SISTEMAS OPERACIONAIS

Sistemas de Arquivos

Funções de um SO

- Gerência de processos
- Gerência de memória
- **Gerência de Arquivos**
- Gerência de I/O
- Sistema de Proteção

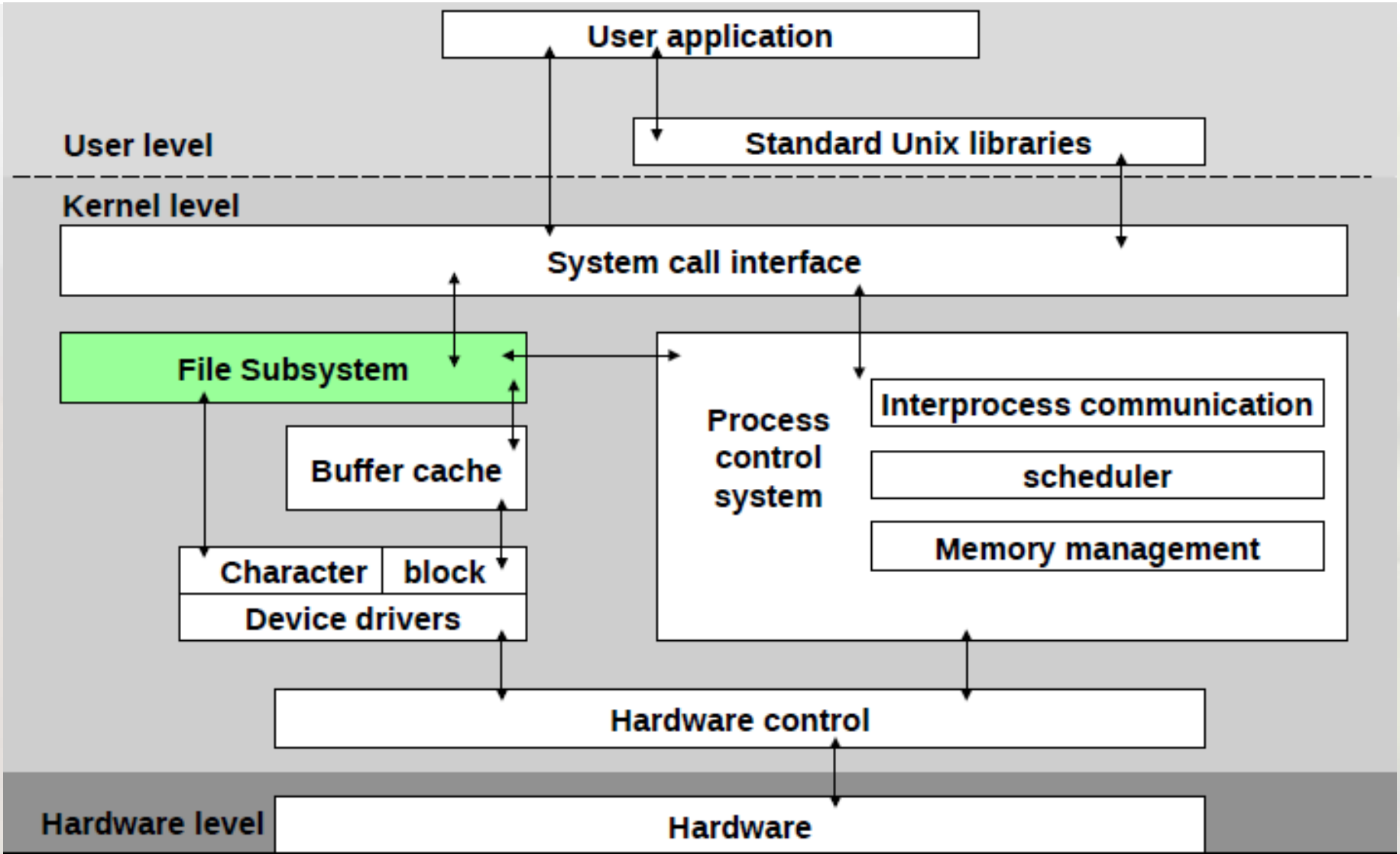
Necessidade de Armazenamento

- Grandes quantidades de informação têm de ser armazenadas
- Informação armazenada tem de sobreviver ao fim do processo que a utiliza
- Múltiplos processos devem poder acessar a informação de um modo concorrente
- ARQUIVO
  - Abstração criada pelo S.O. para gerenciar e representar os dados

Gerência de Arquivos

- Oferece a abstração de arquivos (e diretórios)
- Atividades suportadas
  - Primitivas para manipulação (chamadas de sistema para manipulação de arquivos)
    - criar, deletar
    - abrir, fechar
    - ler, escrever
    - posicionar
- Mapeamento para memória secundária

