Sistemas Operacionais

Sistema de Arquivos

Prof. MSc. Rodrigo D. Malara

Sumário

- Introdução
- Arquivos
  - Organização de Arquivos
  - Métodos de Acesso
  - Operações de Entrada/Saída
- Atributos
- Diretórios
- Gerência de Espaço Livre em Disco
- Gerência de Alocação de Espaço em Disco
- Implementação de Caches

Sistema de arquivos

Introdução

- O sistema de arquivos é a parte mais visível do SO
- Cria um recurso lógico a partir de recursos físicos através de uma interface coerente, simples e fácil de usar
- Mecanismos de armazenamento de dados e acesso a programas
- Duas partes básicas
  - Arquivos
    - Armazenamento de dados e programas
  - Diretórios
    - Organização e informações de arquivos

Objetivos do sistema de arquivos

- Fornecer mecanismos para o usuário manipular arquivos e diretórios
- Garantir a validade e coerência dos dados
  - Minimizar ou eliminar o risco de perda / alteração das informações
- Otimizar o acesso
- Fornecer suporte a outros sistemas de arquivos
- Suporte a vários usuários: Uso compartilhado
  - Proteção ou Segurança: Direitos de acesso
  - Acesso concorrente

Requisitos mínimos: ponto de vista do usuário

- Cada usuário deve ser capaz de:
  - Criar, apagar, ler e alterar arquivos
  - Controlar as permissões de acesso a seus arquivos
  - Nomear arquivos de forma simbólica
  - Estruturar os arquivos de forma a adequá-los a suas necessidades especificas
    - Criação de diretórios e subdiretórios
  - Realizar backups e restaurar em caso de perdas

Requisitos mínimos: ponto de vista do sistema

- O SO deve ser capaz de :
  - Descrever a localização de todos os arquivos e seus atributos
    - Via diretório
  - Gerenciar espaço físico de disco
    - Alocar blocos
    - Liberar blocos de arquivos removidos
    - Mecanismos para localizar eficientemente blocos (setores) que compõem arquivos
      - Evitar a fragmentação de arquivos

Arquivos

- Extensão de arquivos

Extensão	Descrição
ARQUIVO.BAS	Arquivo fonte em BASIC
ARQUIVO.COB	Arquivo fonte em COBOL
ARQUIVO.EXE	Arquivo executável
ARQUIVO.OBJ	Arquivo objeto
ARQUIVO.PAS	Arquivo fonte em Pascal
ARQUIVO.TXT	Arquivo texto

Atributos

- Atributos de arquivos

Atributos	Descrição
Tamanho	Especifica o tamanho do arquivo.
Proteção	Código de proteção de acesso.
Dono	Identifica o criador do arquivo.
Criação	Data e hora de criação do arquivo.
Backup	Data e hora do último backup realizado.
Organização	Indica a organização lógica dos registros.
Senha	Senha necessária para acessar o arquivo.

Organização de Arquivos

- Formato texto ou binário
  - Texto – legíveis por humanos sem necessidade de software específico
    - No fundo são seqüências de 0s e 1s
    - Necessitam de codificação
      - EBCDIC – usada em mainframes
      - ASCII ou ISO-8859-1 – obsoleta
        - » MS/DOS – 256 caracteres
        - » Necessita de uma tabela diferente para cada conjunto de caracteres
        - » Inviabiliza compartilhamento de arquivos com tabelas diferentes
        - » Ex: Chines, Russo, Árabe, etc

Organização de Arquivos

- Formato texto ou binário
  - Texto – legíveis por humanos sem necessidade de software específico
    - No fundo são seqüências de 0s e 1s
    - Necessitam de codificação
      - UNICODE
        - » Surgiu em 1986 na XEROX
        - » Unicode Consortium em 1991
        - » Consolida todas as tabelas de c
        - » Permite 136993 códigos distintos
        - » 16 bits variáveis ou 2 palavras de 16 bits
        - » UTF-8 – versão simplificada – 8 bits – 1 a 4 bytes por símbolo

Organização de Arquivos

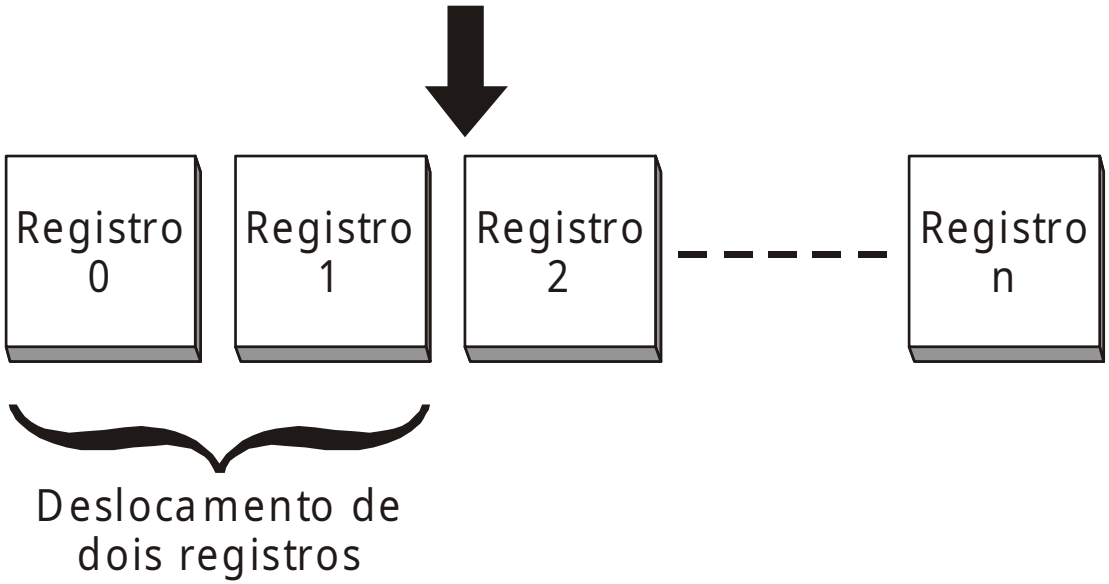
- Formato texto ou binário
  - Binário – é gerado de acordo padrão específico
    - Exemplos
      - PDF – Portable Document Format
      - PNG, JPG, GIF – imagens
      - DOCX, XLSX, … - Word Excel, …
- Mas quem interpreta o arquivo executável ?
  - Sistema Operacional
- Exemplo: Programas feitos em Linguagem C
  - Compilador pode ser o mesmo para diferentes SOs.
    - Desde que perservada a arquitetura (x86 ou x64)
  - Linker é específico para cada S.O. e arquitetura

Métodos de Acesso

- Acesso Sequencial
  - Implementação mais simples
  - O arquivo é processado de forma sequencial
  - Na abertura posicionar ponteiro no início ou no final
    - Modos de abertura em C
  - O ponteiro do arquivo é automaticamente atualizado (incrementado) quando é realizada uma leitura
  - É possível fazer o reposicionamento do ponteiro no início do arquivo
  - Usado para manipular arquivos no formato texto

