Universidade Federal de Uberlândia
Faculdade de Computação
Sistemas Operacionais

Sistema de Arquivos

Implementação

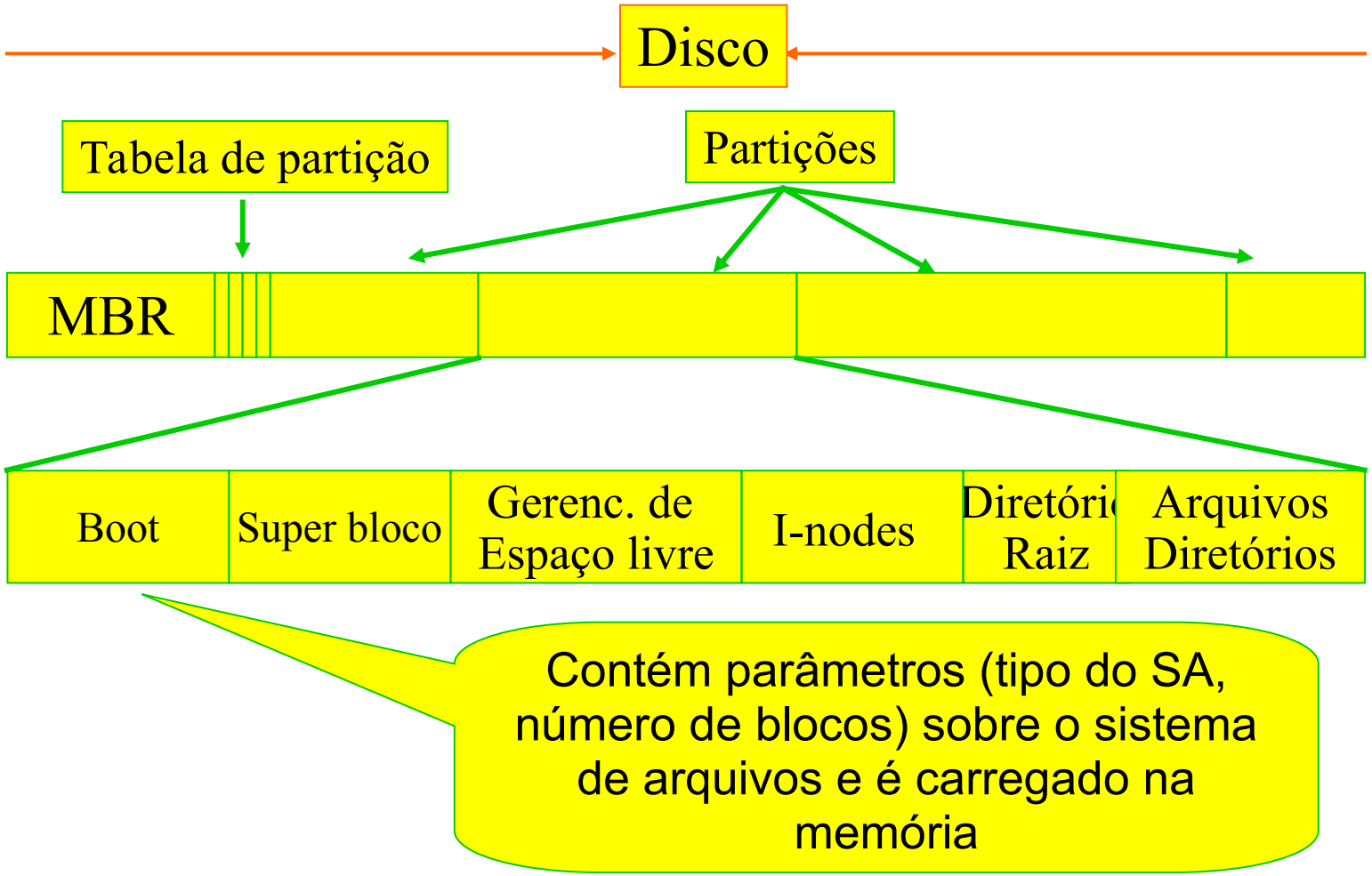
Prof. Dr. Marcelo Zanchetta do Nascimento

Roteiro

- Esquema de Sistema de Arquivos
- Implementação de Sistema de Arquivos
- Implementação de Diretório
- Diretório como grafos direcionados acíclicos
- Gerenciamento de Espaço em Disco
- Consistência de Arquivos
- Desempenho de Sistema de Arquivos
- Leituras Sugeridas

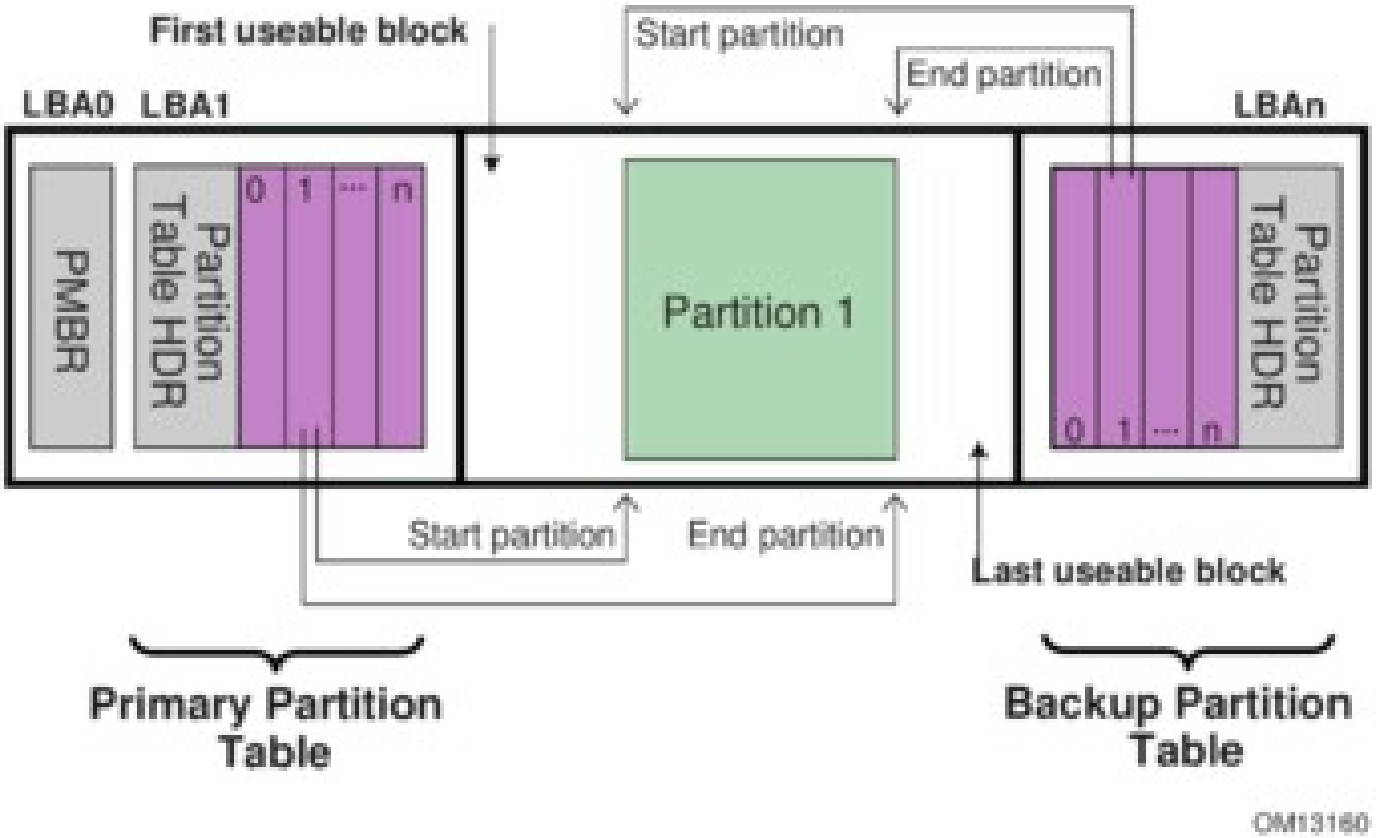
Esquema de Sistema de Arquivos

- Com o sistema *Master Boot Record (*MBR) – registro mestre de inicialização com 2^{32} setores, ou seja, 2,2 TB.



Esquema de Sistema de Arquivos

- Temos também a tabela de partições com Identificador Único Global (GUID);
- Suporta até 2^{64} setores, ou seja, 9,4 ZB (zettabytes).



Esquema de Sistema de Arquivos

- **Superbloco** → Contém todos os principais parâmetros sobre o sistema de arquivos, tal como, tamanho do bloco, nº de blocos, nº mágico, etc;
  - Lido do disco e salvo na memória quando o computador é inicializado ou quando o SA é utilizado pela primeira vez;
  - Nº mágico – indica o tipo de sistema de arquivos;
- **Gerenciamento de Espaço Livre** → implementado via mapas de bits, listas livres, etc.
- **I-nodes** → blocos de indexação com ID de cada arquivo;
- **Root** → diretório base da árvore de diretórios;
- **Arquivos e Diretórios** → blocos propriamente ditos;

Implementação do Sistema de Arquivos

- Controle de quais blocos de disco estão relacionados a quais arquivos;
  - Projeto depende:
    - Do dispositivo;
    - Do sistema de arquivos (FAT32, NTFS, EXT2, EXT3, EXT4, etc);

Alocação Contígua

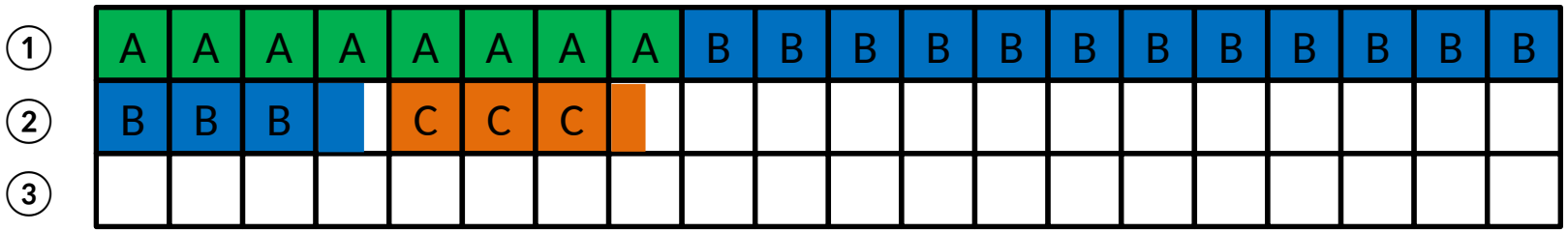
- Esquema mais simples;
- Arquivo é armazenado em blocos contíguos no disco;
- Arquivos são sempre armazenados iniciando no início do bloco;
- Caso o tamanho do arquivo não coincida com um múltiplo do tamanho de um bloco, espaço em disco será desperdiçado;

Alocação Contígua

- Vantagens:
  - Simples de implementar – dois números, bloco inicial e final;
  - Desempenho de leitura é excelente, pois todo o arquivo encontra-se armazenado sequencialmente no disco.
- Desvantagens:
  - Com o tempo, o disco fica fragmentado;
  - Hoje em dia é adequado apenas para sistemas de arquivos nos quais sabe a priori o tamanho dos arquivos (ex: DVD).

