É a soma do tempo de busca + a latência rotacional + o tempo de transferência
  - A latência é basicamente o tempo de espera necessário para que o setor desejado chegue à posição correta

Algoritmos de escalonamento de braço em disco

CONCEITO

- Determinam a ordem em que as requisições de leitura/gravação serão atendidas no disco rígido.

- Minimizam o tempo de acesso total e otimizam o desempenho do disco.

FUNCIONAMENTO GERAL

1. Aplicação solicita acesso a um arquivo ou bloco de dados no disco;
2. As solicitações são colocadas em uma fila de requisições;
3. O algoritmo determina a ordem de processamento dessas solicitações, movendo o braço de leitura/gravação do disco para o cilindro desejado

ALGORITMOS DE ESCALONAMENTO

- FCFS (First-Come, First-Served)
- SSTF (Shortest Seek, Time First)

FCFS (FIRST-COME, FIRST-SERVED)

- No FCFS, as requisições são atendidas na ordem em que chegam na fila do disco;

- O mais simples dos algoritmos de escalonamento de disco, semelhante ao algoritmo FIFO;

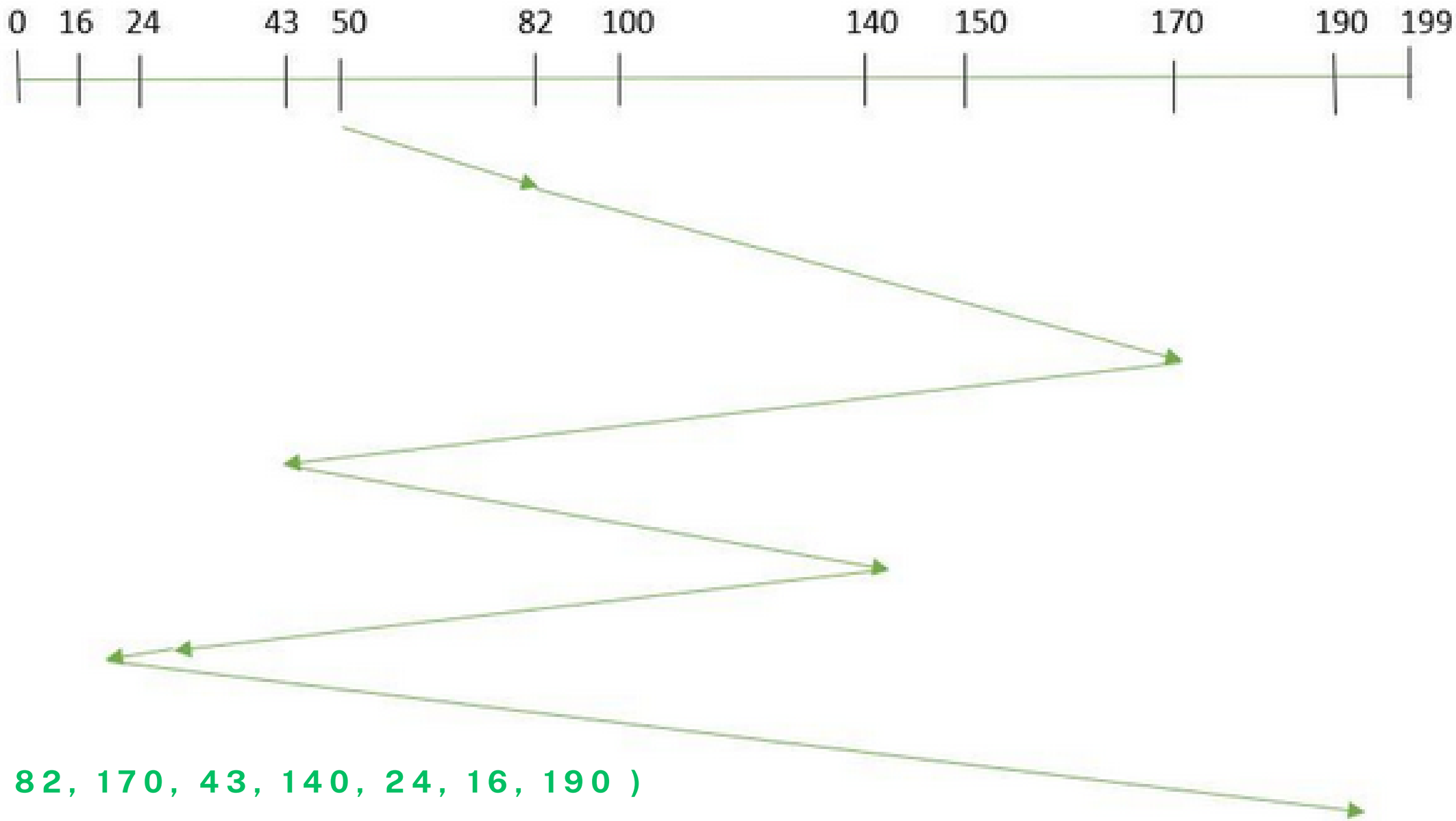
COMO O FCFS FUNCIONA?