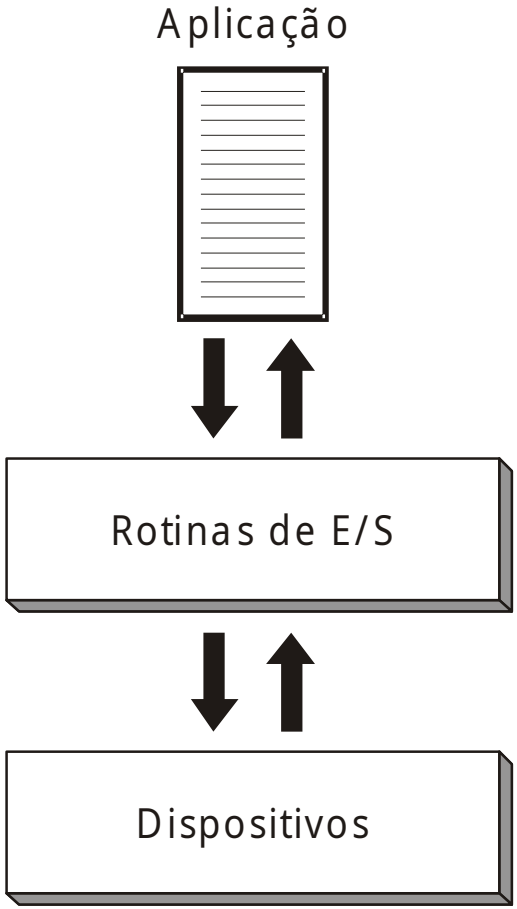
Métodos de Acesso

- Acesso Direto
  - O arquivo composto por “registros” de tamanho fixo ou deslocamento de 1 em 1 byte
  - Uma operação de leitura e escrita é realizada diretamente em um endereço n
  - Acesso randômico



Operações de Entrada/Saída

- Operações de Entrada/Saída



Operações de Entrada/Saída

- Rotinas de E/S sobre arquivos

Rotina	Descrição
CREATE	Criação de arquivos.
OPEN	Abertura de um arquivo.
READ	Leitura de um arquivo.
WRITE	Gravação em um arquivo.
CLOSE	Fechamento de um arquivo.
DELETE	Eliminação de um arquivo.

Diretórios

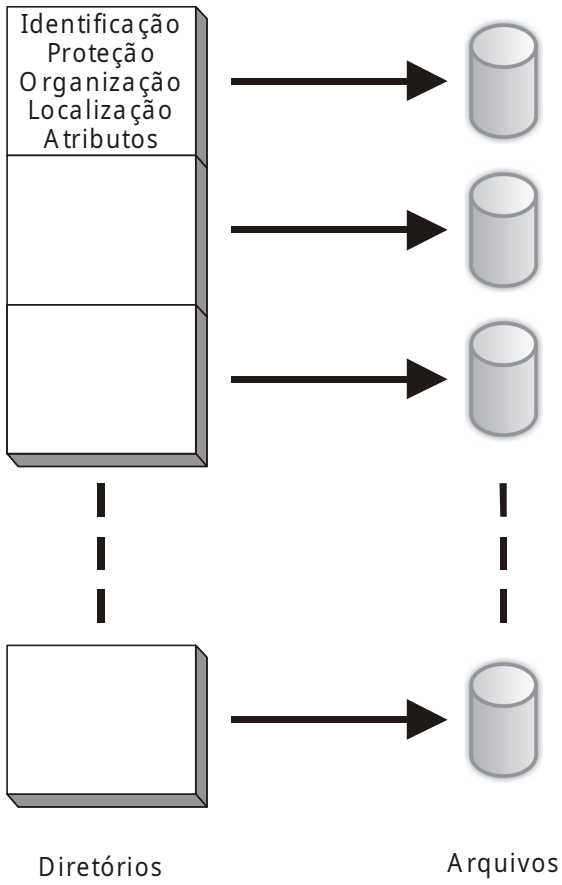
- Directory = Lista telefônica (em inglês)
- Contém informações dos arquivos armazenados no disco
- Cada disco ou partição contém uma estrutura de diretórios
- Operações
  - Buscar um arquivo
  - Criar um arquivo
  - Apagar um arquivo
  - Listar os arquivos
  - Renomear um arquivo
  - Verificar o conteúdo do sistema de arquivo

Diretórios

- Eficiência: localizar um arquivo rapidamente
- Nomes: apropriado para usuários
- Agrupamento
  - Arquivos pertencentes a uma mesma aplicação são organizados através dos diretórios

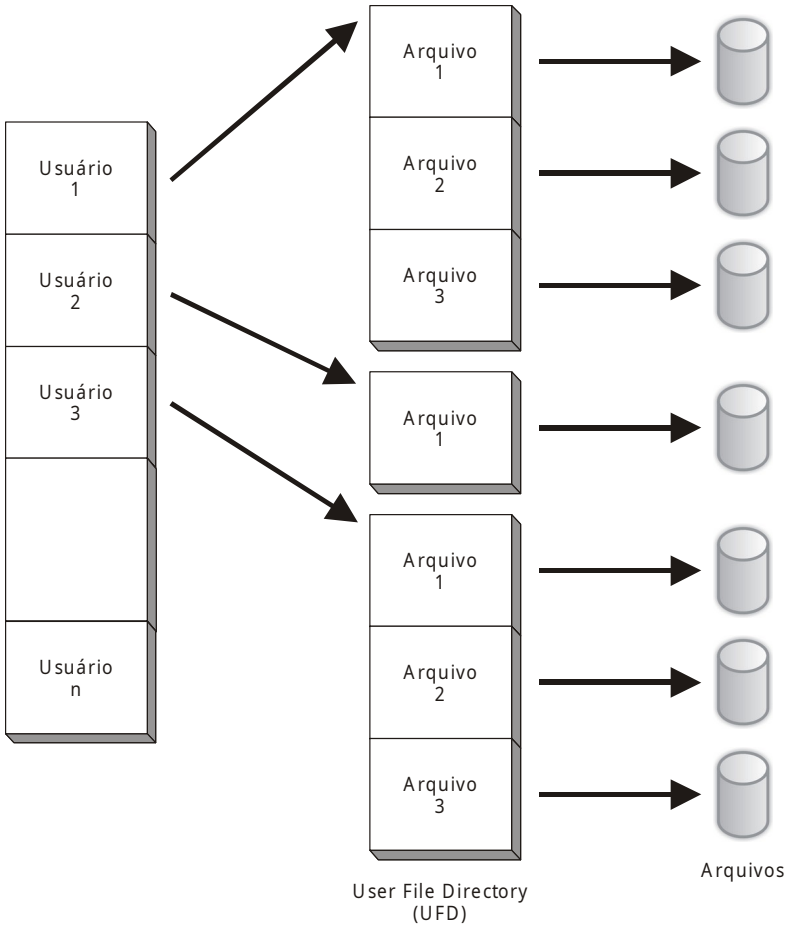
Diretórios

- Estrutura de diretórios de nível único
- Único nível para todos os usuários
- Fácil implementação
- Problemas: conflitos de nomes