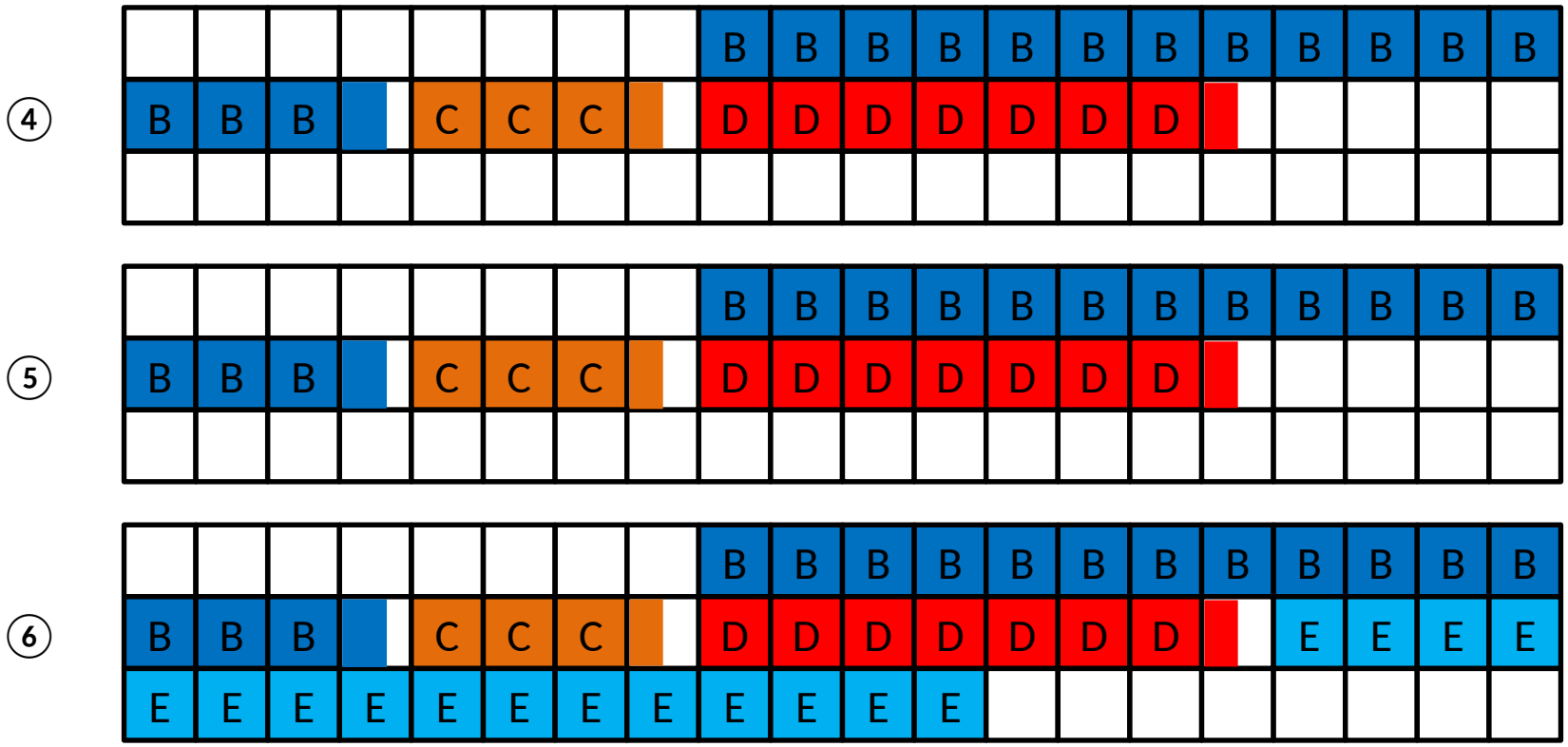
Alocação Contígua

- **(INS) Arquivo A = 32KB;**
- **(INS) Arquivo B = 63KB;**
- **(INS) Arquivo C = 14KB;**
- (DEL) Arquivo A;
- **(INS) Arquivo D = 30KB;**
- **(INS) Arquivo E = 64KB;**
- (DEL) Arquivo C;
- **(INS) Arquivo F = 32KB;**
- (DEL) Arquivo D;
- **(INS) Arquivo G = 64KB;**



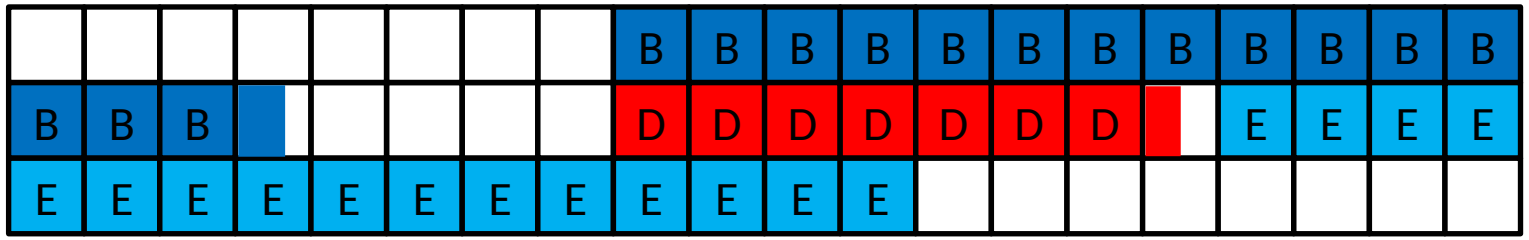
→ tamanho do bloco = 4KB

Alocação Contígua

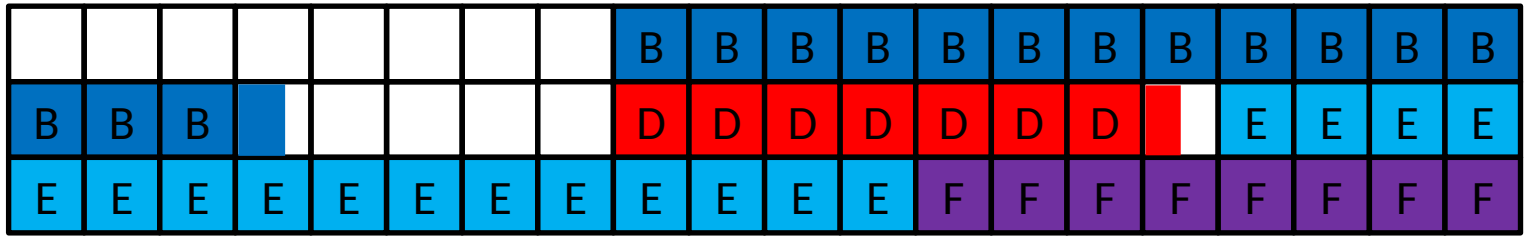


Alocação Contígua

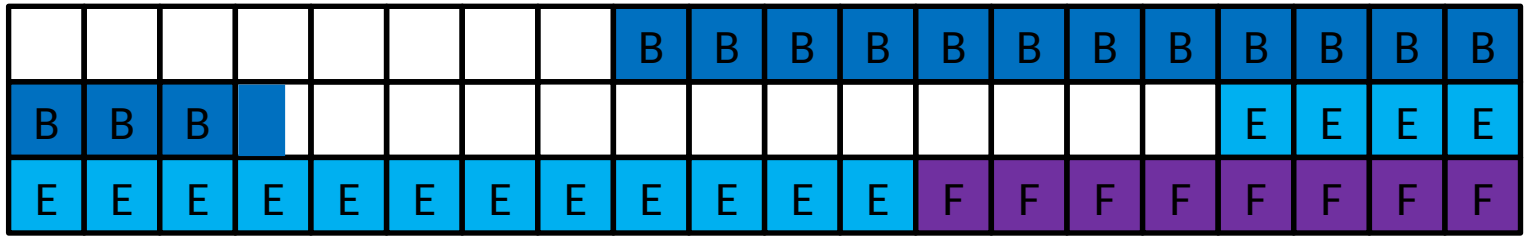
⑦



⑧



⑨

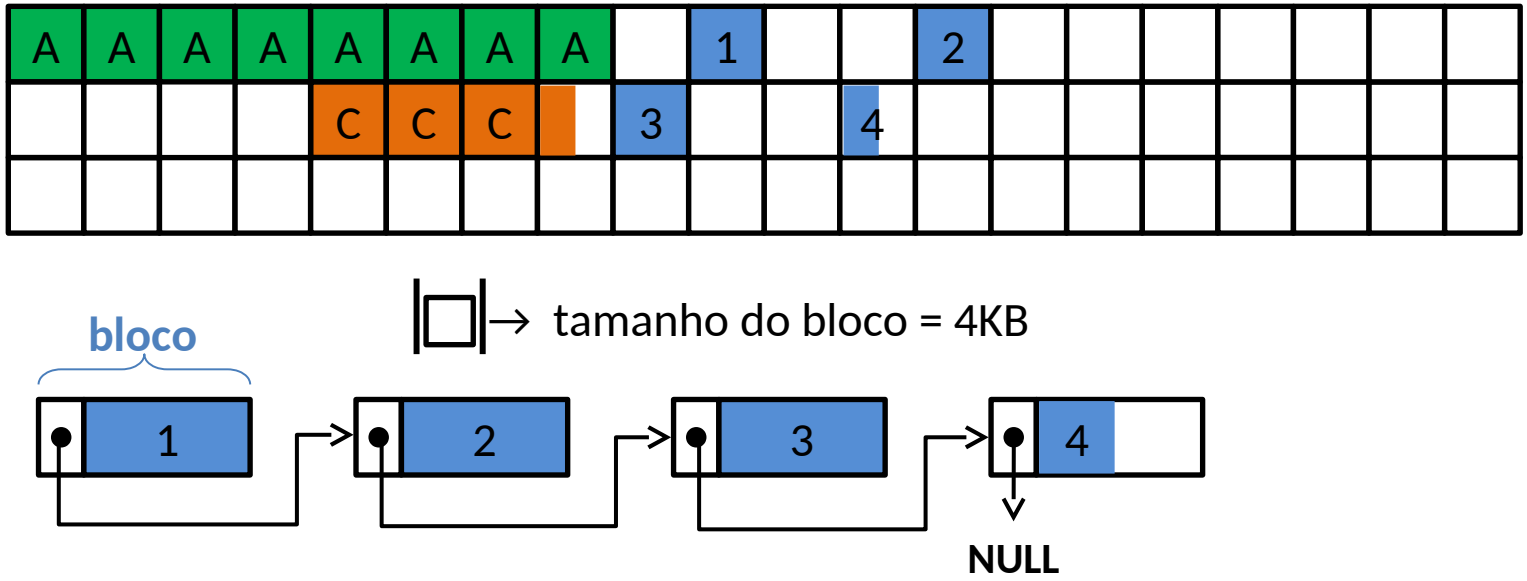


⑩ **(INS) Arquivo G = 64KB → Embora haja espaço no disco é impossível acomodar o arquivo pois os blocos não são contíguos;**