1. Se escolhe a primeira posição do vetor de requisições para movimentação do braço.
2. Busca-se a próxima posição do vetor de requisições (independente da distância para a posição atual), e assim sucessivamente, seguindo a ordem do vetor, até a última requisição ser atendida.

EXEMPLO

Ordem de solicitação: (82, 170, 43, 140, 24, 16, 190)

Posição do cabeçote de leitura: (50)



(82, 170, 43, 140, 24, 16, 190)

EXEMPLO

Portanto, podemos calcular o tempo total de busca:

T = (82-50) + (170-82) + (170-43) + (140-43) + (140-24) + (24-16) + (190-16)

T = 642

VANTAGENS

- **Simplicidade:** fácil de implementar e compreender;
- **Justiça:** sem favorecimento de solicitações, pois são atendidas na ordem em que chegam;
- **Baixo overhead:** não há sobrecarga adicional de processamento para determinar a ordem de atendimento das requisições;

DESVANTAGENS

- **Ineficiência:** não considera a localização física das requisições, causando longos tempos de busca.
- **Tempo de espera longo:** requisições são atendidas na ordem de chegada, causando espera considerável, mesmo para solicitações mais curtas.
- **Possível "starvation":** Requisições de curta duração podem esperar indefinidamente se uma requisição de longa duração for atendida primeiro.

SSTF (SHORTEST SEEK, TIME FIRST)

- O SSTF busca sempre minimizar a distância percorrida pelo braço de leitura, portanto, as requisições com o menor tempo de busca, são executadas primeiro.