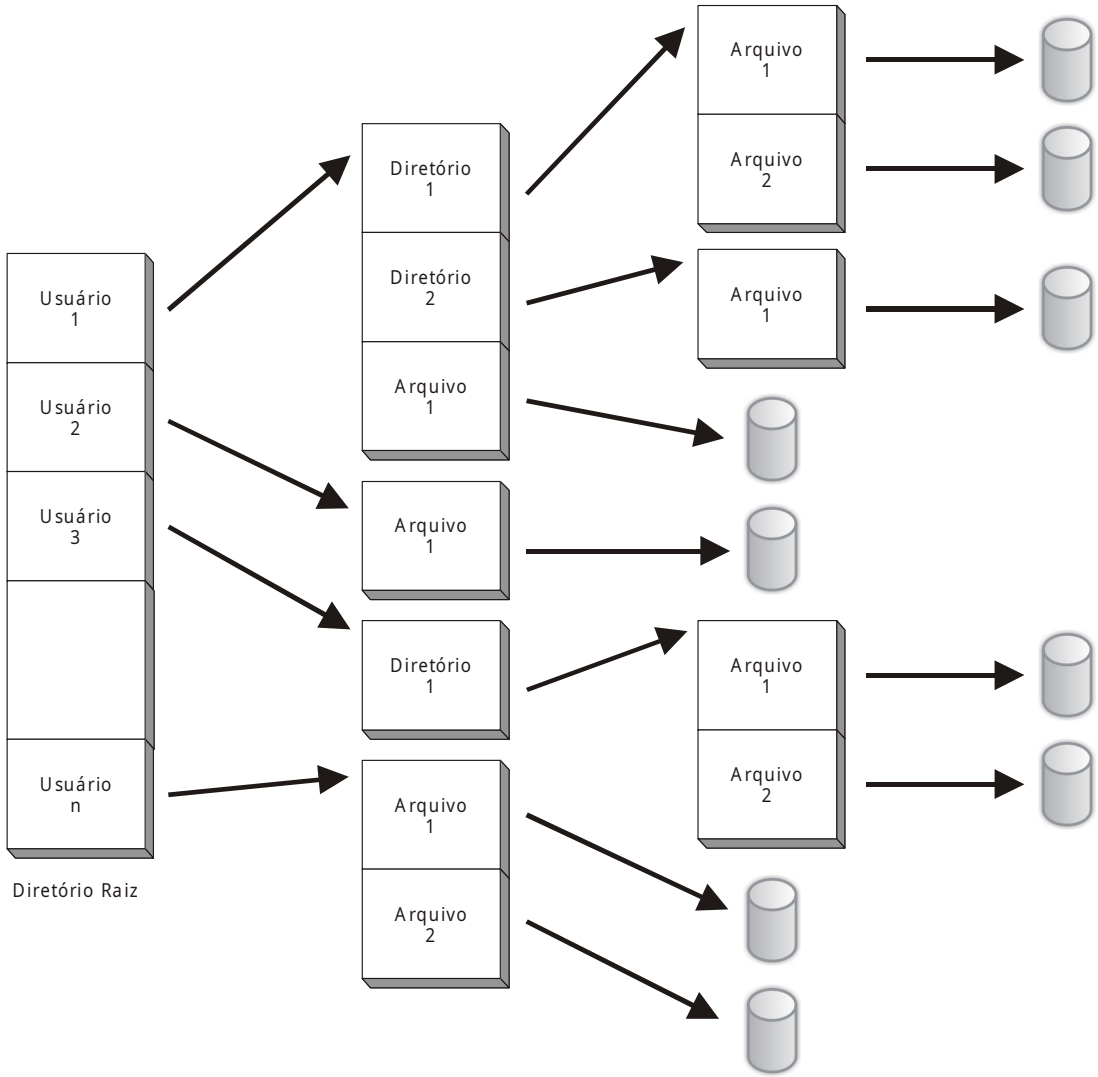
Diretórios

- Estrutura de diretórios com dois níveis
- Cada usuário tem o seu diretório
- Usuários podem ter arquivos com o mesmo nome
- Nomes de arquivos compostos por caminhos (path)
- Busca mais eficiente



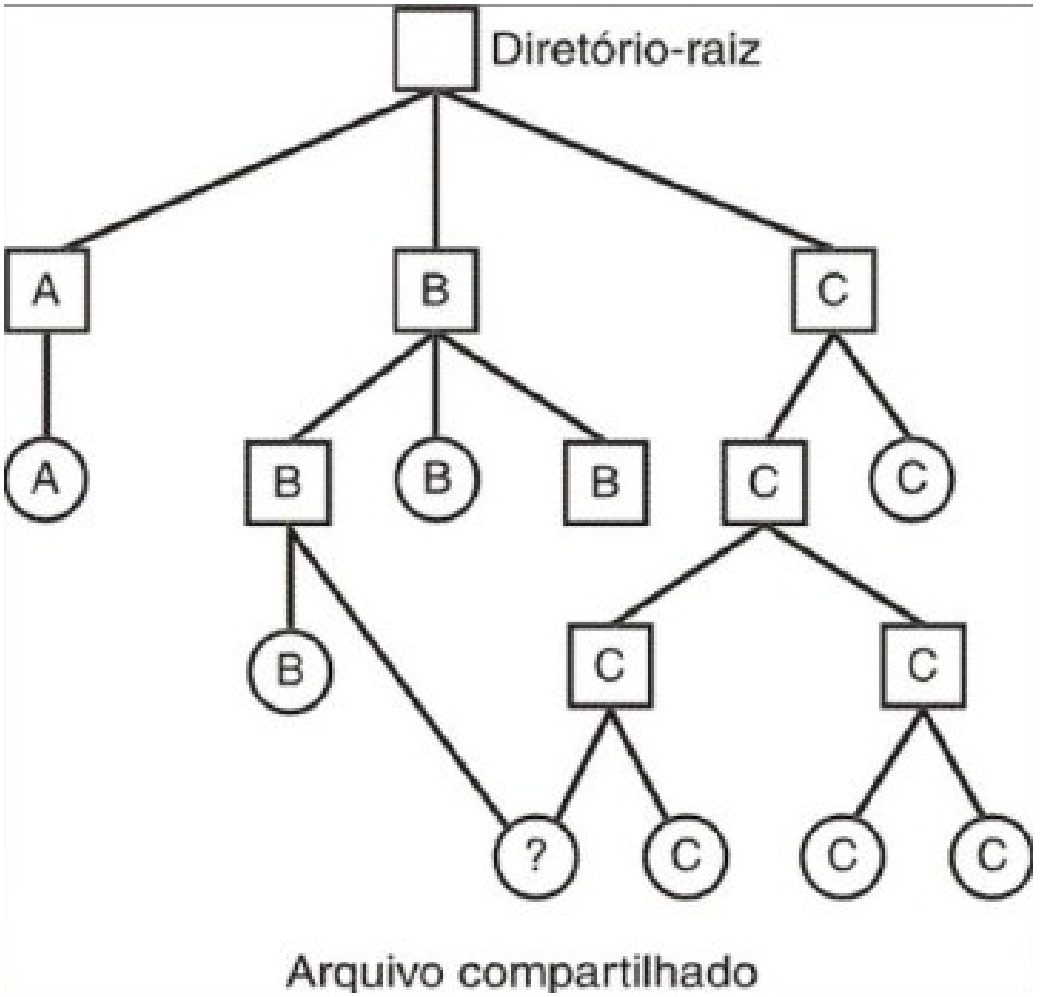
Diretórios

- Estrutura de diretórios em árvore
- Agrupamento
- Busca mais eficiente
- Conceito de diretório corrente ao executar um programa



Diretórios

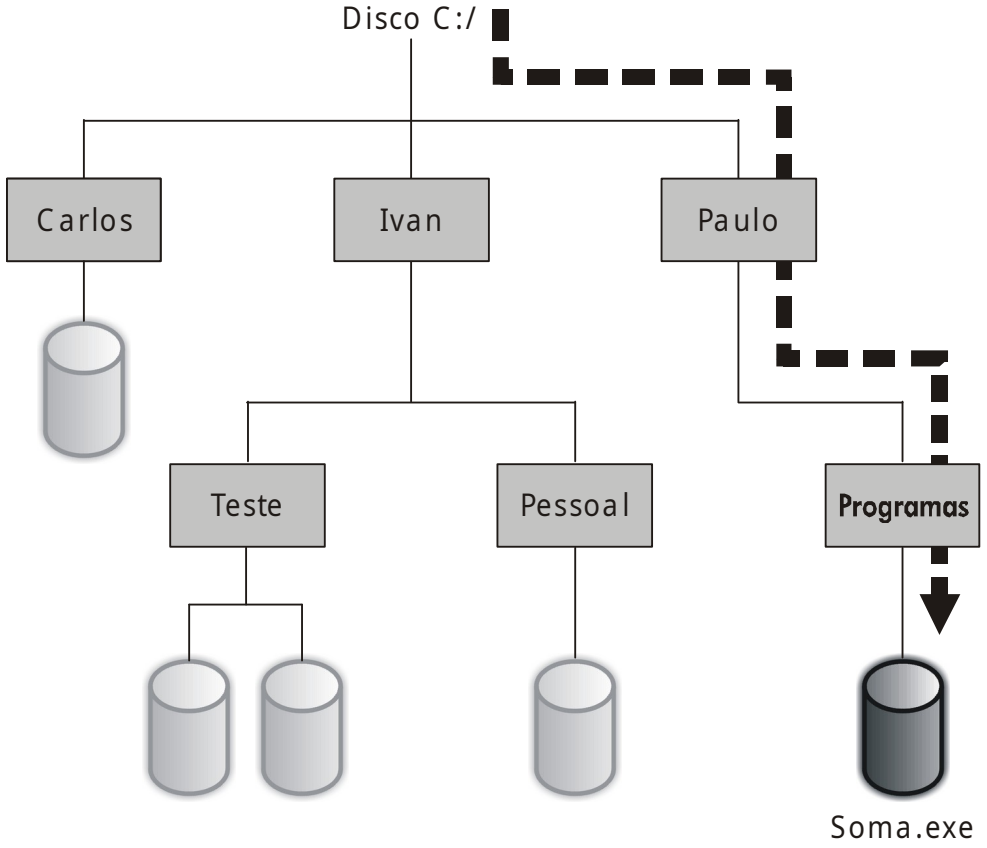
- Estrutura em Grafo Acíclico
  - Diretórios são quadrados
  - Arquivos círculos
- O arquivo ? Acessível de 2 diretórios diferentes
- Nomes podem ser diferentes