Estrutura Interna do Kernel UNIX

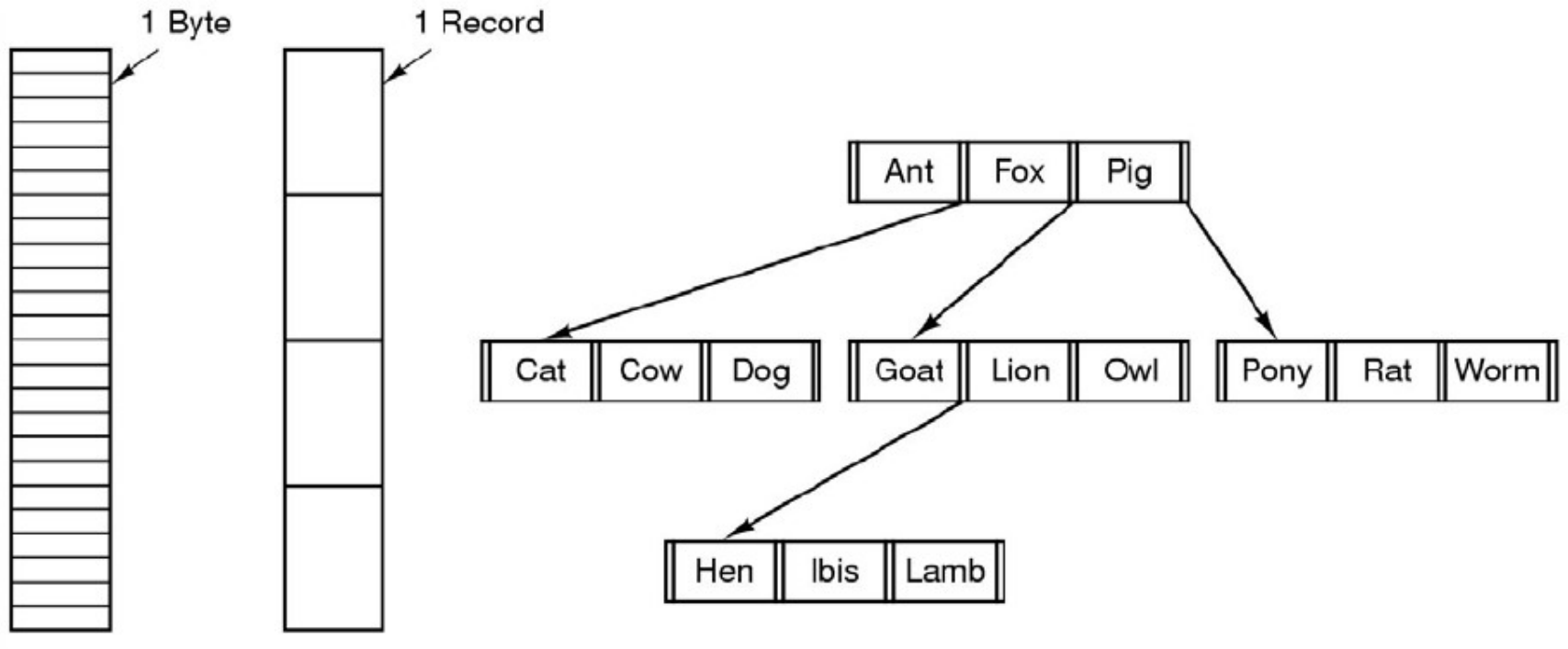


Sistema de Arquivos

- O que é?
  - Um conjunto de arquivos, diretórios, descritores e estruturas de dados auxiliares gerenciados pelo sub sistema de gerência de arquivos
  - Permitem estruturar o armazenamento e a recuperação de dados persistentes em um ou mais dispositivos de memória secundária (discos ou bandas magnéticas)

- Arquivo
  - Um conjunto de dados persistentes, geralmente relacionados, identificado por um nome
  - É composto por:
    - Nome: identifica o arquivo perante o utilizador
    - Descritor de arquivo: estrutura de dados em memória secundária com informação sobre o arquivo (dimensão, datas de criação, modificação e acesso, dono, autorizações de acesso)
    - Informação: dados guardados em memória secundária

Estrutura Interna de Arquivos (1)



Estrutura Interna de Arquivos (2)

- Seqüência não-estruturada de bytes
  - Forma mais simples de organização de arquivos
  - Sistema de arquivos não impõe nenhuma estrutura lógica para os dados, a aplicação deve definir toda a organização
  - Vantagem: flexibilidade para criar estruturas de dados, porém todo o controle de dados é de responsabilidade da aplicação
  - Estratégia adotada tanto pelo UNIX quanto pelo Windows

Estrutura Interna de Arquivos (3)

- Seqüência de Registros
  - Em geral, registros de tamanho fixo
  - Operação de leitura retorna um registro
  - Operação de escrita sobrepõe/anexa um registro

- Árvore de Registros
  - Cada registro é associado a uma chave
  - Árvore ordenada pela chave
  - Computadores de grande porte / aplicações que fazem muita leitura aleatória

Tipos de Arquivos (1)

- Arquivos Regulares
  - Arquivos ASCII
  - Binários
    - Apresentam uma estrutura interna conhecida pelo S.O.
- Diretórios
  - Arquivos do sistema
  - Mantêm a estrutura do Sistemas de Arquivos

Operações sobre Arquivos

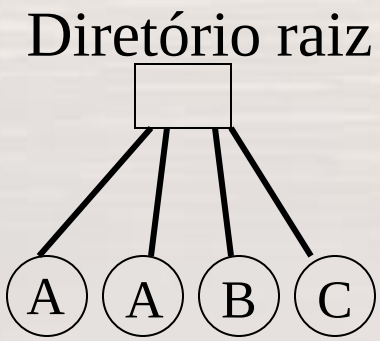
- Dependem do tipo
  - create
  - delete
  - open
  - close
  - read
  - write
  - append
  - seek
  - get attributes
  - set attributes
  - rename

Diretórios (1)

- Modo como o sistema organiza os diferentes arquivos contidos num disco
- É a estrutura de dados que contém entradas associadas aos arquivos
  - localização física, nome, organização e demais atributos
- Quando um arquivo é aberto, o sistema operacional procura a sua entrada na estrutura de diretórios
- As informações do arquivo são armazenadas em uma tabela mantida na memória principal (tabela de arquivo abertos)
  - Fundamental para aumentar o desempenho das operações com arquivos

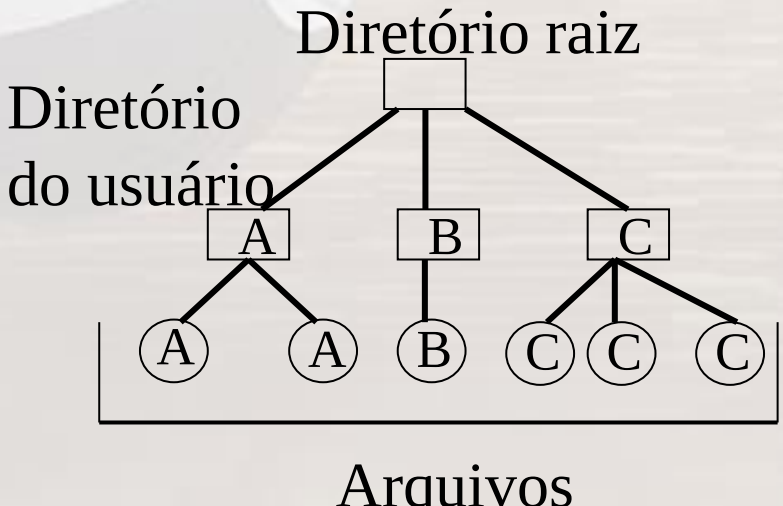
Diretórios (2)

- Sistemas de Diretório em Nível Único
  - Implementação mais simples
  - Existe apenas um único diretório contendo todos os arquivos do disco
  - Bastante limitado já que não permite que usuários criem arquivos com o mesmo nome
    - Isso ocasionaria um conflito no acesso aos arquivos



Diretórios (3)

- Estrutura de diretórios com dois níveis
  - Para cada usuário existe um diretório particular e assim poderia criar arquivos com qualquer nome.
  - Deve haver um nível de diretório adicional para controle que é indexado pelo nome do usuário
    - Cada entrada aponta para o diretório pessoal.