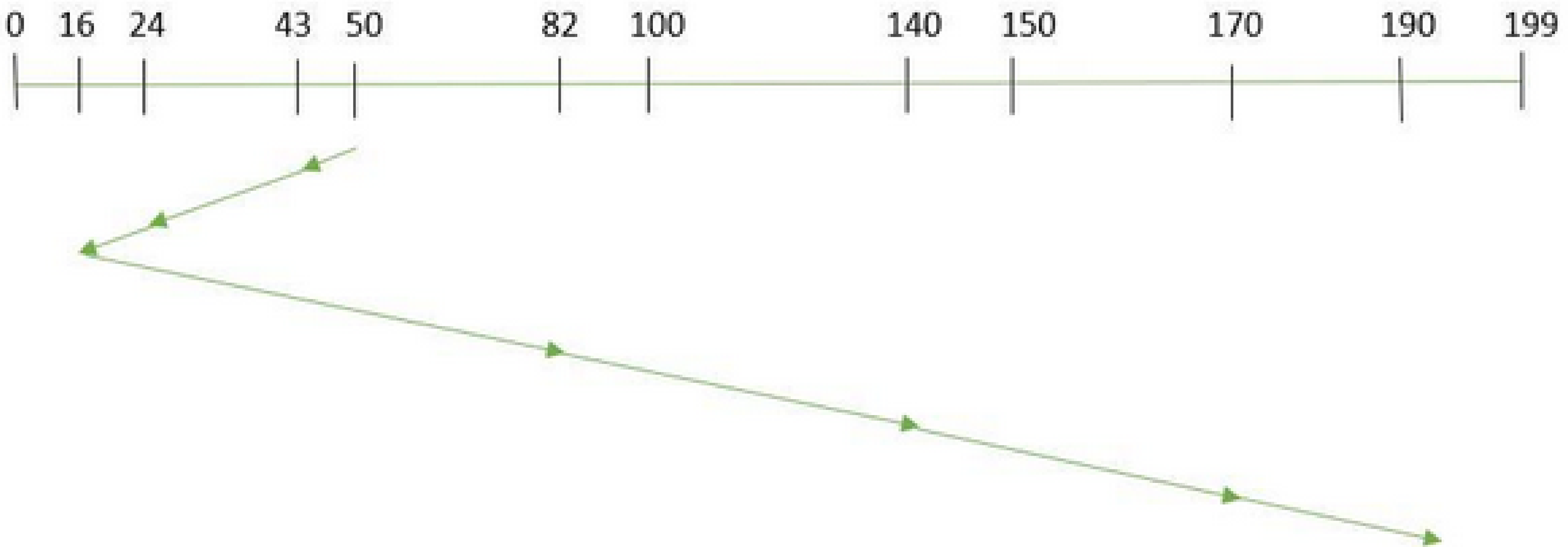
COMO O SSTF FUNCIONA?

1. O algoritmo seleciona a requisição que está mais próxima do cabeçote de leitura/gravação do disco.
2. O cabeçote se move em direção à requisição selecionada, reduzindo o tempo de busca.
3. Após o movimento, a requisição selecionada é atendida.
4. Então a fila de requisições é atualizada para refletir as alterações na posição do cabeçote e nas requisições restantes.

EXEMPLO

Ordem de solicitação: (82, 170, 43, 140, 24, 16, 190)

Posição do cabeçote de leitura: (50)



(82, 170, 43, 140, 24, 16, 190)

EXEMPLO

Portanto, podemos calcular o tempo total de busca:

T = (50-43) + (43-24) + (24-16) + (82-16) + (140-82) + (170-40) + (190-170) + (190-16)

T = 208

VANTAGENS

- **Menor tempo médio de busca:** prioriza requisições mais próximas do cabeçote de leitura/gravação;
- **Eficiência na utilização do disco:** reduz o tempo de busca, resultando no uso mais eficiente do disco;
- **Simplicidade de implementação:** mesmo sendo mais complexo que o FCFS, ainda é mais simples de implementar em comparação com outros algoritmos;

DESVANTAGENS

- **Possivel Starvation:** requisições distantes podem nunca ser atendidas se continuamente chegarem requisições mais próximas;
- **Propenso a ciclos de busca curtos:** o cabeçote pode alternar rapidamente entre requisições próximas, ignorando as mais distantes.
- **Potencial para desempenho irregular:** dependendo da distribuição das requisições e do uso do sistema, o desempenho pode ser inconsistente.

Noop

Implementa o algoritmo FCFS em sistemas **Linux**

Funcionamento

- O escalonador Noop não realiza nenhum reordenamento de requisições de E/S
- Atende as solicitações na ordem que são recebidas
- Isso o torna eficaz em cenários de cargas de trabalho previsíveis ou quando o escalonamento de disco é gerenciado em um nível mais alto (por exemplo, em um sistema de arquivos).

CAM - FreeBSD

É um sistema operacional **Unix-like**, também implementa o escalonamento FCFS, embora com algumas nuances

Funcionamento

- Tradicionalmente usa um escalonador de disco chamado "CAM" (Common Access Method), que geralmente segue uma política FCFS.
- Prioriza as solicitações na ordem em que são recebidas, sem otimizações baseadas em localização física
- O FreeBSD também pode ser configurado para usar outros escalonadores