Alocação por Lista Encadeada

- Lista encadeada de blocos em disco;
- A primeira palavra de cada bloco é um ponteiro para o próximo bloco usado pelo arquivo;
- Qualquer bloco pode ser usado, consequentemente nenhum espaço em disco é desperdiçado pela fragmentação;
- Indexar é mais simples – basta registrar qual o primeiro bloco do arquivo;
- Acesso aleatório é extremamente lento;
- Para acessar o bloco **n** (note, apenas o bloco n), todos os n-1 blocos devem ser lidos;

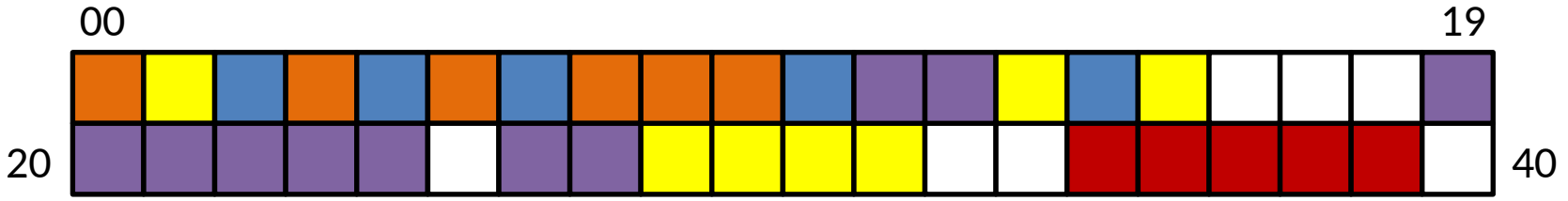
Alocação por Lista Encadeada



Alocação Usando uma Tabela na Memória

- Busca eliminar as desvantagens da alocação por lista encadeada;
- Coloca as palavras do ponteiro de cada bloco em uma tabela na memória;
- Esta tabela na memória principal é chamada de **FAT – File Allocation Table**;
- Mas rápido acessar dados diretamente da memória;
- Tabela inteira deve existir na memória;

Alocação Usando uma Tabela na Memória



0	3
1	13
2	6
3	5
4	-1
5	7
6	10
7	8
8	9
9	-1
10	14
11	12
12	19
13	15
14	4
15	-1

16	
17	
18	
19	20
20	21
21	22
22	23
23	24
24	26
25	
26	27
27	-1
28	29
29	30
30	31
31	1

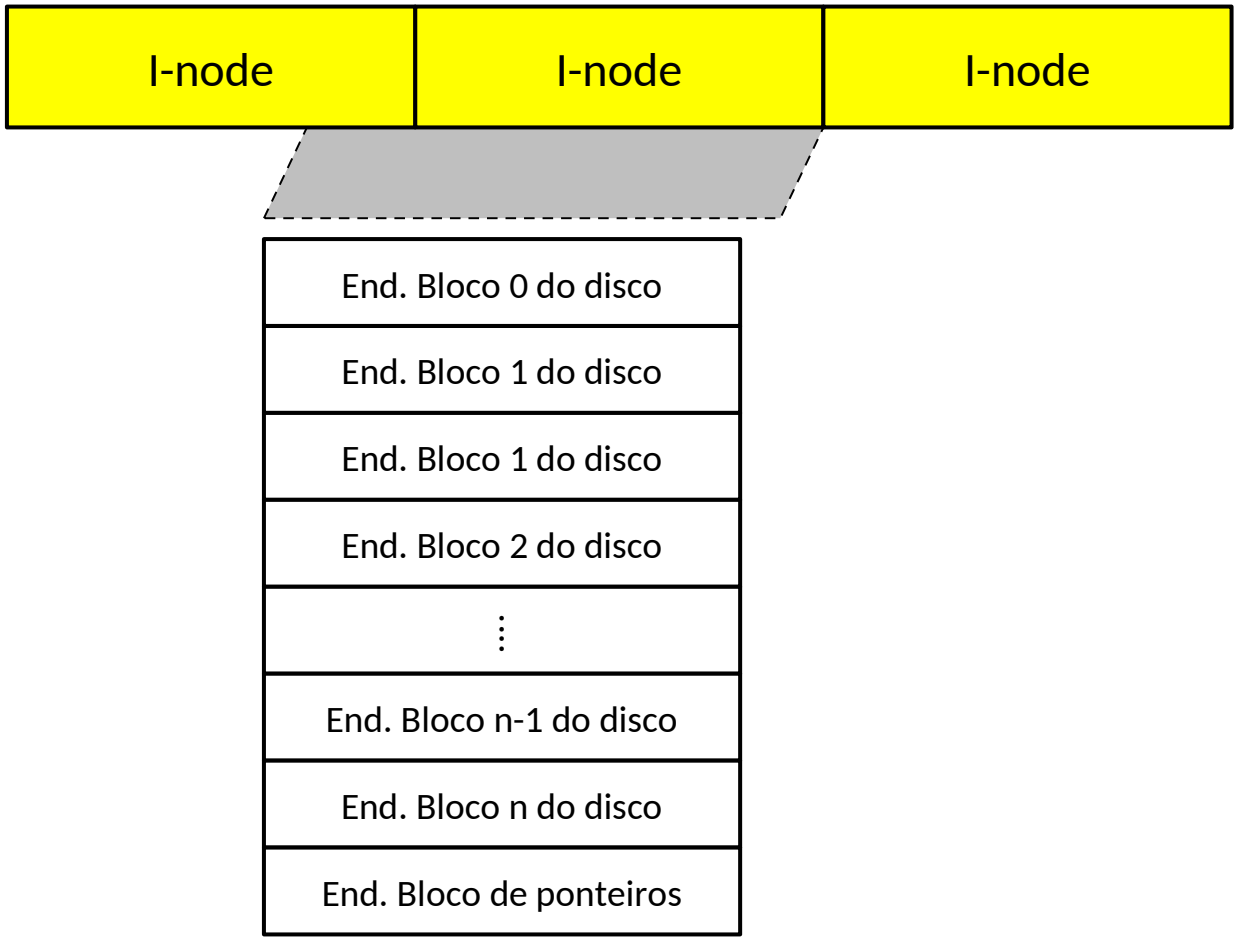
32	
33	
34	
35	36
36	37
37	38
38	39
39	-1
40	

I-nodes

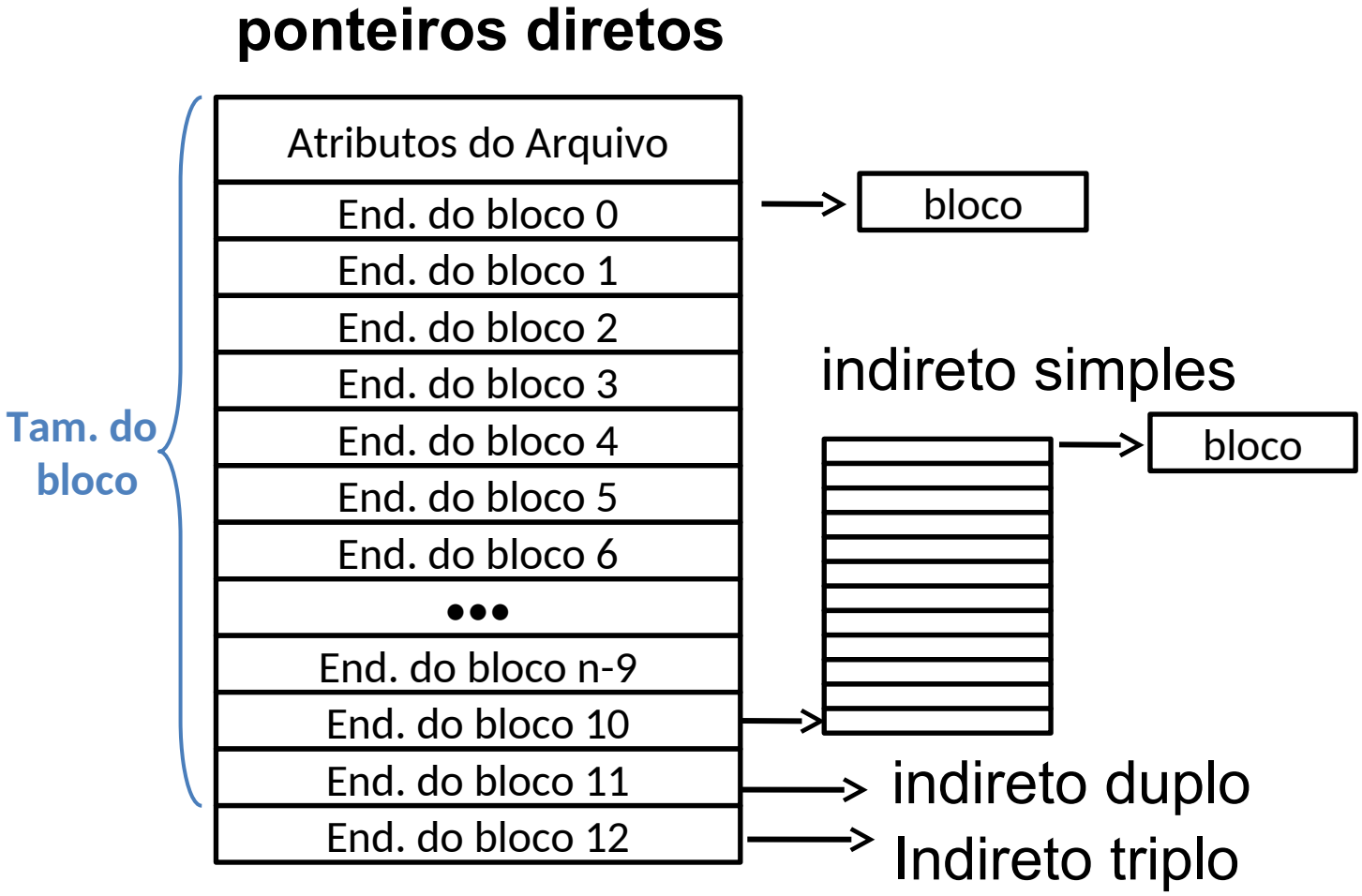
- Um tipo de estrutura de dados;
- Utilizada na maioria dos sistemas de arquivos modernos;
- É necessário apenas um ponteiro para o primeiro i-node do arquivo para indexá-lo;
- Os requisitos de memória são muito menores se comparados com a alocação por lista encadeada usando uma tabela na memória (FAT);
- A tabela precisa ter no máximo N x K bits:
  - N é o tamanho do i-node;
  - K é o número máximo de arquivos que podem estar abertos no sistema em um dado momento.