Arquivo compartilhado

Diretórios

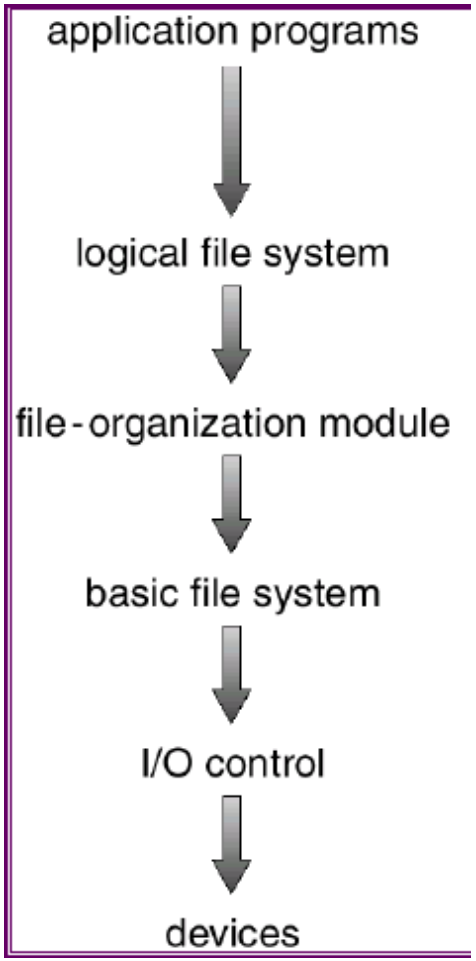
- Path de um arquivo
- Também chamado de caminho



Implementação do sistema de arquivos

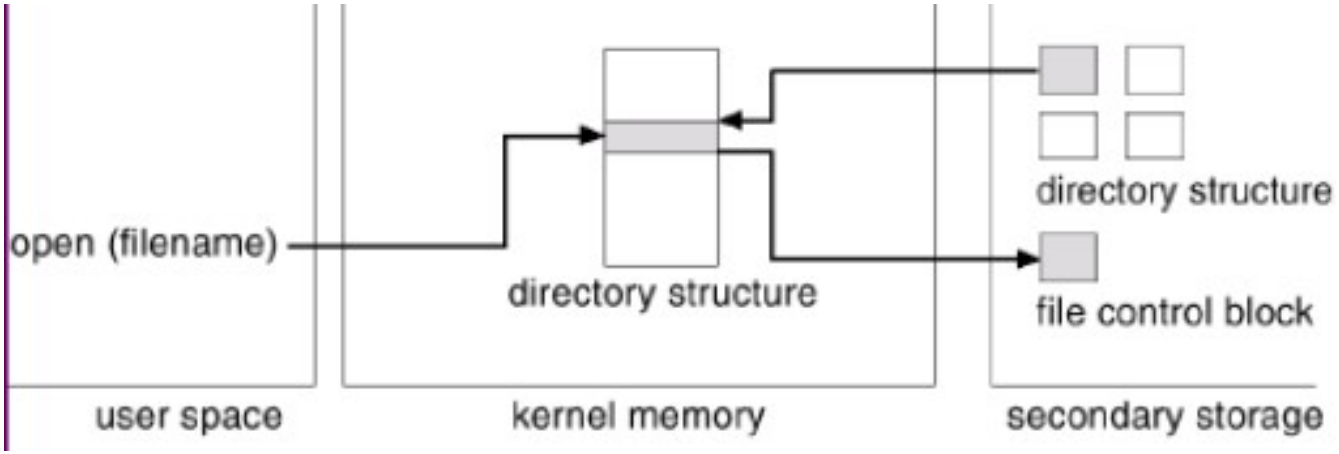
Sistema de arquivos

- Organizado em camadas
- **Bloco de Controle do Arquivo** (file control block) ou **BCA**: estrutura de dados que armazena as informações do arquivo
  - Permissões
  - Acessos (data/hora)
  - Dono / Grupo
  - Tamanho
  - Endereços dos blocos do arquivo em disco



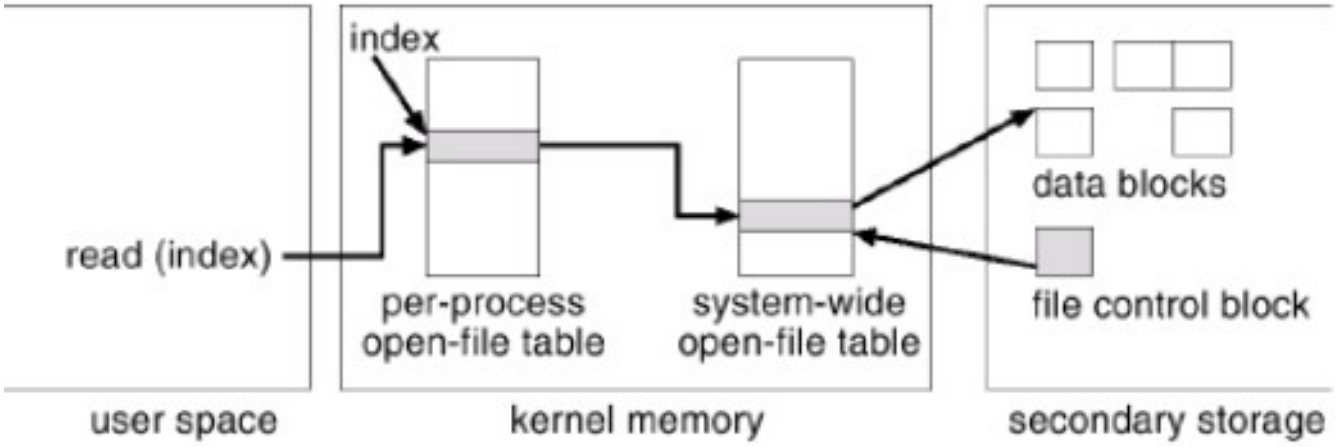
Estrutura de dados no acesso a arquivos

- Ao **abrir** um arquivo o bloco de controle de arquivo é acessado
  - Arquivo disponível?
  - Usuário tem direitos de acesso?