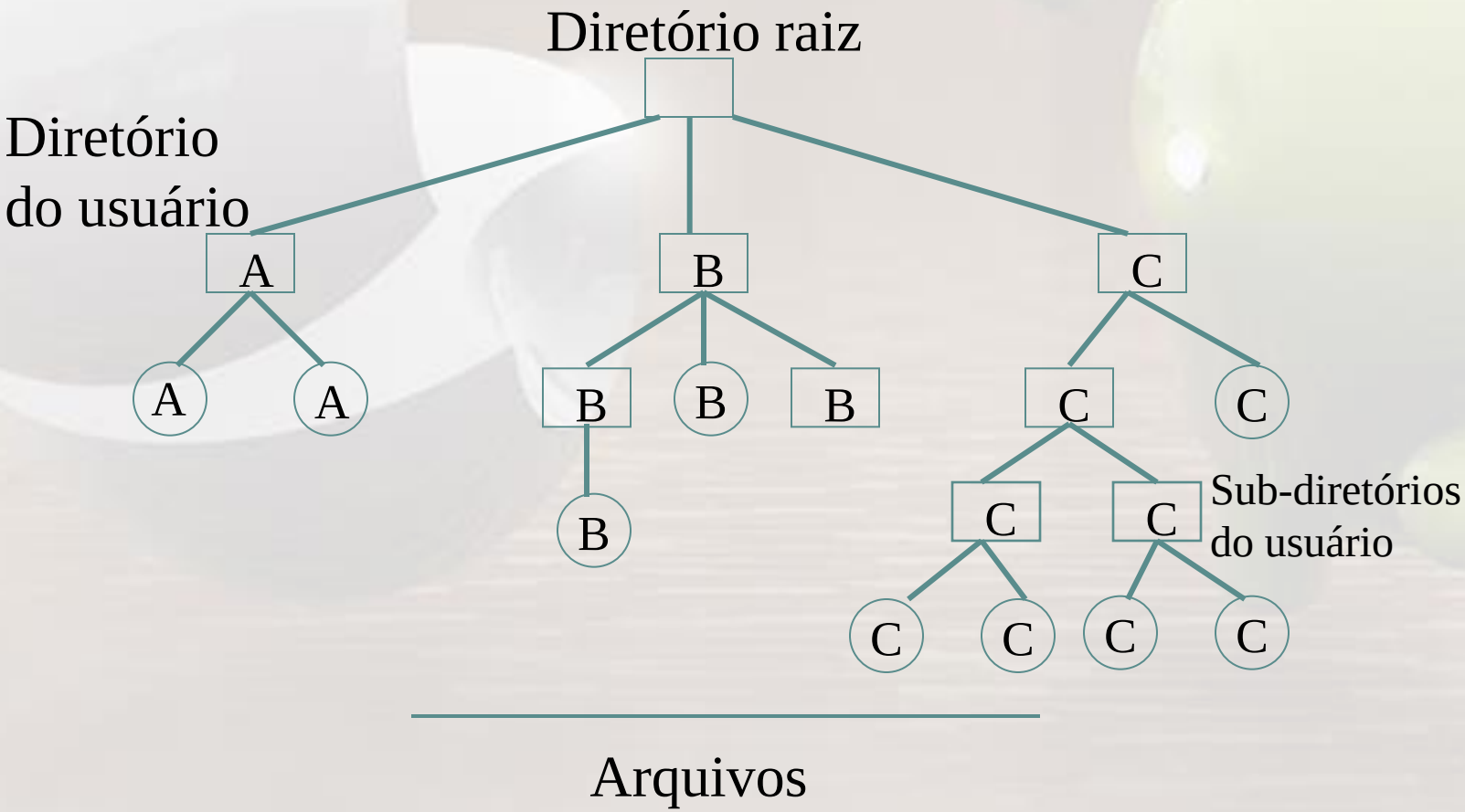


Diretórios (4)

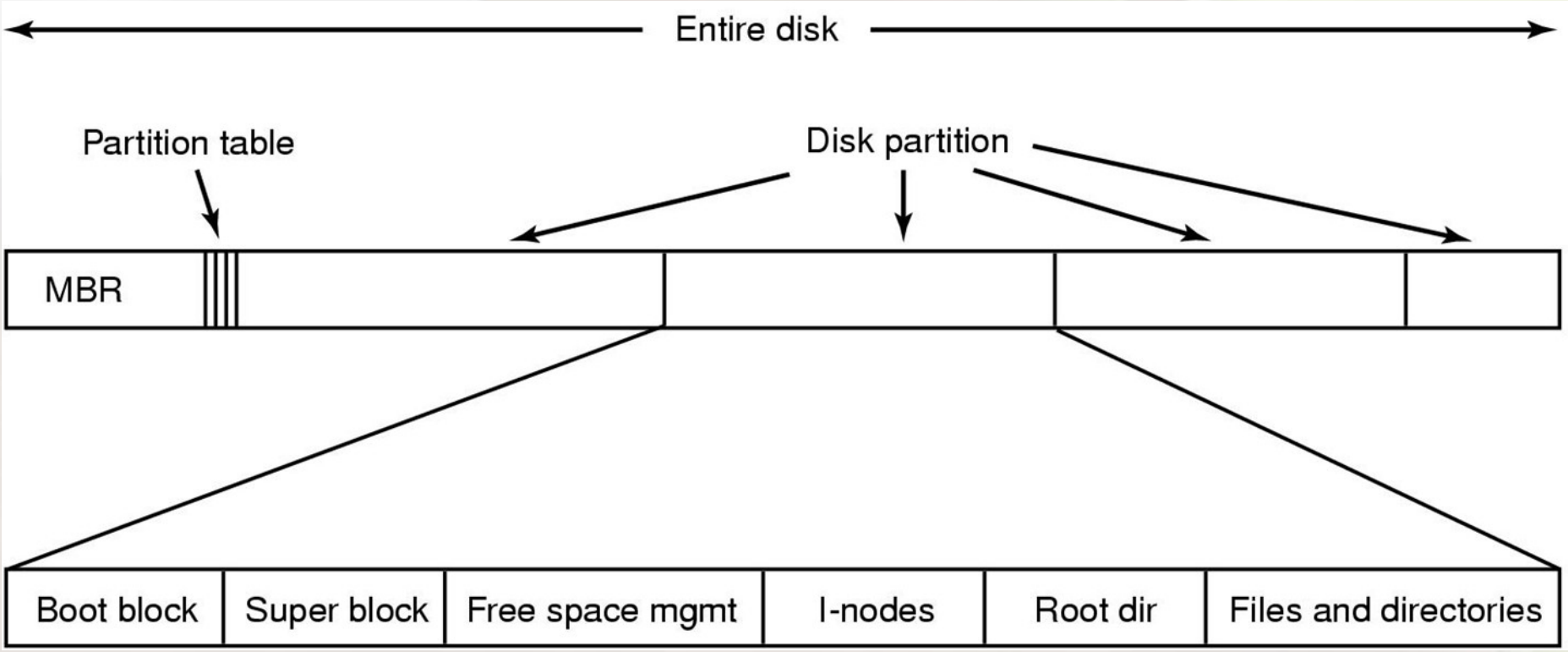
- Estrutura de diretórios Hierárquicos
  - Adotado pela maioria dos sistemas operacionais
  - Logicamente melhor organizado
  - É possível criar quantos diretórios quiser
  - Um diretório pode conter arquivos e outros diretórios (chamados subdiretórios)
  - Cada arquivo possui um **path** único que descreve todos os diretórios da raiz (MFD – Master File Directory) até o diretório onde o arquivo está ligado
  - Na maioria dos S.O.s os diretórios são tratados como arquivos tendo atributos e identificação