I-nodes

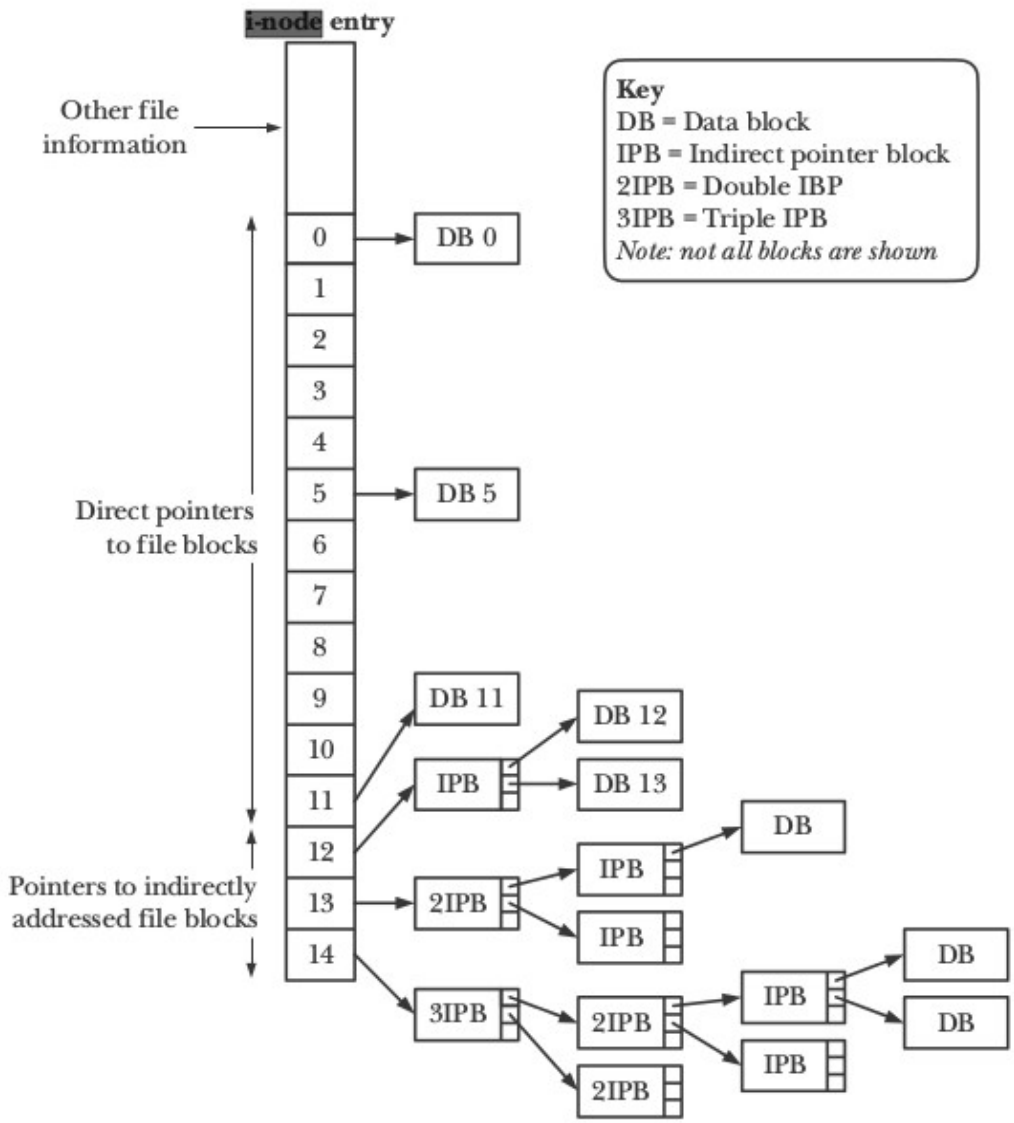
Em Unix, você pode obter o número dos i-nodes para um arquivo com o comando: $ ls -i;



I-nodes


ponteiros diretos
Atributos do Arquivo
End. do bloco 0
End. do bloco 1
End. do bloco 2
End. do bloco 3
End. do bloco 4
End. do bloco 5
End. do bloco 6
•••
End. do bloco n-9
End. do bloco 10
End. do bloco 11
End. do bloco 12
Tam. do bloco
bloco
indireto simples
bloco
indireto duplo
Indireto triplo

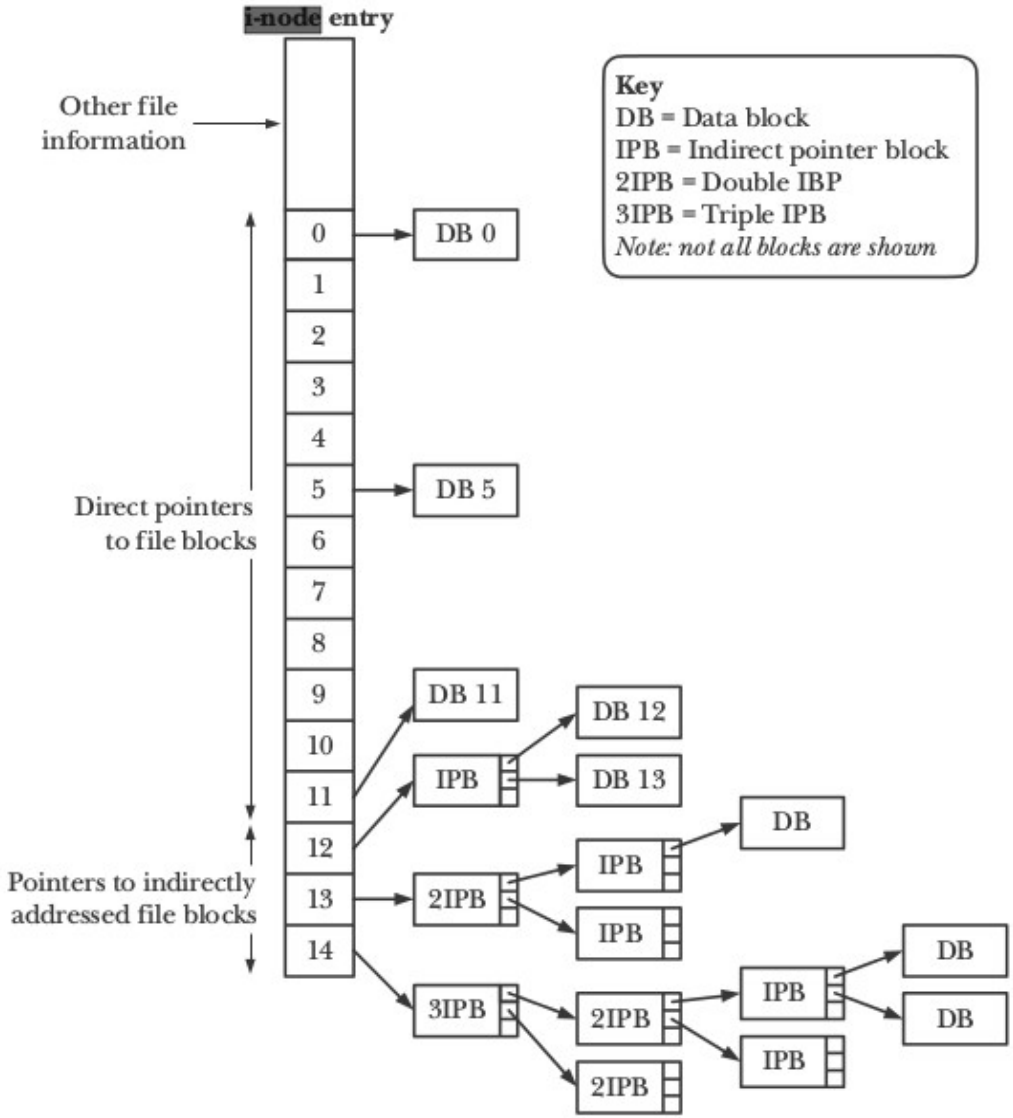
I-nodes - Linux



- Os primeiros 12 ponteiros apontam para a localização no sistema de arquivo dos 12 blocos do arquivo.
- O próximo é para um bloco de ponteiros que dá a localização 13 e subsequentes blocos do arquivo.

Figura 1 - Estrutura de blocos para um sistema de arquivo ext2

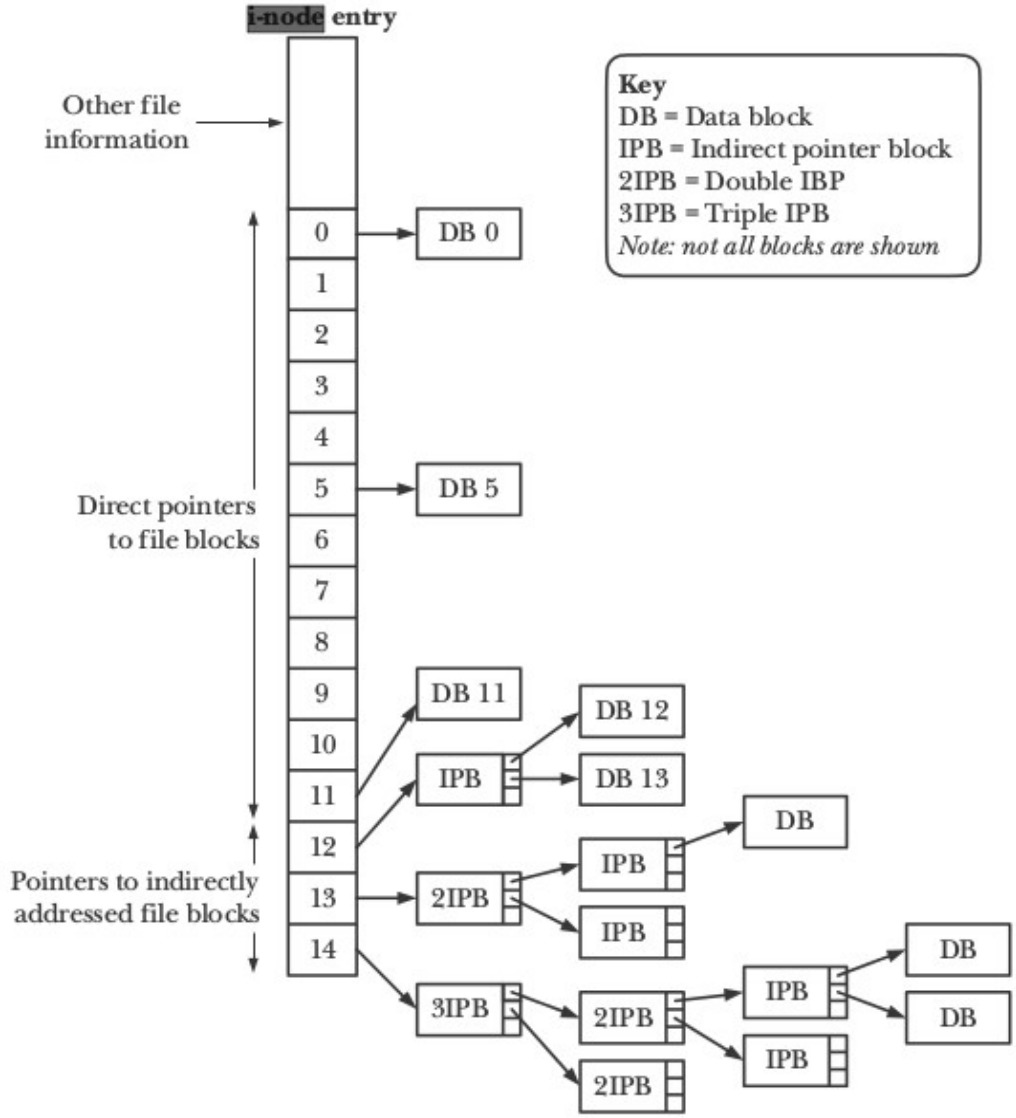
I-nodes - Linux



- O número de ponteiros depende do tamanho do bloco do sistema de arquivo;
- Cada ponteiro requer 4 bytes;
- Exemplo:
  - 256 ponteiros (para um bloco de tamanho 1024 bytes);
  - 1024 ponteiros (para um bloco de tamanho 4096 bytes).