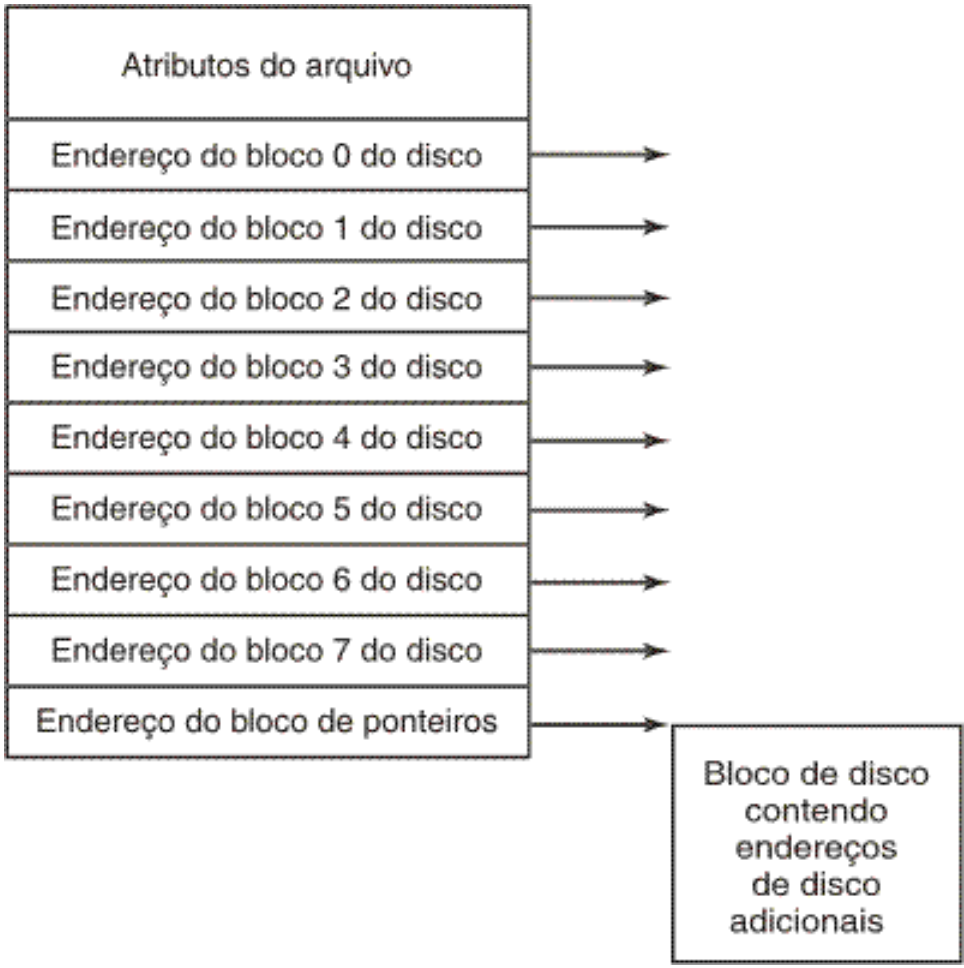
Estrutura de dados no acesso a arquivos

- Ao **ler** de um arquivo o bloco de controle de arquivo é acessado
  - Informa onde ficam os blocos do arquivo no HD



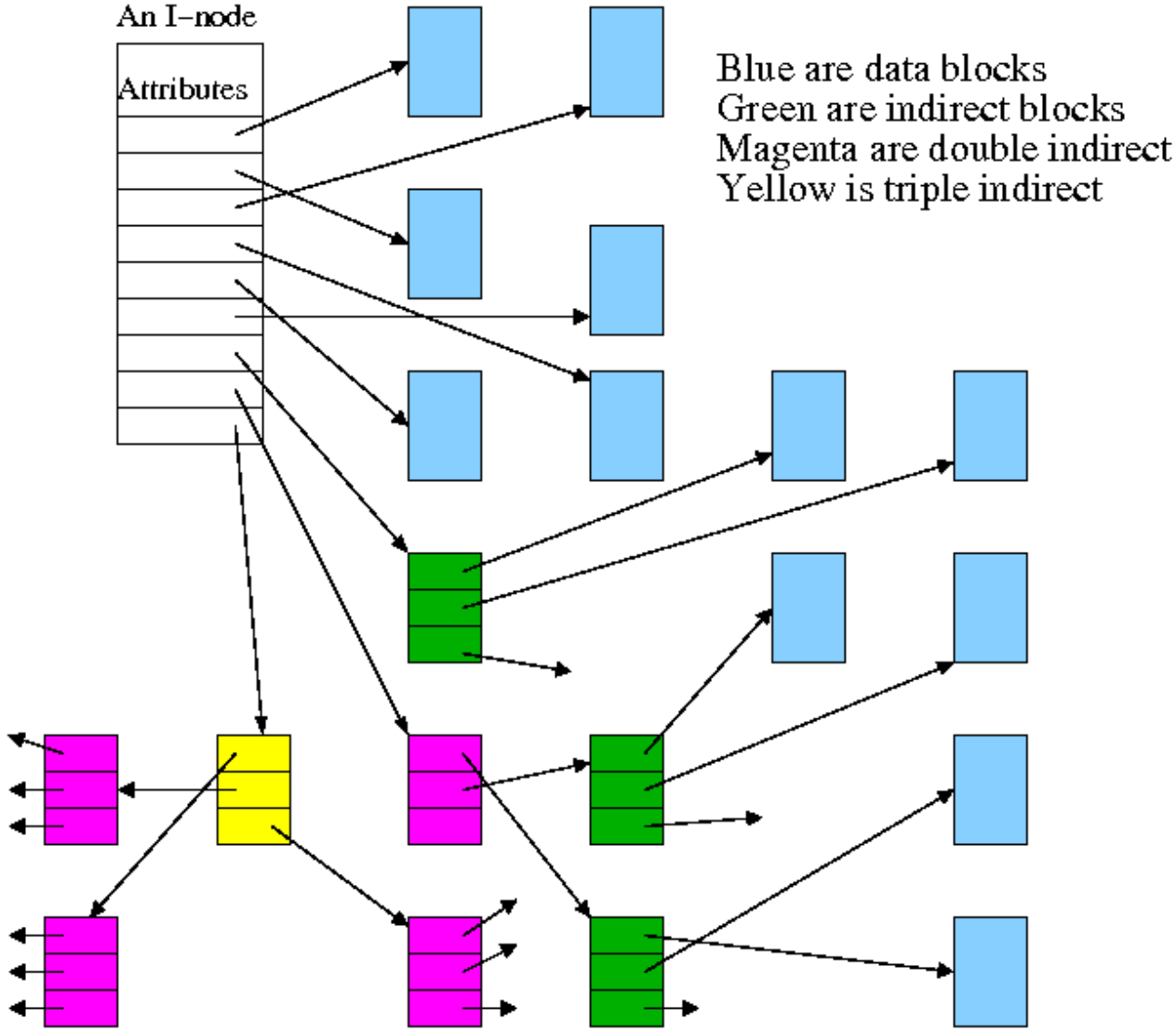
Estrutura de um I-Node

- No UNIX o BCA é chamado de I-Node
  - Um I-Node pode armazenar o endereço de até 12 blocos
  - Para arquivos grandes, um I-Node pode apontar para outro



Estrutura de um I-Node

- Branco
  - I-Node
- Azul
  - Blocos de Dados
- Verde
  - I-Nodes indiretos
- Amarelo
  - I-Nodes duplamente indiretos
- Lilás
  - I-Nodes triplamente indiretos



Vantagens dos I-nodes

- Otimizado para sistemas de arquivos com muitos arquivos pequenos
- Cada I-node pode apontar diretamente para 48KB de dados
- Apenas uma camada indireta é necessário para arquivos de 4MB
- Acesso a arquivos mais rápido
- Maior localidade meta-dados → menos busca aleatória
- Não há necessidade de atravessar entradas FAT de cadeia longa
- Facilitar o gerenciamento do espaço livre
- Bitmaps podem ser armazenadas na memória para acesso rápido
- I-node e espaço de dados manipulados independentemente