Diretórios (5)

- Estrutura de diretórios Hierárquicos



Esquema do Sistema de Arquivos (1)



Esquema do Sistema de Arquivos (2)

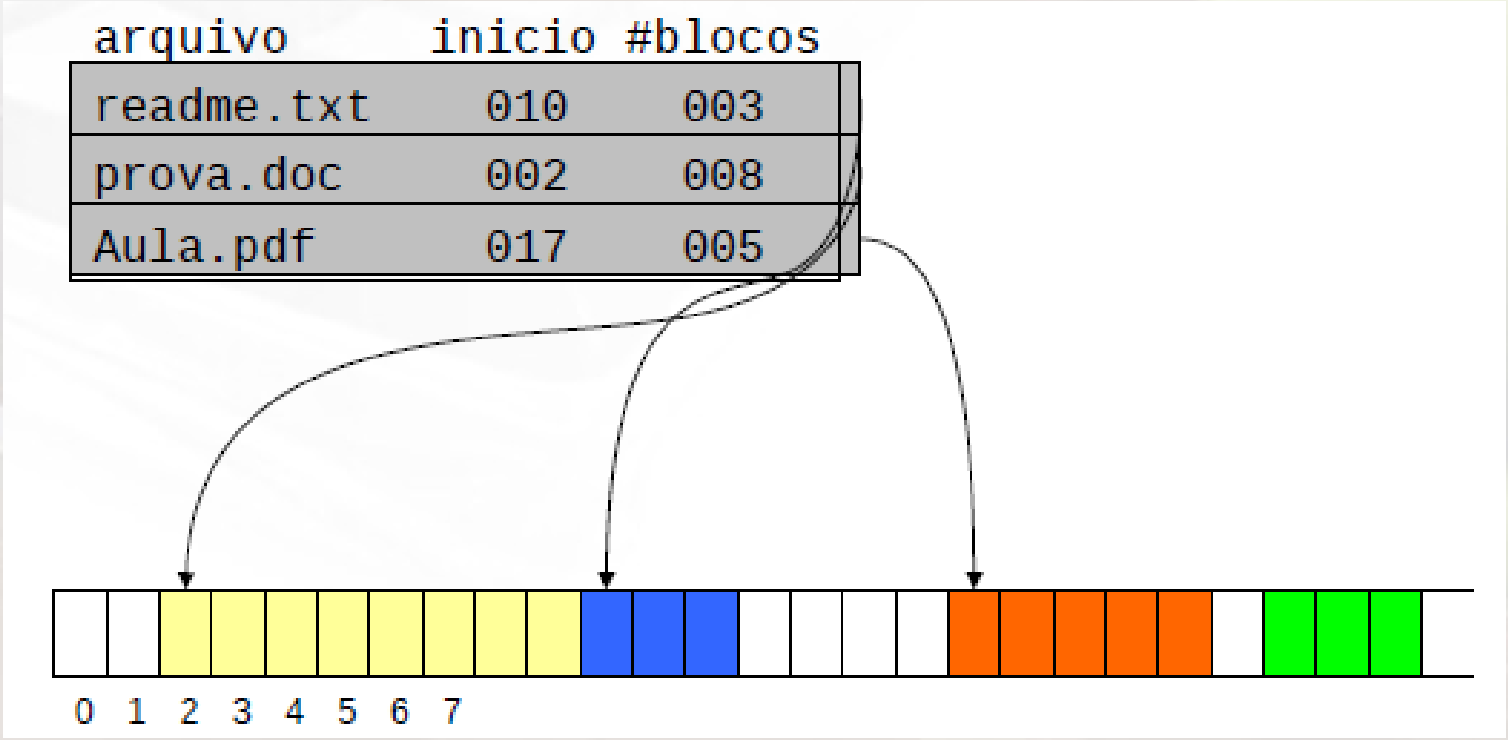
- A maioria dos discos é dividida em uma ou mais partições com Sistemas de arquivos independentes para cada partição
- O setor 0 do disco é chamado de *Master Boot Record* (MBR)
- Na inicialização do sistema, a BIOS lê e executa o MBR
  - O programa do MBR localiza a partição ativa, lê seu primeiro bloco, chamado de bloco de boot
  - O programa no bloco de boot carrega o S.O. contido na partição
- O esquema da partição varia de um S.O. para outro, mas é comum:
  - A definição de um SuperBloco: contém os principais parâmetros do sistema de arquivos (tipo, no. de blocos, etc.)
  - As informações sobre os blocos livres

Implementação de Arquivos (1)

- Alocação Contígua
  - Consiste em armazenar um arquivo em blocos sequencialmente dispostos
  - O sistema localiza um arquivo através do endereço do primeiro bloco e da sua extensão em blocos
  - O acesso é bastante simples
  - Seu principal problema é a alocação de novos arquivos nos espaços livres
    - Para armazenar um arquivo que ocupa n blocos, é necessário uma cadeia com n blocos dispostos sequencialmente no disco
  - Além disso, como determinar o espaço necessário a um arquivo que possa se estender depois da sua criação?
    - Pré-alocação (fragmentação interna)