Figura 1 - Estrutura de blocos para um sistema de arquivo ext2

I-nodes - Linux



- Para arquivos maiores, o decimo terceiro ponteiro é um “double indirect pointer” – o qual aponta para blocos de ponteiros;
- Também há o triple indirect pointer.

Figura 1 - Estrutura de blocos para um sistema de arquivo ext2

Exemplo de I-node

- Se o sistema i-node tem blocos de tamanho 1 KB com **10 ponteiros diretos** para blocos, um bloco para indireto simples, um bloco para indireto duplo e um bloco para indireto tripo;

- Cada ponteiro para o bloco tem 32 bits (4 Bytes);

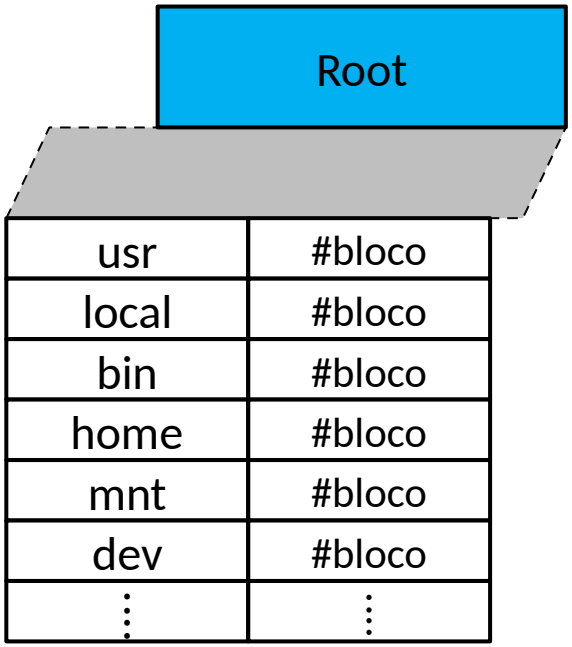
- Como deve ser o procedimento para carregar em memória o conteúdo do arquivo denominado **aula.txt** de tamanho de 270 kB?

Exemplo de I-node

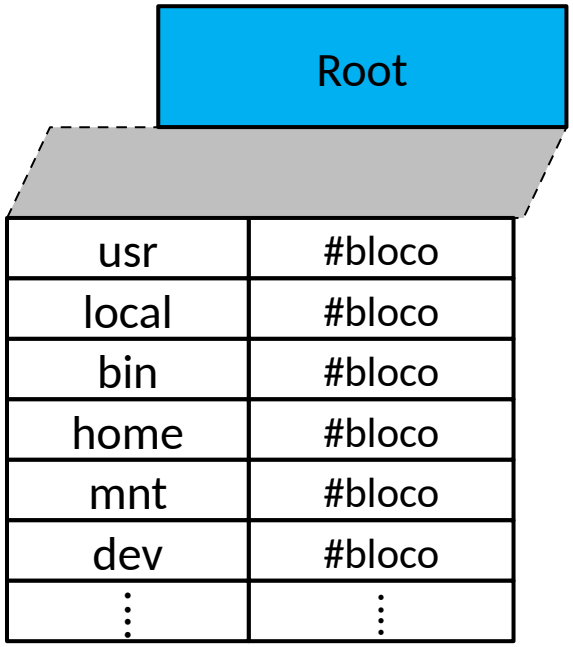
- Quantos blocos são empregados?
  - Primeiro deve ser determinado o número de blocos:
    - Bloco de 1KB e ponteiro de 4B → 1KB/4B = 256 blocos;
- Ponteiro direto → cada bloco tem 1KB
  - Armazena 10 blocos diretos (10 KB);
- Ponteiro indireto simples (bloco número 10) → 256 blocos;
  - 10KB + 256 KB = 266KB;
- Ponteiro indireto duplo → 4KB → 266KB + 4KB = 270KB.
- Se o arquivo tivesse o tamanho de 128KB?

Implementação de Diretórios

- **Arquivo especial** do sistema de arquivos;
- Armazenado em um local conhecido do disco **(partição);**
- Há sempre um diretório especial: **raiz**
  - "/" (root) é um diretório especial;
  - Sempre aberto;
  - Alocação direta no início do disco;
- Todos os **demais diretórios** são tratados como um tipo especial de arquivo;
- Diretórios possuem um **nome** e **atributos** associados.



Abertura de Diretórios



ls /home

① O programa ls é executado pelo shell;

② O programa executa a chamada do sistema OPENDIR("/home");

③ Sistema de arquivos resolve o caminho e encontra o diretório "home" na raiz;

④ O nº do 1º bloco associado a "home" é recuperado da raiz e carregado como um i-node;

⑤ O 1º bloco de dados indexado pelo i-node é carregado na memória;

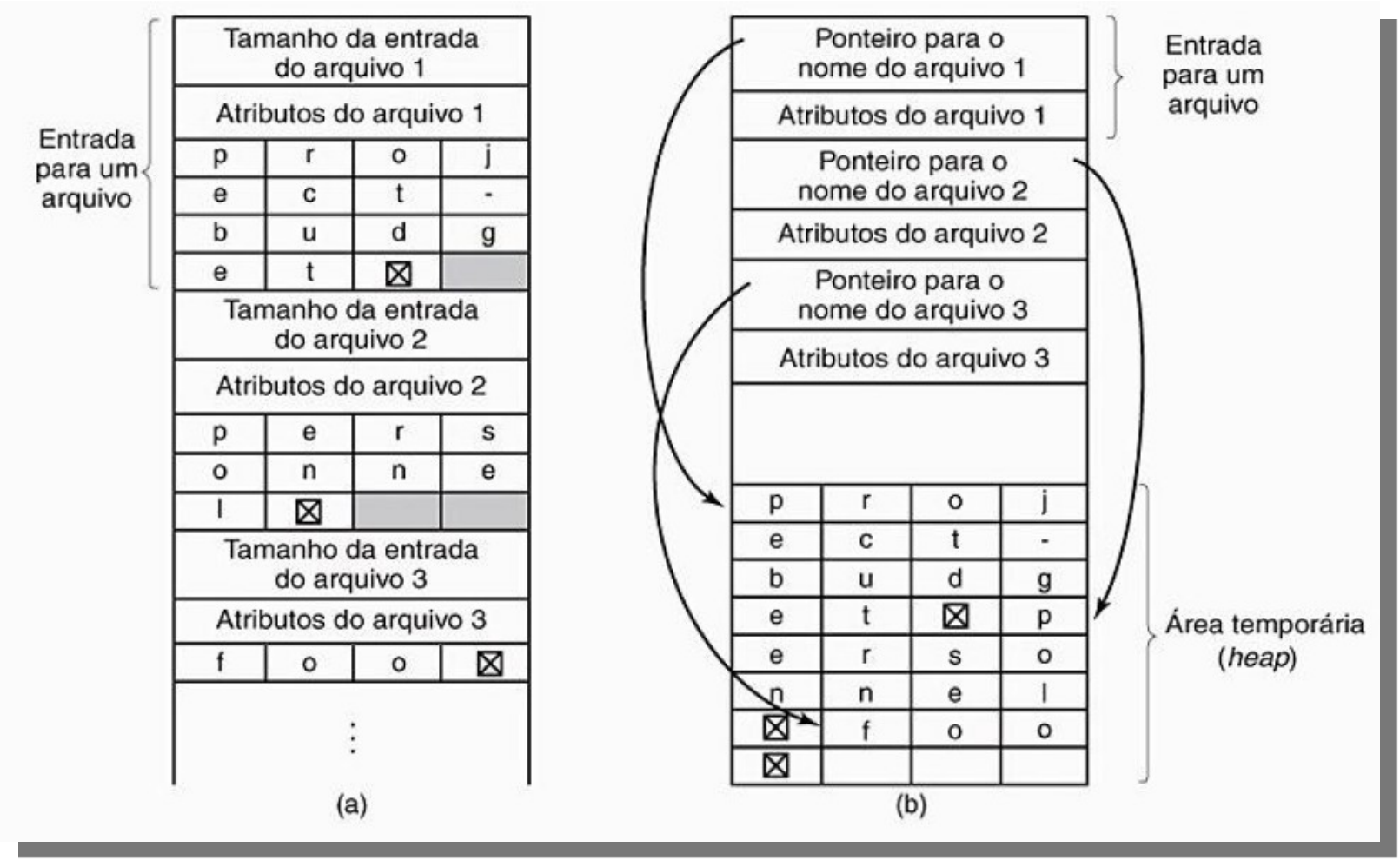
⑥ O programa ls lê então o conteúdo do arquivo utilizando a chamada do sistema READDIR;

⑦ Eventualmente o i-node é consultado para buscar mais blocos de dados do disco;

⑧ Ao final da leitura do diretório a chamada do sistema CLOSEDIR é invocada fechando o diretório;

⑨ O i-node associado ao diretório home é liberado;

Formas de Codificação das Entradas em um Diretório

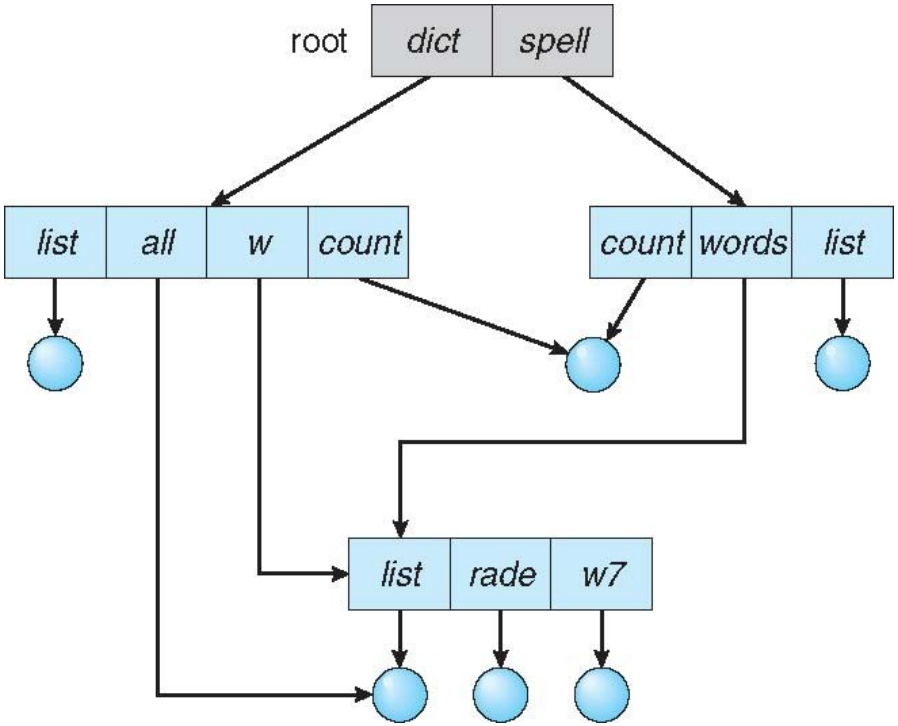


A – sequencialmente – B – Área temporária

Tanenbaum, A. S. Sistemas Operacionais Modernos, 3ª Edição. Pearson.

