

Sistemas Operacionais

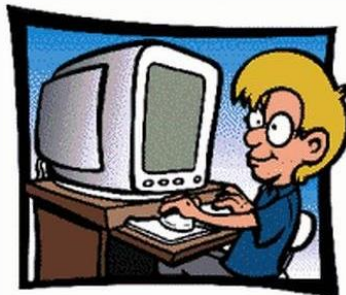
Implementação do Sistema de Arquivos

Visões: Usuário vs Projetista do S.O.

Usuário

Quais operações são permitidas.

Como a árvore de diretórios é manipulada pela interface gráfica.



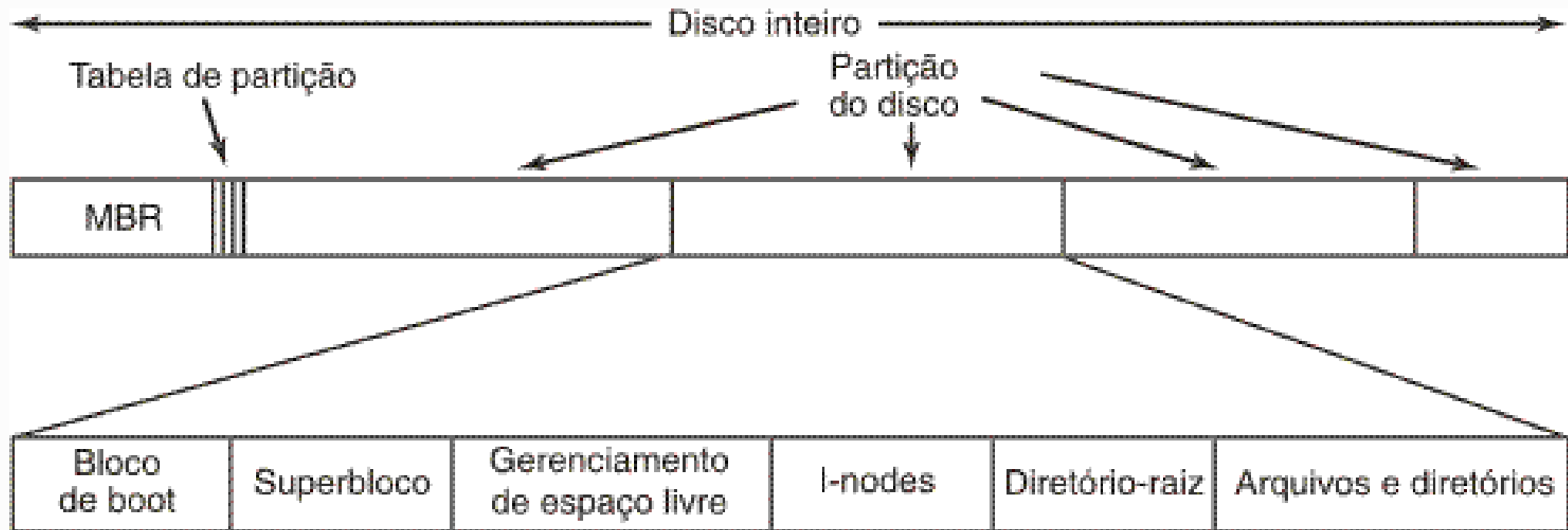
Projetista de S.O.

Como são armazenados os arquivos e diretórios no disco.

Quais são as estruturas mais eficientes.