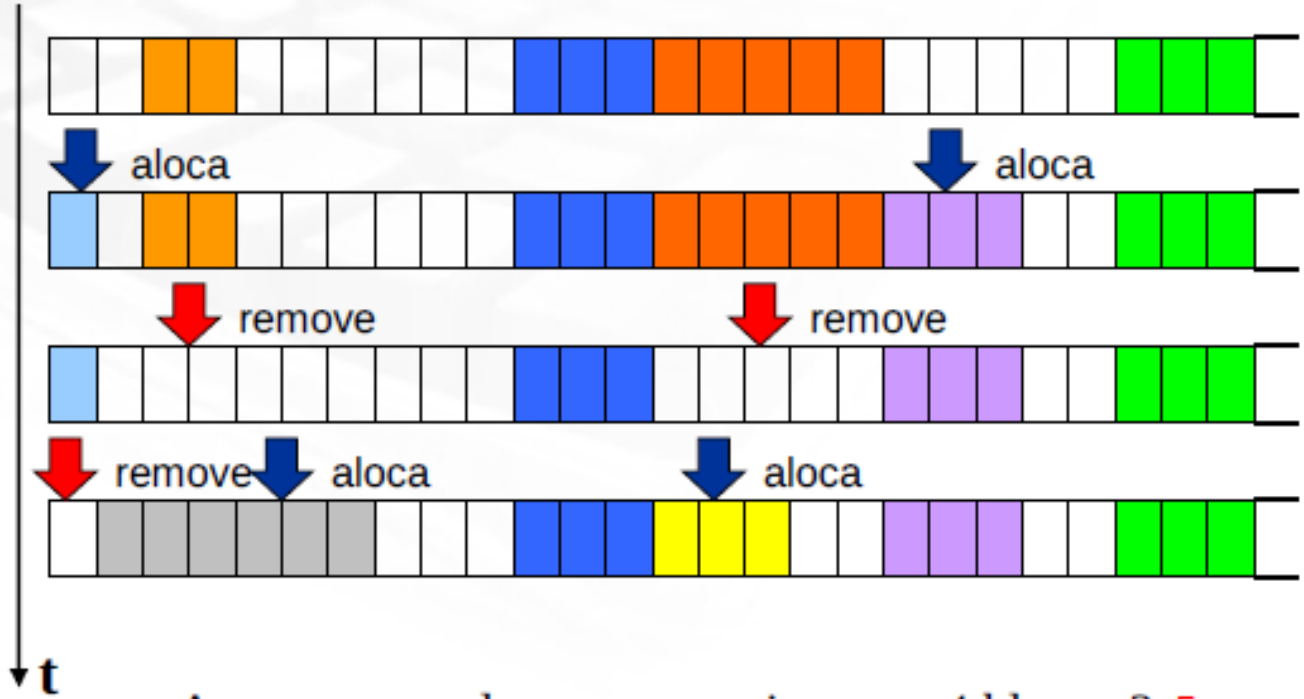
Implementação de Arquivos (2)

- Alocação Contígua (cont.)



Implementação de Arquivos (3)

- Alocação Contígua (cont.)



Agora, como alocar um arquivo com 4 blocos ? Fragmentação Externa !

Implementação de Arquivos (3)

- Alocação contínua:
  - Vantagens:
    - Simplicidade: somente o endereço do primeiro bloco e número de blocos no arquivo são necessários;
    - Desempenho para o acesso ao arquivo: acesso sequencial;
  - Desvantagens (discos rígidos):
    - Fragmentação externa:
      - Compactação → alto custo;
      - Reuso de espaço → atualização da lista de espaços livres;
        - Conhecimento prévio do tamanho do arquivo para alocar o espaço necessário;
  - CD-ROM e DVD-ROM (quando somente escrita);

Implementação de Arquivos (3)

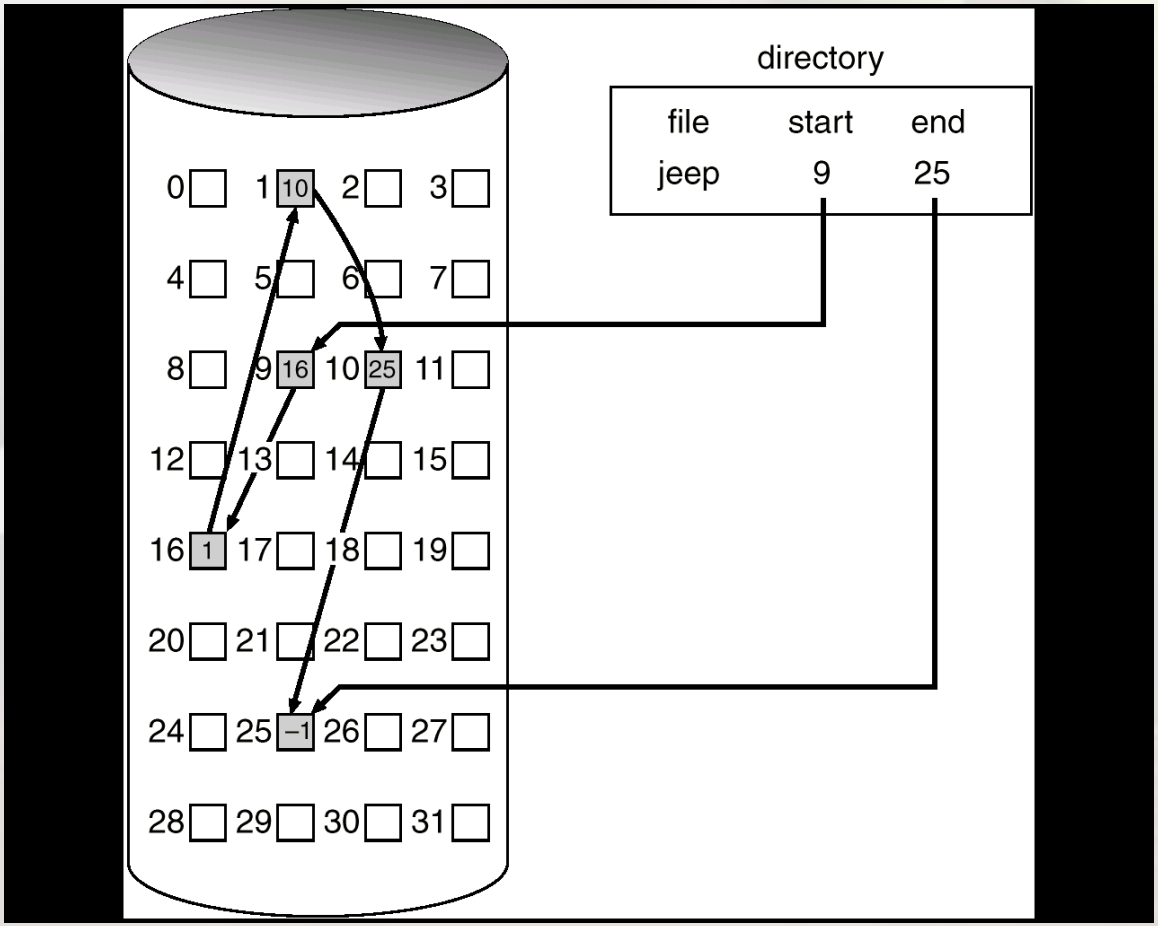
- Alocação com lista encadeada:
  - A primeira palavra de cada bloco é um ponteiro para o bloco seguinte;
  - O restante do bloco é destinado aos dados;
  - Apenas o endereço em disco do primeiro bloco do arquivo é armazenado;
    - Serviço de diretório é responsável por manter esse endereço;