Diretório como grafos direcionados acíclicos

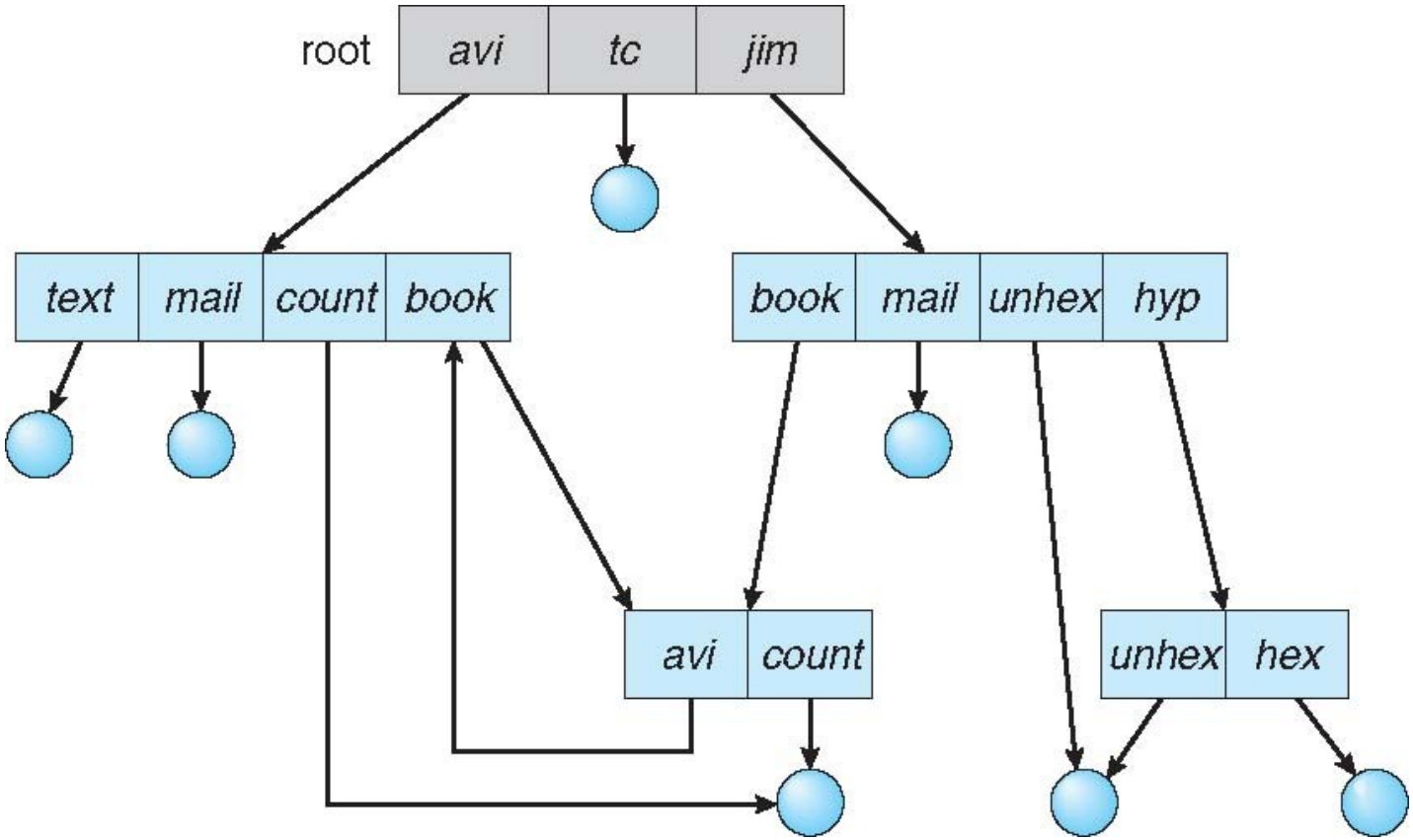
- **Generalização da estrutura** em árvore para permitir o **compartilhamento de arquivos** entre diretórios;
- Fornece compartilhamento através de **caminhos alternativos** para um arquivo em diferentes diretórios;
- Cria uma ligação (link).



Estrutura de Grafo Acíclico

Diretório como grafos direcionados acíclicos

- O ***link*** é uma forma comum de alias;
  - **Ponteiro** para outro arquivo ou subdiretório.



Diretório como grafos direcionados acíclicos

- *Link* é uma entrada na estrutura que pode ser;
  - *Soft link* (simbólico): fornece o caminho do arquivo com um atalho.
  - *Hard link*: fornece a localização (bloco) do arquivo no disco para o I-node do arquivo;
- Vantagem: compartilhamento de arquivos ou diretório;
- Desvantagens: Um arquivo pode possuir mais de um caminho de acesso (problemas para contabilização de acessos, *backups*, etc) e criação de laços através de links (algoritmo para verificar a não criação de um laço (desempenho)).

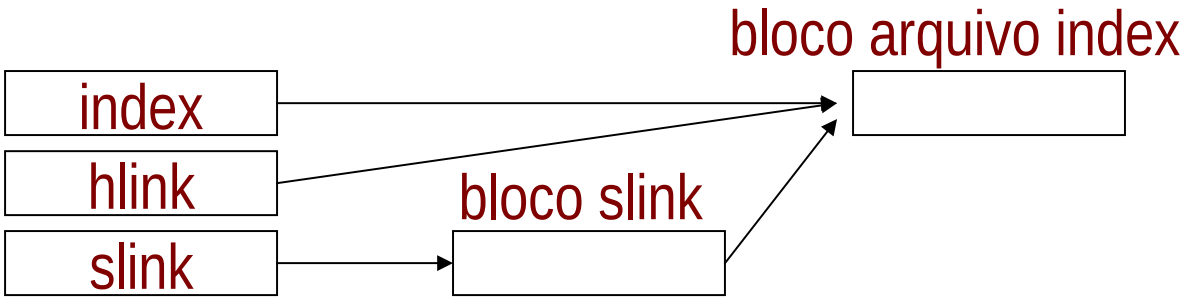
Diretório como grafos direcionados acíclicos

Exemplos UNIX

```
$ ln index  (hlink)
$ ln -s index  (slink)
$ ls -l
$ ls -lia
```

Contador de referências

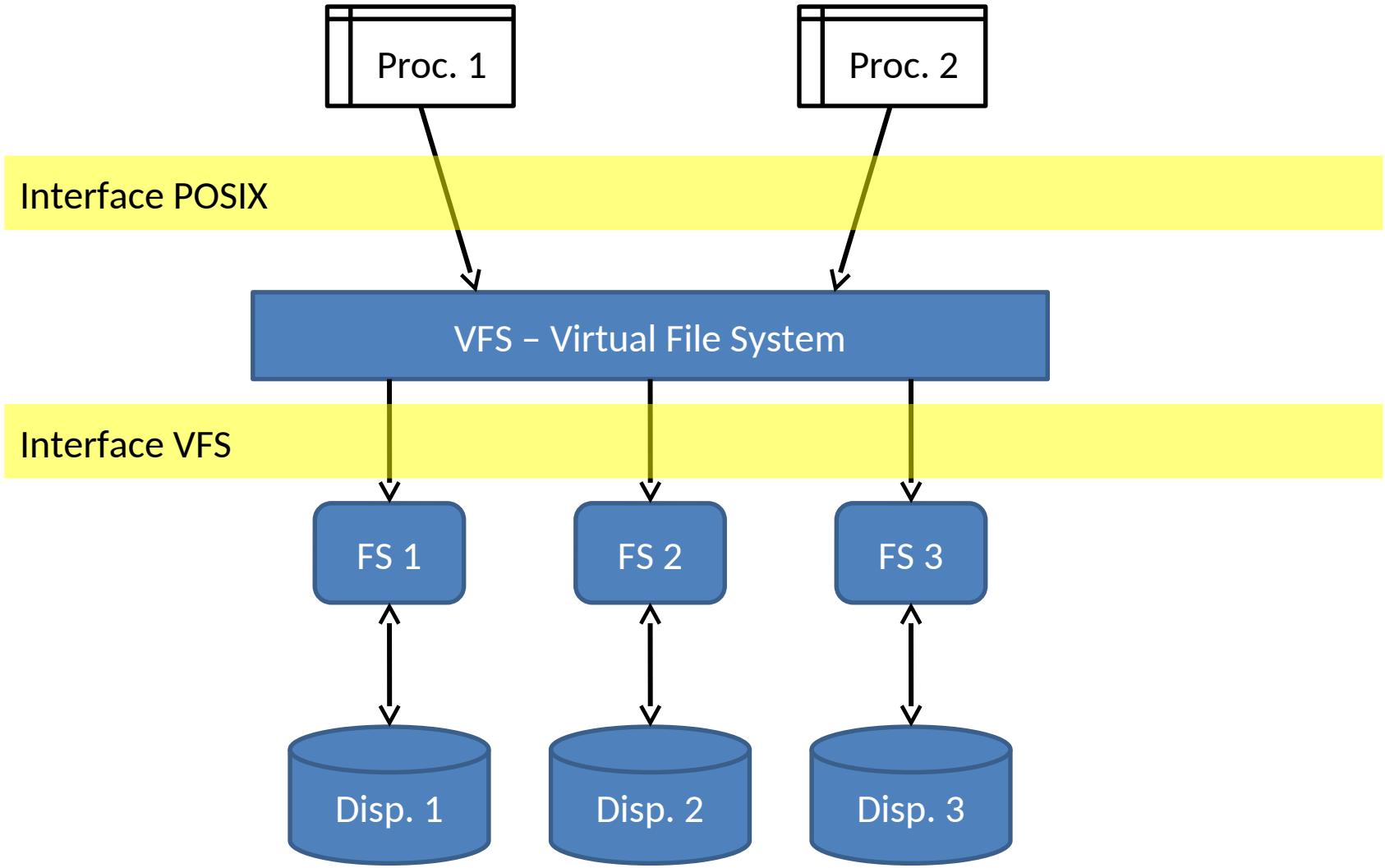
```
-rw- - - - - - -   2      chavez chem   5228   Mar 12 11:36  index
-rw- - - - - - -   2      chavez chem   5228   Mar 12 11:36  hlink
lrwx rwx rwx       1      chavez chem   5      Mar 12 11:36  slink→index
```



Sistemas de Arquivos Virtuais

- Diferentes sistemas de arquivos em uso no mesmo sistema operacional;
- Exemplo: Windows
  - Principal – **NTFS**
  - CD-ROM – **ISO 9660**
  - HD-Antigo – **FAT-16**
  - DVD – **UDF**
- Cada **sistema de arquivos** é identificado por uma unidade (C:, D:, …);
- Usando a unidade, o SO sabe para qual sistema de arquivos repassar a requisição;
- O SO não tenta unificar sistemas de arquivos heterogêneos.

Organização do VFS



Gerenciamento do Espaço em Disco

- Os dispositivos de armazenamento, tal como todos os outros recursos em um sistema computacional, são recursos **finitos**;

- O gerenciamento eficiente é uma das preocupações fundamentais dos sistemas operacionais;

- **Estratégias:**

  - Armazenamento de arquivos;
  - Tamanho dos blocos;
  - Gerenciamento de espaço;
  - Cotas de usuários.