Exemplo acesso arquivos em C

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>

void main () {
   printf (“Cópia de arquivo texto.\n");
   FILE *arqEntrada, *arqSaida;
   int ch;
   arqEntrada = fopen ("existente.txt", "r"); // Obtem BCA
   arqSaida = fopen ("novo.txt", "w"); // Obtem BCA

   printf ("Copiar de caracteres .\n");
   while ((ch = fgetc (arqEntrada)) != EOF) { // Lendo do arquivo
       fputc (ch, arqSaida); // Escrevendo no novo arquivo
   }
   fclose (arqEntrada); // Fecha BCA
   fclose (arqSaida);  // Fecha BCA
}
```

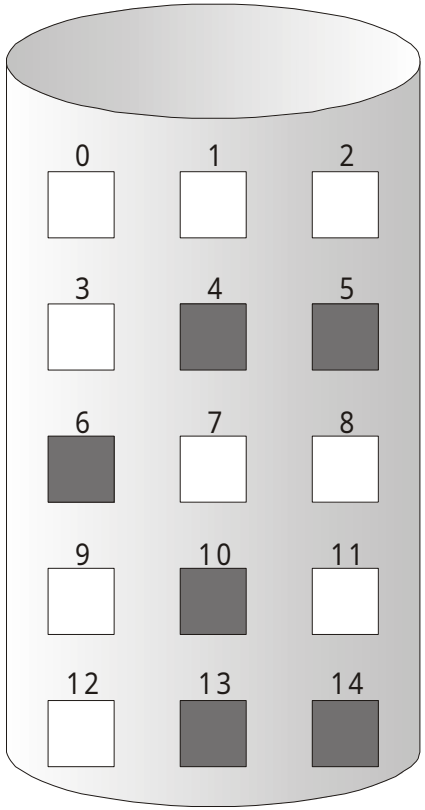
Métodos de alocação

Alocação contígua

- Cada arquivo contém um conjunto de blocos alocados de forma contígua no disco
- Armazenar apenas o bloco inicial e o número de blocos do arquivo
- Acesso randômico
- Problema da alocação dinâmica
- Aumento de tamanho do arquivo?
  - Não pode crescer
  - Alocar um novo espaço

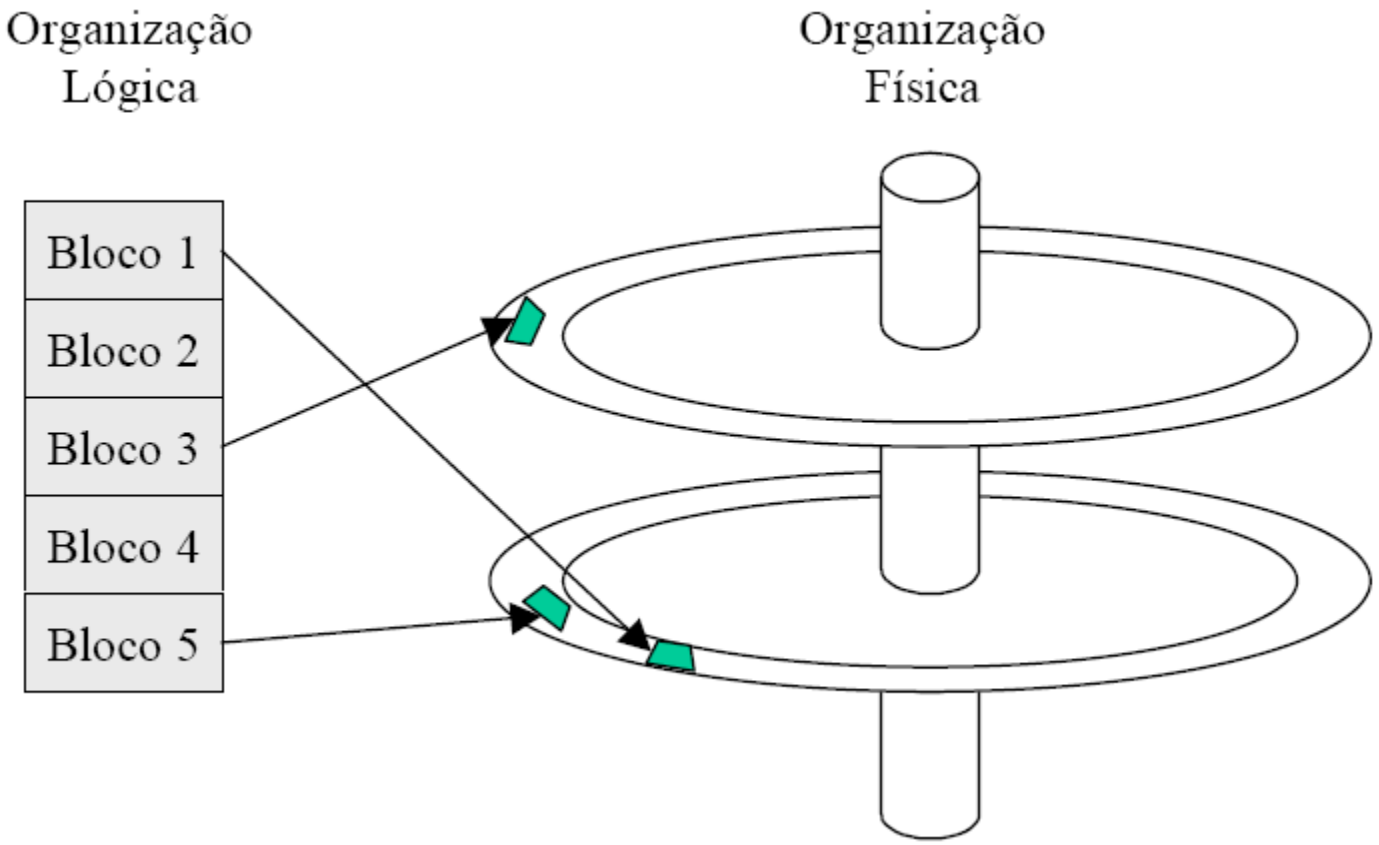
Gerência de Alocação de Espaço em Disco

- Alocação Contígua



Arquivo	Bloco	Extensão
A. TXT	4	3
B. TXT	10	1
C. TXT	13	2