Implementação de Arquivos (3)

- Alocação com lista encadeada:
  - Desvantagens:
    - Acesso aos arquivos é feito aleatoriamente
      - processo mais lento;
    - A informação armazenada em um bloco não é mais uma potência de dois, pois existe a necessidade de se armazenar o ponteiro para o próximo bloco;
  - Vantagem:
    - Não se perde espaço com a fragmentação externa;

Implementação de Arquivos (3)

- Alocação com lista encadeada



Implementação de Arquivos (3)

- Alocação com lista encadeada utilizando uma tabela na memória:
  - O ponteiro é colocado em uma tabela na memória ao invés de ser colocado no bloco;
    - FAT → Tabela de alocação de arquivos (File Allocation Table);
    - Assim, todo o bloco está disponível para alocação de dados;
  - Serviço de diretório é responsável por manter o início do arquivo (bloco inicial);
  - MS-DOS e família Windows 9x (exceto WinNT, Win2000 e WinXP - NTFS);

Implementação de Arquivos (3)

- Acesso aleatório se torna mais fácil devido ao uso da memória;
- Desvantagem:
  - Toda a tabela deve estar na memória;
  - Exemplo:
    - Com um disco de 20Gb com blocos de 1kb, a tabela precisa de 20 milhões de entradas, cada qual com 3 bytes (para permitir um acesso mais rápido, cada entrada pode ter 4 bytes) ocupando 60 (80) Mb da memória;

Implementação de Arquivos (3)

- Alocação com lista encadeada utilizando FAT

Arquivo A
4 7 2 10 12

Arquivo B
6 3 11 14

Implementação de Arquivos (3)

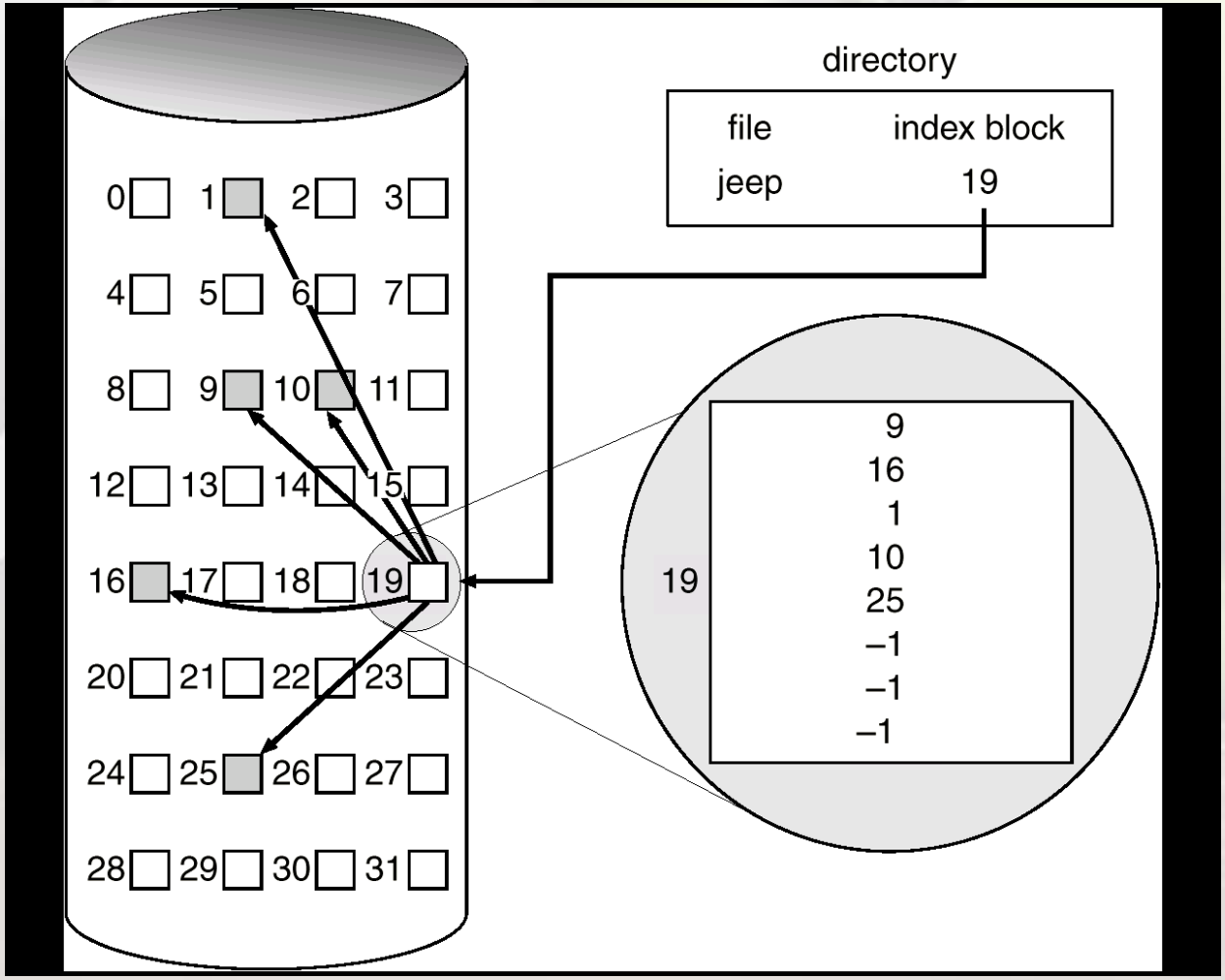
- *I-nodes*:
  - Cada arquivo possui uma estrutura de dados chamada i-node (index-node) que lista os atributos e endereços em disco dos blocos do arquivo;
    - Assim, dado o i-node de um arquivo é possível encontrar todos os blocos desse arquivo;
  - Se cada i-node ocupa n bytes e k arquivos podem estar aberto ao mesmo tempo → o total de memória ocupada é kn bytes;
  - UNIX e Linux;