Gerenciamento do Espaço em Disco

- O S.O. pode utilizar uma das duas estratégias possíveis para armazenar um arquivo de **n** bytes:
  - São alocados ao arquivo **n** bytes consecutivos do espaço disponível em disco;
  - Arquivo é espalhado por um número de blocos não necessariamente contíguos: blocos com tamanho fixo;
  - A maioria dos sistemas de arquivos utilizam essa estratégia;
- Algo próximo ao que ocorre na memória.

Gerenciamento do Espaço em Disco

Qual é o tamanho ideal para um bloco?

- Tamanho do bloco:
  - Se for grande pode ocorrer desperdício de espaço;
  - Se for pequeno, um arquivo irá ocupar muitos blocos, tornando o acesso/busca lento.

O tamanho do bloco tem influência na eficiência de utilização e de acesso ao disco (desempenho).

- Exemplos:
  - UNIX: 1KB;
  - MS-DOS: 512 Bytes;
  - Tamanho do bloco depende também do tamanho do disco;

Gerenciamento do Espaço em Disco

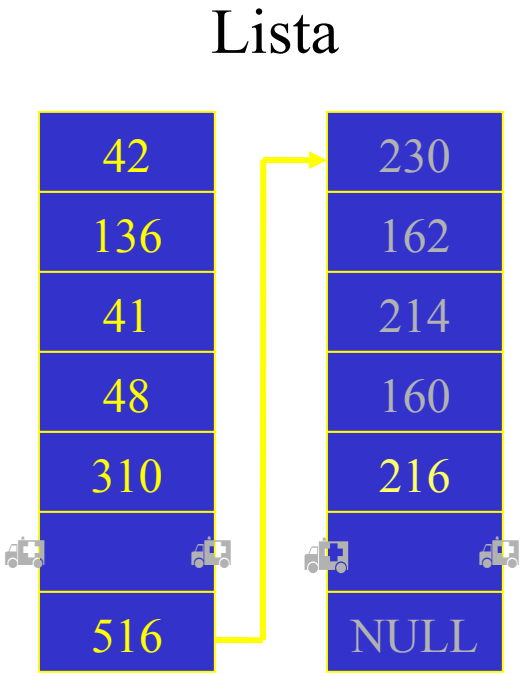
Monitoramento dos blocos livres

- Contém a localização dos blocos livres;
- Lista encadeada de blocos;
- Última entrada da lista armazena um ponteiro nulo para indicar que não há mais lista de blocos livres;
- Vantagens:
  - Requer menos espaço se existem poucos blocos livres (disco quase cheio);
  - Armazena apenas um bloco de ponteiros na memória;
- Desvantagem:
  - Precisa de mais espaço se houver um grande conjunto de blocos livres (disco quase vazio);

Gerenciamento do Espaço em Disco

Monitoramento dos blocos livres

- Exemplo:



Gerenciamento do Espaço em Disco

- **Mapa de bits:**
  - Contém um bit para cada bloco do sistema de arquivos;
  - Cada bloco pode armazenar número de 32 bits;
  - **Vantagem:**
    - o sistema de arquivo pode determinar rapidamente se há blocos contíguos disponíveis em certa localização;
  - **Desvantagem:**
    - Terá que pesquisar o mapa de bits inteiro para encontrar um bloco livre, resultando em sobrecarga de execução;

Gerenciamento do Espaço em Disco

- **Mapa de bits:**

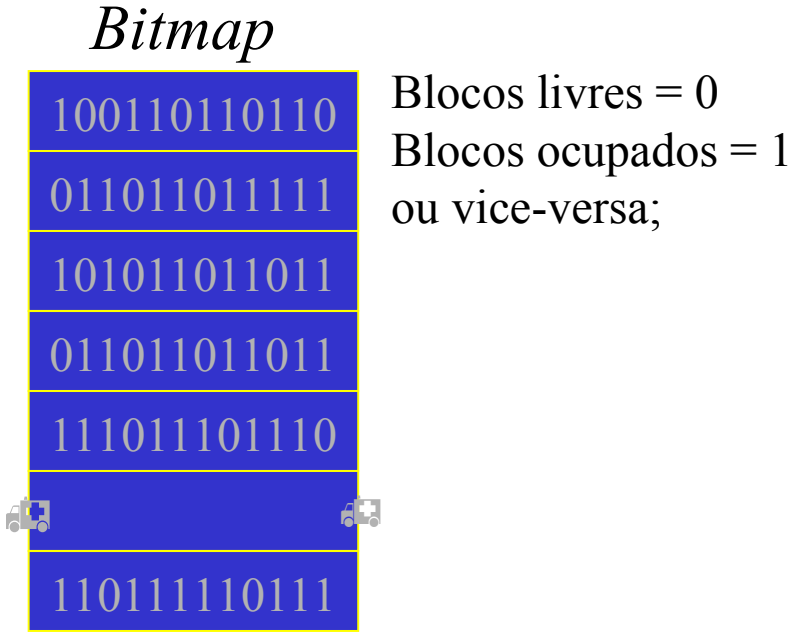
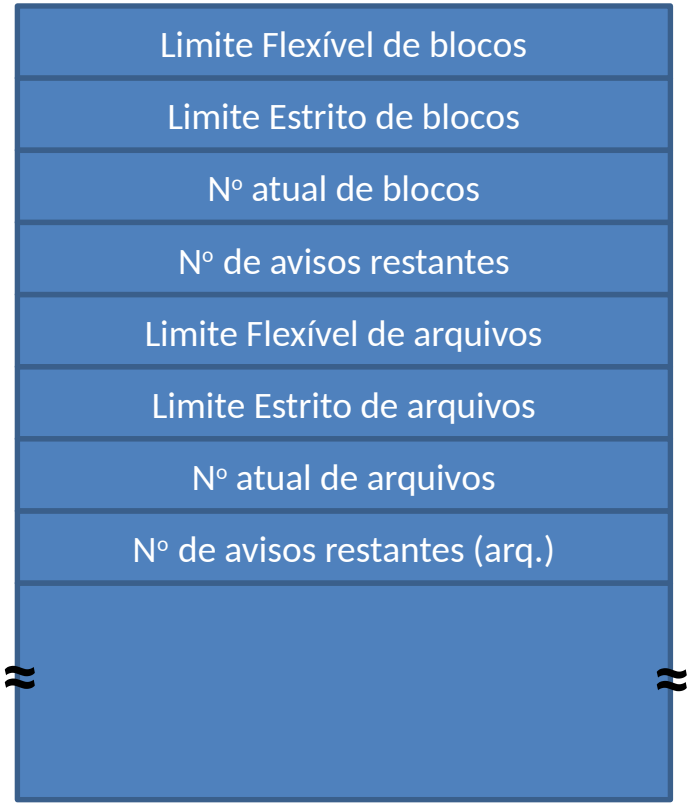


Tabela de Cotas

- Sistemas multiusuários implementam uma política de cotas para garantir que apenas uma porção do dispositivo por cada usuário;
- Cotas são controladas pelo sistema operacional.



Registro na tabela de Cotas para o usuário X

Segurança do Sistema de Arquivos

- Os dados são recursos importantes de um sistema computacional;
- Na verdade, os dados são provavelmente → recurso mais valioso das instituições (**Era da Informação, lembram?**);
- Falhas de hardware e software que levam a perda de (dados) informações são eventos catastróficos;
- Exemplo:
  - Ataque em 11/09/2001 - Dados financeiros perdidos durante a queda das torres gêmeas nos EUA;

Segurança do Sistema de Arquivos

- Problemas:
  - Recuperação em caso de desastre;
  - Recuperação em caso de falta do usuário;
- A caso de **desastre é auto-explicativo**:
  - Falha mecânica, falha elétrica, desastre natural, etc. São problemas que fisicamente destroem o hardware e consequentemente todos os dados contidos são perdidos;
- O caso de **falta do usuário é complicado:**
  - Muitas vezes o usuário remove arquivos que poderão ser necessários em um momento futuro;

Segurança do Sistema de Arquivos

- **Conceito Lixeira em SO:**
  - Arquivos removidos pelo usuário - não são diretamente removidos do S.A.;
  - Movidos para um diretório especial chamado “lixeira”;
  - Esses arquivos podem ser recuperados;
  - Arquivos são mantidos na lixeira por tempo (in) determinado;
  - Arquivos na lixeira são permanentemente removidos em dois casos:
    - Quando se faz necessário liberar espaço;
    - Quando o usuário explicitamente assim o estipula.

Segurança do Sistema de Arquivos

- **Cópia de segurança** dos dados de um ou mais sistemas de arquivos;

- Fazer cópia de todo o sistema de arquivos em uma unidade de backup é dispendioso e desnecessário:

  – Alguns arquivos simplesmente não precisam ser copiados, tais como arquivos do sistema ou de programas;

  – Arquivos de dados de usuário que não sofreram alteração desde o último backup não precisam ser copiados.

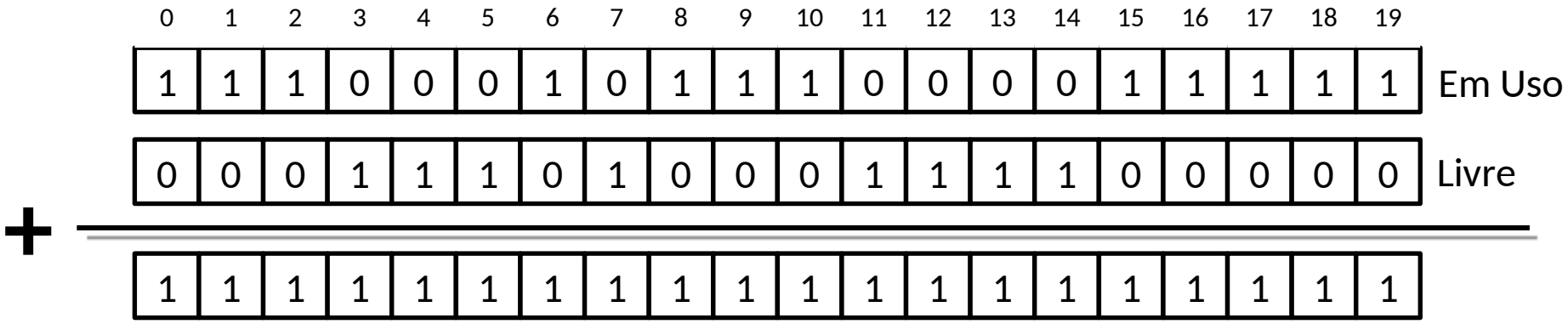
Cópias Incrementais

- **Copiar apenas os arquivos alterados** a partir do último backup;
- Vantagem: requer menos espaço de armazenamento do que sempre copiar todos os arquivos de dados do sistema;
- Desvantagens:
  - Faz-se necessário verificar e encontrar os arquivos alterados;
  - Geralmente em fita dat, dispositivo sequencial, o que consequentemente é mais lento;