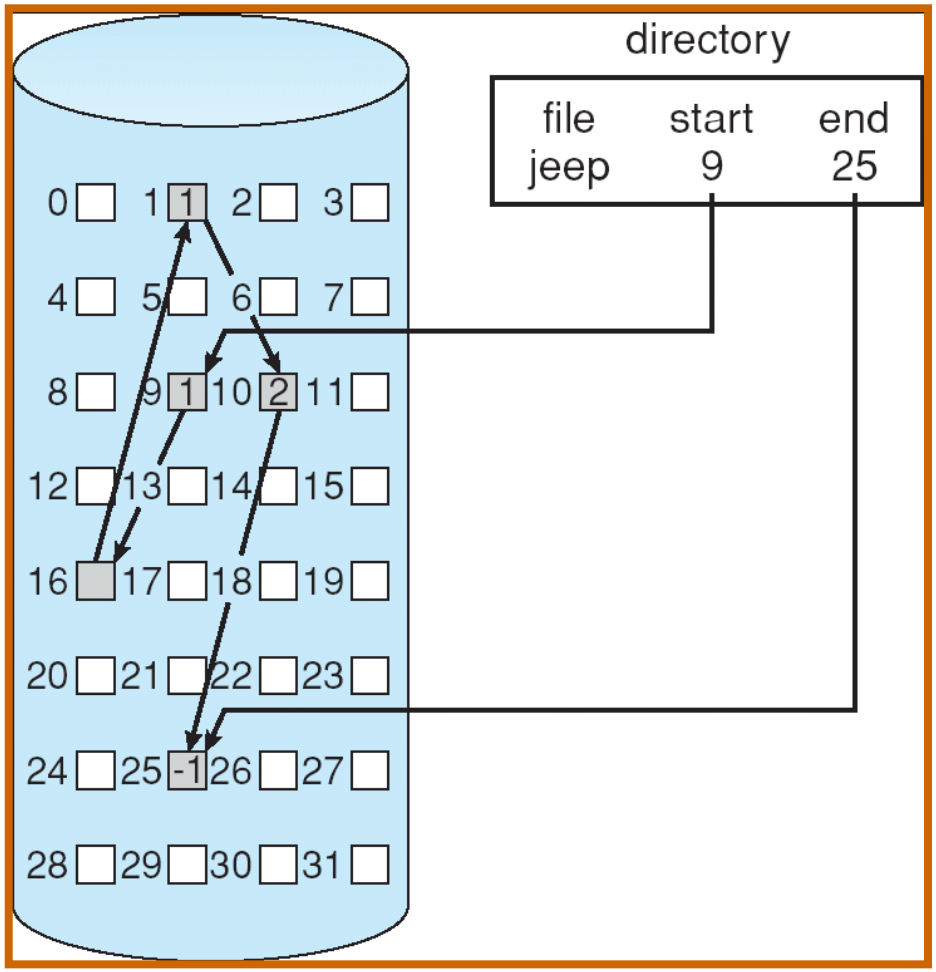
Alocação não contígua



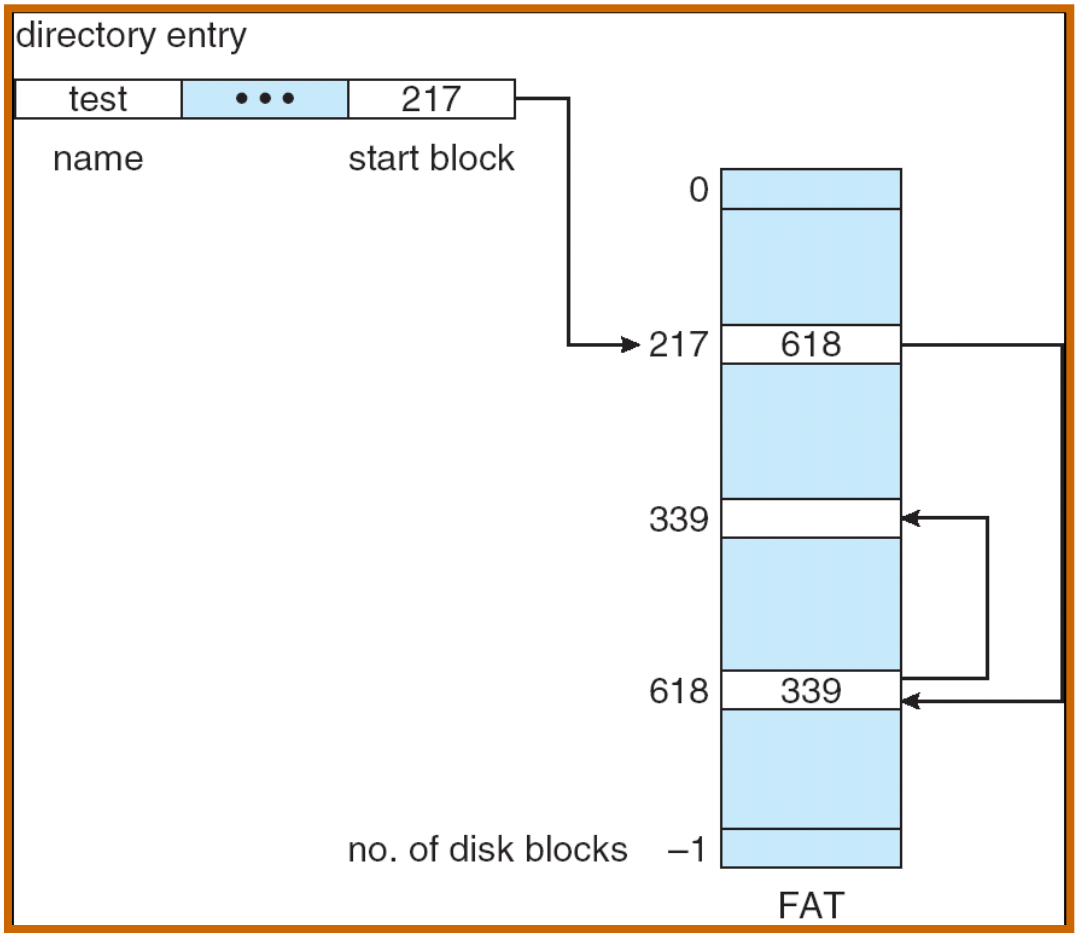
Alocação – lista ligada

- Cada arquivo composto por uma lista ligada de blocos do disco
- Não precisa ser contígua
- Armazena apenas o bloco inicial
- Sem acesso randômico

Alocação – lista ligada

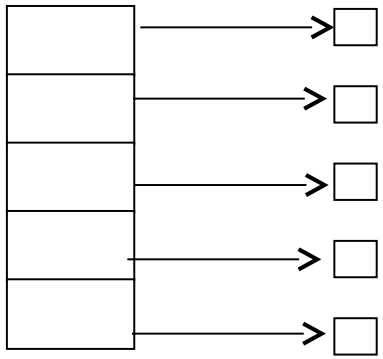


FAT – file allocation table



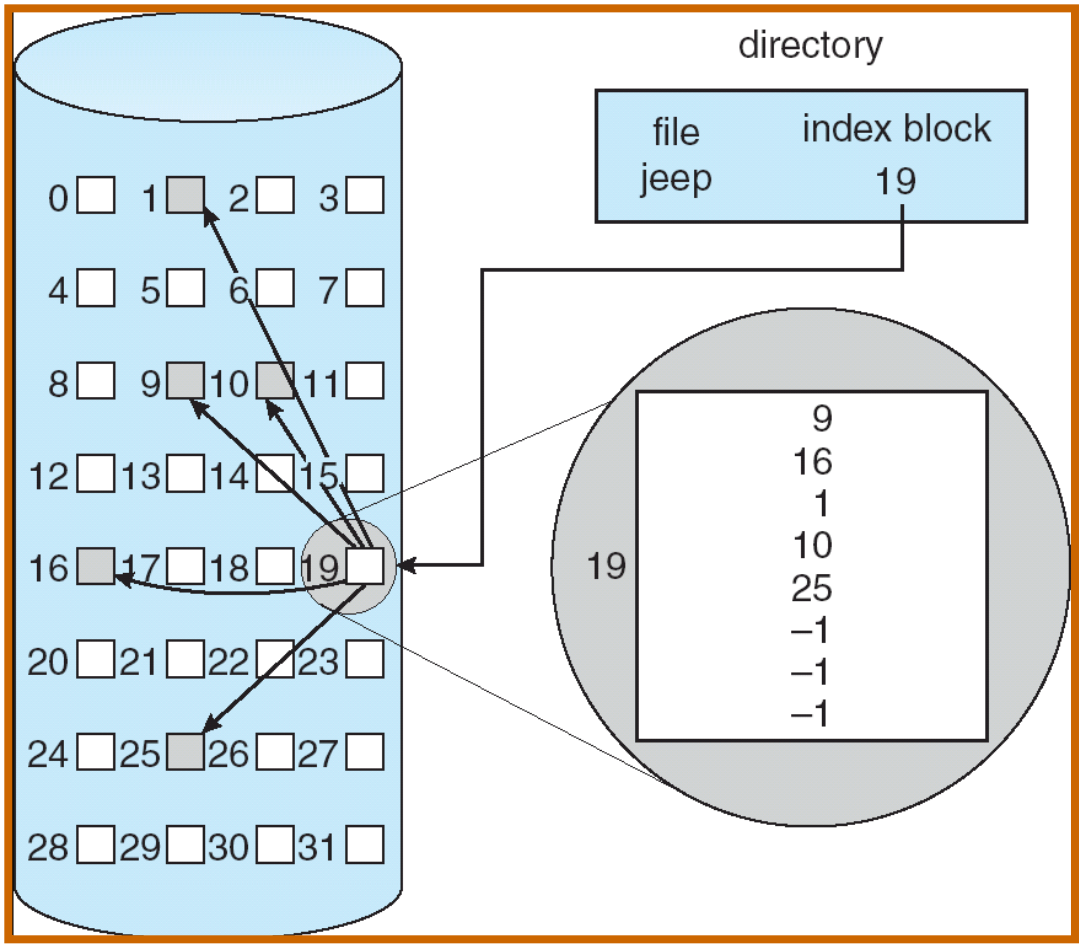
Alocação indexada

- Todos os ponteiros para o arquivo são armazenados em uma tabela
  - Tabela de índices
  - I-Nodes
- Viabiliza acesso randômico



index table

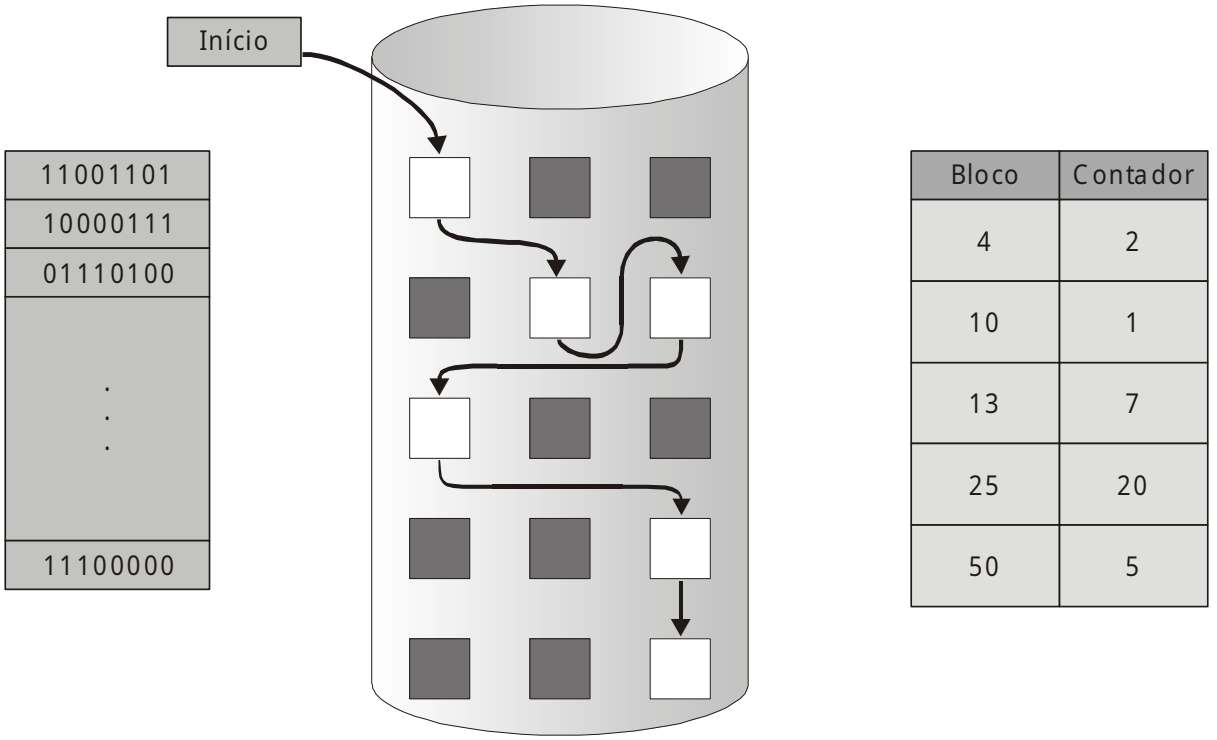
Alocação indexada



Gerenciamento de espaços livres

Gerência de Espaço Livre em Disco

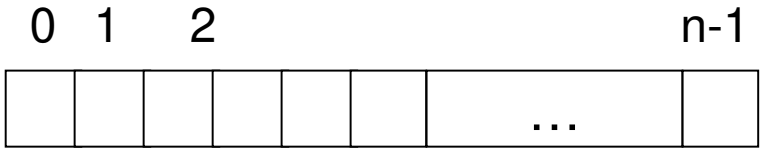
- Alocação de espaço em disco



(a) Mapa de bits (b) Lista encadeada © Tabela de blocos livres

Mapa de bit

- Cada bit representa um bloco do disco
- Calculo do bloco livre
  - (número de bits por palavra) * (número de 0) + deslocamento até o primeiro bit 1
- É necessário armazenar os mapas de bits no disco
  - Qual o overhead?