Implementação de Arquivos (3)

- Espaço de memória ocupado pelos i-nodes é proporcional ao número de arquivos abertos; enquanto o espaço de memória ocupado pela tabela de arquivo (FAT) é proporcional ao tamanho do disco;
- Vantagem:
  - O i-node somente é carregado na memória quando o seu respectivo arquivo está aberto (em uso);
- Desvantagem:
  - O tamanho do arquivo pode aumentar muito
    - Solução: reservar o último endereço para outros endereços de blocos;

Implementação de Arquivos (3)

- *I-nodes*

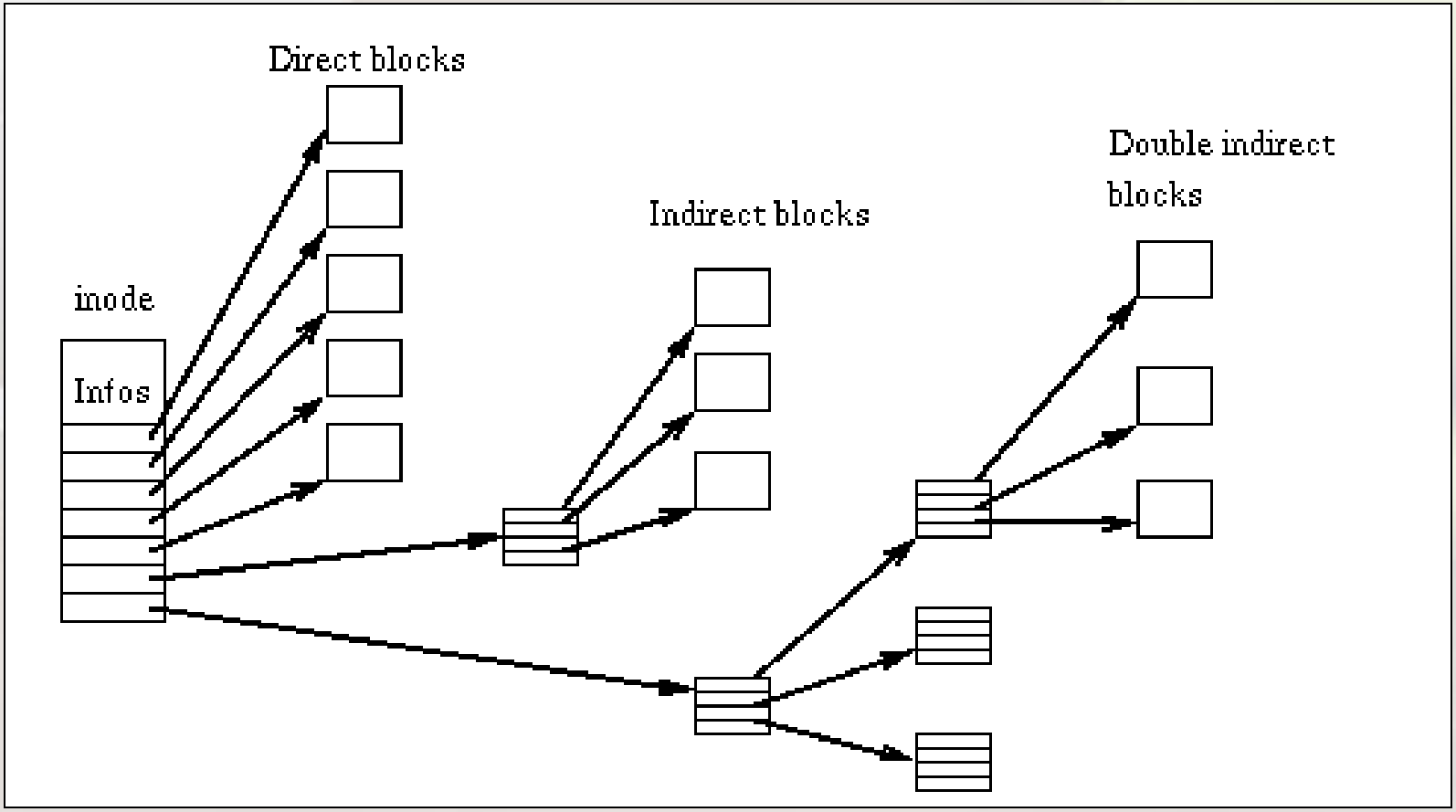


Implementação de Arquivos (3)

- *I-nodes*:

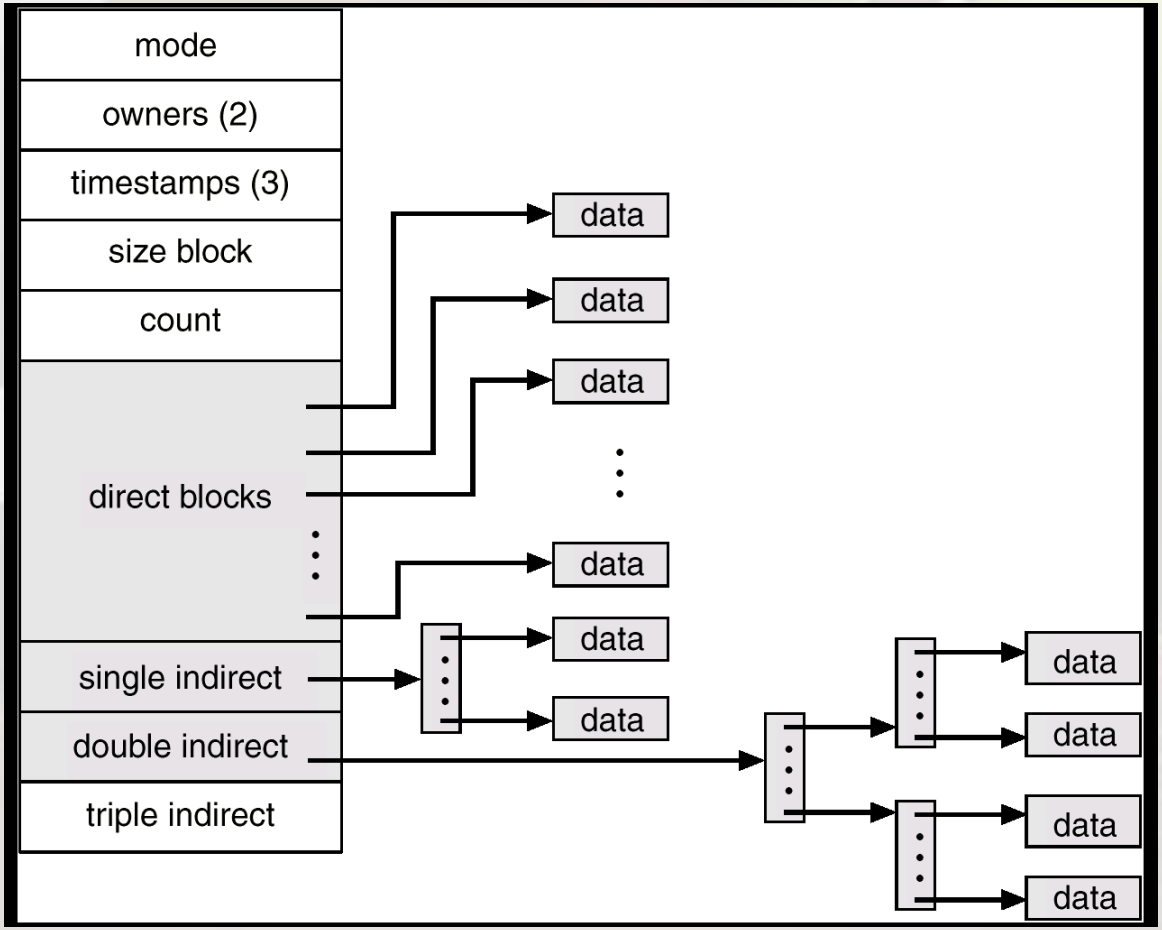
Implementação de Arquivos (3)

- *I-nodes*:



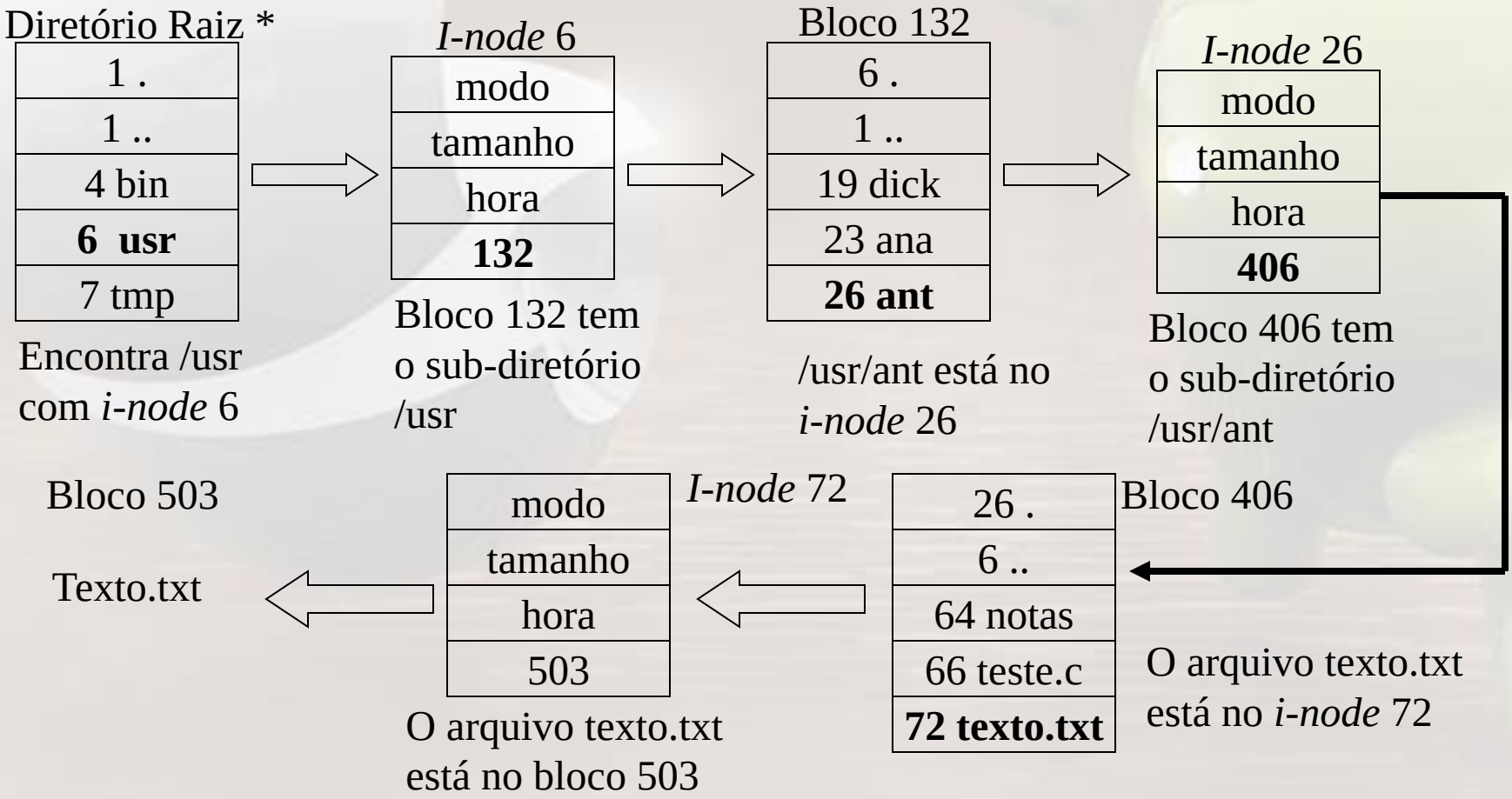
Implementação de Arquivos (3)

- *I-nodes*



Implementação de Arquivos (3)

- *I-nodes*



Exercícios

- Considerando as estruturas de i-nodes, quantos acessos a disco são necessários para encontrar o seguinte arquivo: /etc/inet/hosts? Considere que apenas o i-node do diretório-raiz está na memória RAM e que todos os arquivos ocupam apenas um bloco. Mostre toda a seqüência de acesso.

- Considerando que um i-node possua 10 endereços de blocos diretos e um endereço de bloco indireto, todos com 4 bytes, e se todos os blocos forem de 1024 bytes, qual é o tamanho máximo do arquivo? E se todos os blocos forem de 2048 bytes, qual é o tamanho máximo do arquivo?