Consistência do Sistema de Arquivos

- Atualizar arquivos em um sistema de arquivos não é uma **tarefa atômica**;
- Blocos devem ser lidos, armazenados na memória, alterados e reescritos para o dispositivo de armazenamento;
- O sistema de arquivos fica em um **estado inconsistente** caso ocorra uma falha durante esses passos;
- É comum os SOs fornecerem **aplicativos para verificação** da consistência do SA:
  - Linux – fsck (file system check)
  - Windows – scandisk

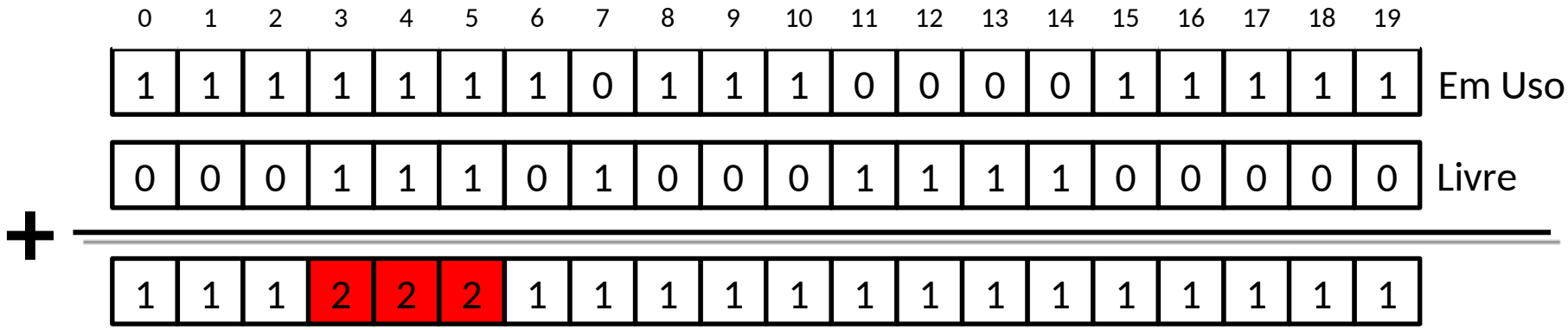
Consistência do Sistema de Arquivos



Exemplo:

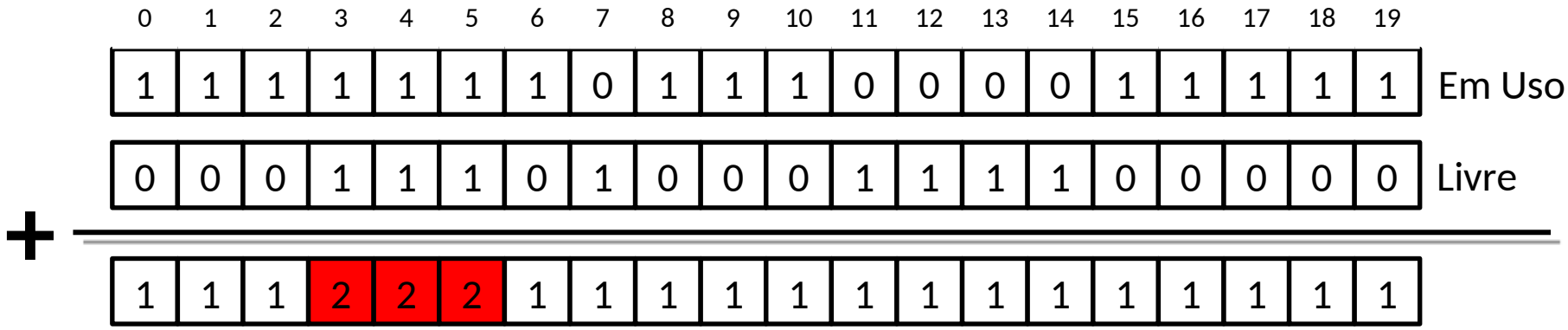
- Todos os blocos aparecem na **tabela de blocos em uso** ou **na tabela de blocos livres** apenas uma vez;
- Esta configuração indica que o sistema de arquivos está em **um estado consistente**.

Inconsistência do SA



- Blocos 3–5 estão sendo usados **por um arquivo** e listados como **livres**;
- Esta configuração se configura como um estado inconsistente;

Inconsistência do SA



- Estado inconsistente;
- Isso pode significar que o sistema gerou problema durante a remoção do arquivo ou qualquer outra razão não especificada;
- A forma de corrigir este estado de inconsistência fica a critério do SO → Uma possibilidade seria simplesmente remover os blocos 3–5 da lista de blocos livres;

Desempenho do Sistema de Arquivos

- Acesso a disco normalmente é mais lento que o acesso à memória;
  - Os HDDs podem ter uma taxa de acesso em torno de 10MB/s;
  - A memória RAM pode alcançar taxas de 400MB/s (os HDDs são 40x mais lento);
  - Normalmente, 5 a 10ms devem ser computados para o posicionamento da cabeça de leitura.
- Uso de otimizações de acesso pode melhorar o desempenho :
  - Cache de blocos;
  - Leitura antecipada de blocos;
  - Redução do movimento do braço do disco;

Cache de Blocos

- Técnica mais usada para reduzir acesso ao disco;
- Neste contexto, uma cache é uma coleção de blocos que, do ponto de vista lógico, pertencem ao disco, mas que estão sendo mantidos na memória principal para fins de desempenho;
- Vários algoritmos existem para o gerenciamento da cache de disco;

Cache de Blocos

- O algoritmo mais comum:
  - Verificar todas as requisições de leitura de modo a identificar se o bloco desejado se encontra na cache;
  - Caso o bloco esteja na cache, a requisição de leitura pode ser atendida sem um acesso ao disco;
  - Caso o bloco não esteja na cache, ele primeiro será lido do disco repassado ao processo requisitante. Subsequentemente ele será copiado para a cache;

Leitura Antecipada de Blocos

- Busca transferir os blocos para a cache de disco antes que eles sejam necessários;
- Tenta aumentar a taxa de acertos;
- Devido a fatores históricos, muitos programas acessam arquivos de maneira sequencial;
- Quando um bloco k é solicitado ao sistema de arquivos dele o busca e verifica se o bloco k+1 já se encontra na cache. Em caso negativo, ele agenda para recuperação;
- Para acesso aleatório, esta estratégia piora o tempo de acesso.

Redução do Movimento do Braço

- Os pratos de um HDD podem girar numa alta velocidade;
- No entanto, o braço de leitura se movimenta a uma velocidade consideravelmente inferior;
- Minimizar o número de movimentos do braço para efetuar a leitura dos blocos de um arquivo pode otimizar em muito o desempenho do sistema de arquivos;
- Isso pode ser alcançado via uma reorganização dos blocos dos arquivos no disco;

Redução do Movimento do Braço

- Para sistemas que utilizam os *i-nodes*, são necessários dois acessos: uma para o bloco e outro para o *i-node*;

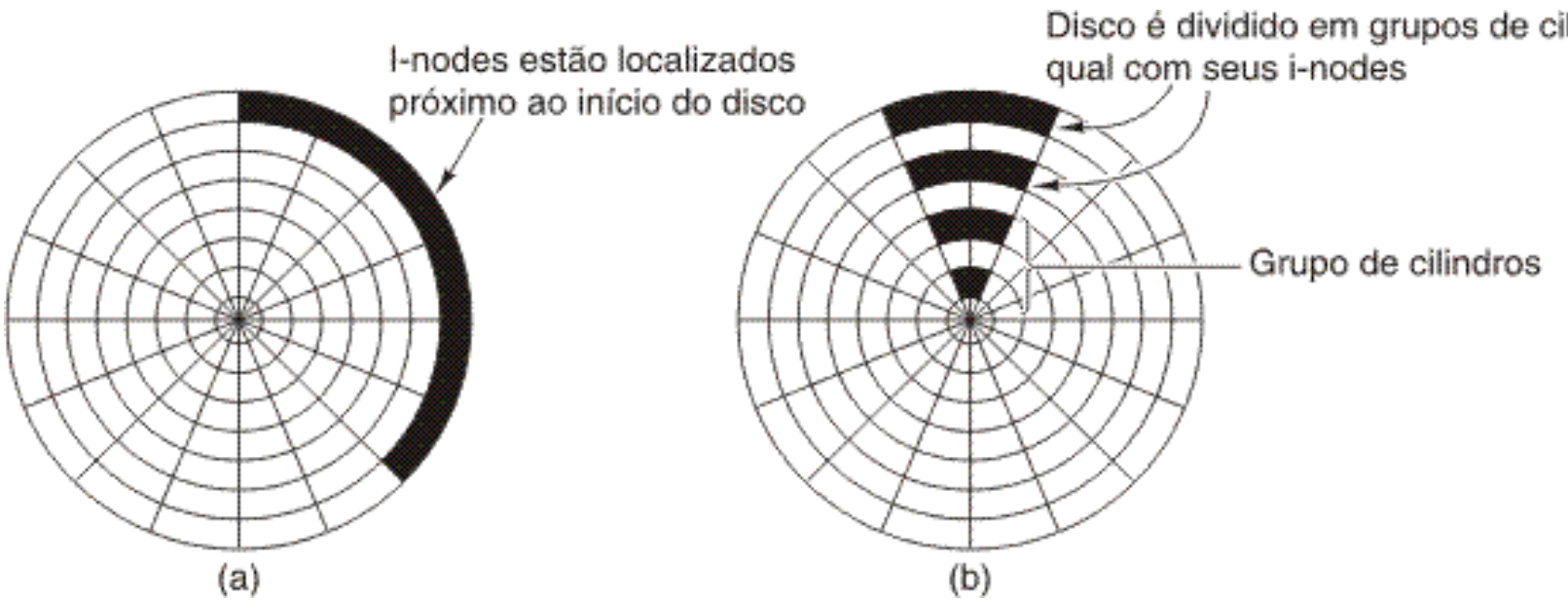
- Três estratégias podem ser utilizadas para armazenamento dos *i-nodes*:

  - Os *i-nodes* são colocados no início do disco;

  - Os *i-nodes* são colocados no meio do disco;

  - Dividir o disco em grupos de cilindros, nos quais cada cilindro tem seus próprios *i-nodes*, blocos e lista de blocos livres (*bitmap*);

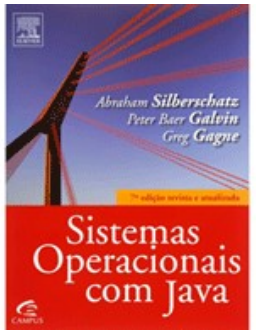
Redução do Movimento do Braço



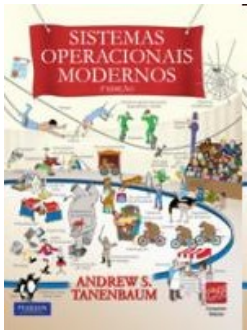
1. I-nodes colocados no início do disco
2. Disco dividido em grupos de cilindros cada qual com seus próprios blocos e i-nodes

Leituras Sugeridas

- Silberschatz, A., Galvin, P. B. Gagne, G. Sistemas Operacionais com Java. 7º edição. Editora Campus, 2008 .



- TANENBAUM, A. Sistemas Operacionais Modernos. Rio de Janeiro: Pearson, 3 ed. 2010



- Material de Aula do Prof. Dr. Daniel Abdala.

Agradecimento

- Prof. Dr. Daniel Abdala pelo material de aula disponibilizado.