- Fácil obtenção de um espaço contíguo
- Performance ruim

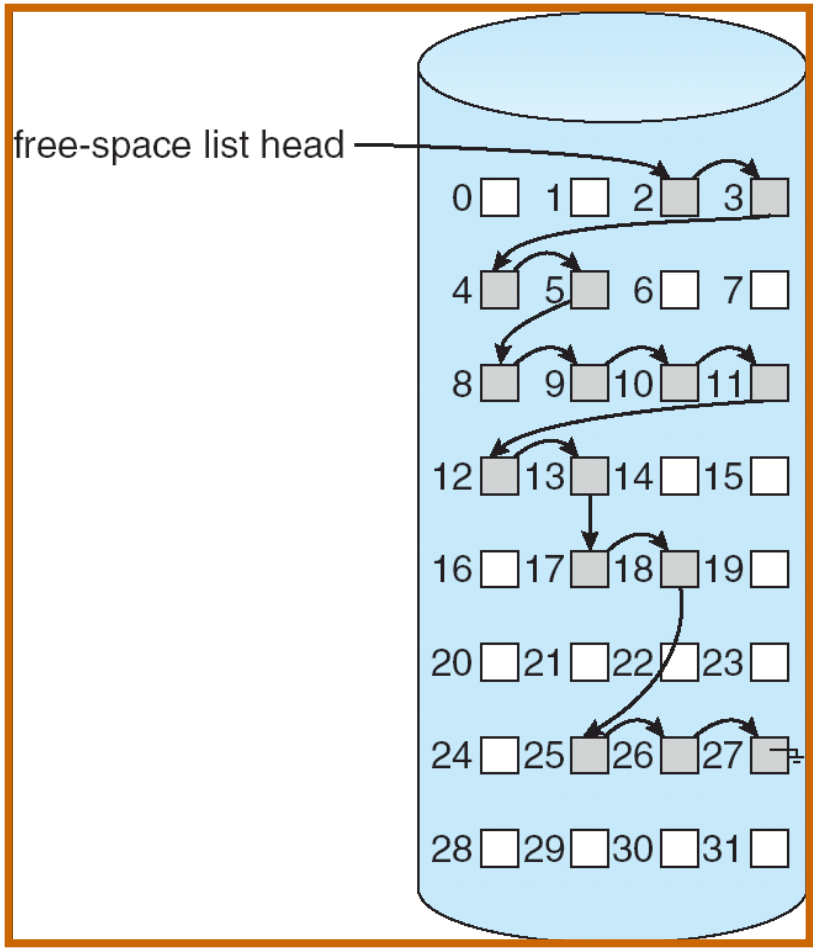


Lista ligada

- Espaço livre: Armazenar apenas o ponteiro do início da lista
  - Sem perda de espaço
- Mais rápido que mapas de bits
- Essa técnica é a mesma utilizada pelo gerenciador de memória, abordado anteriormente

Lista ligada

- Blocos em Cinza estão disponíveis
- São quatro blocos livres:
  - De 2 a 5
  - De 8 a 13
  - De 17 a 18
  - De 25 a 27
- Lista ligada com 4 nós



O Papel de Caches e Buffers

- Acesso a disco é bastante lento
- Buffers e caches minimizam este problema
- Quando uma operação de **leitura** é realizada o sistema verifica se a informação se encontra no cache
  - Em caso positivo, não é necessário o acesso ao disco
  - Caso o bloco requisitado não se encontre no cache, a operação de E/S é realizada e o cache é atualizado
- Quando uma operação de **escrita** é realizada a mudança é feita no buffer e assim que for conveniente aplica-se ao disco rígido
  - Quando o buffer estiver cheio
  - Quando passar um tempo de esvaziamento do buffer (flush)