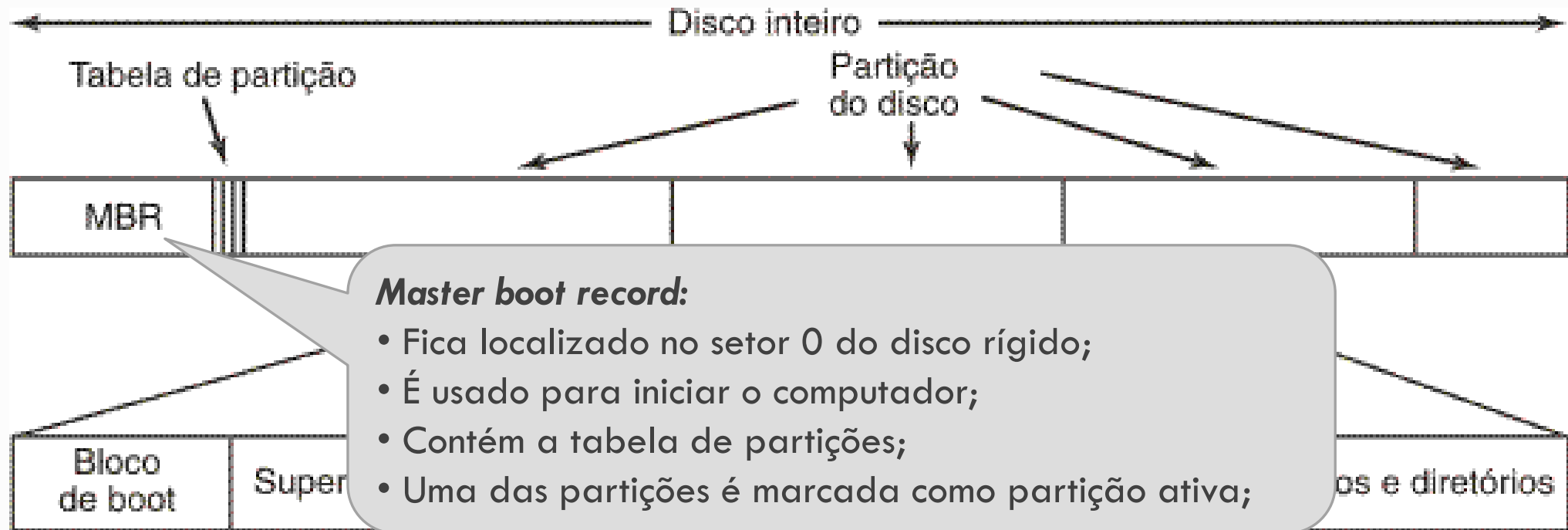


Implementação do Sistema de Arquivos



Um possível *layout* de Sistema de Arquivos

Implementação do Sistema de Arquivos



Um possível *layout* de Sistema de Arquivos

Implementação do Sistema de Arquivos

Quando o computador é iniciado, a BIOS lê e executa o MBR.

- O primeiro passo é localizar a partição ativa.
- Cada partição possui um bloco de *boot*.
- O sistema operacional carregado é aquele que é referenciado no bloco de *boot* da partição ativa.
- Portanto, erros no *boot* podem estar localizados em dois lugares diferentes:
 - Na MBR ou;
 - No bloco de *boot* da partição ativa.

Implementação do Sistema de Arquivos

Caracteriza-se por permitir até quatro partições, ditas primárias.

Caso seja necessário um número maior, pode-se usar uma partição como estendida.

- Neste caso, essa partição será um repositório de unidades lógicas ou partições lógicas.

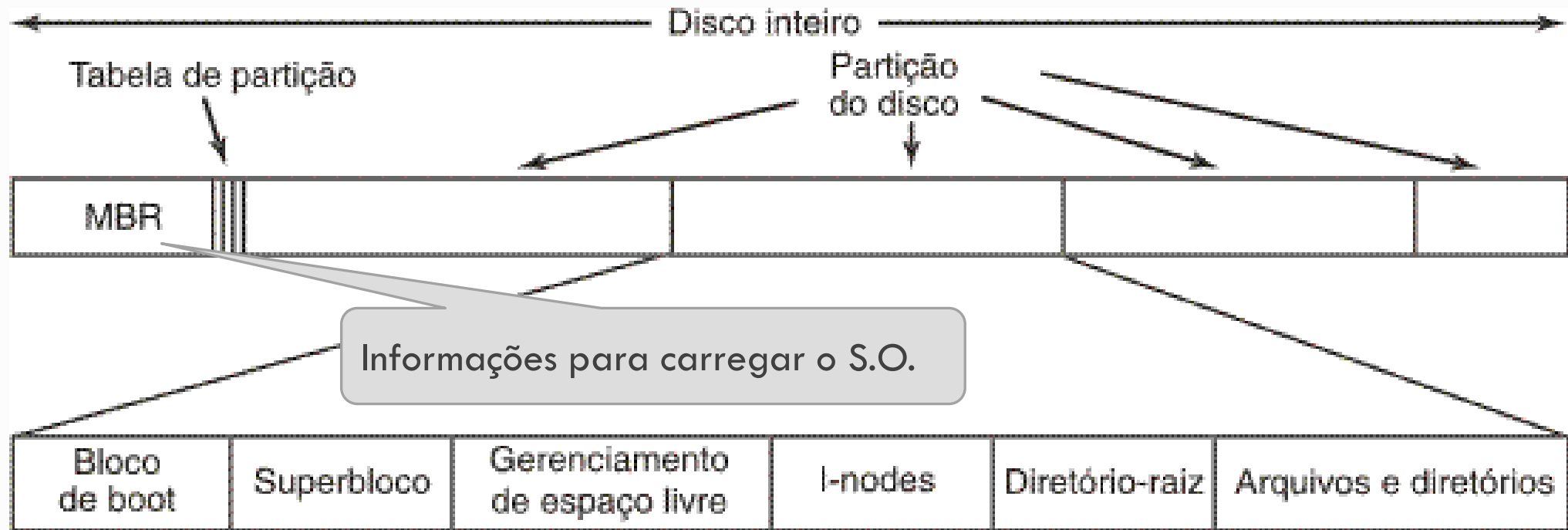
Implementação do Sistema de Arquivos

Nem todas as partições precisam ter um S.O. (o que é bem comum nos dias de hoje):

- C:\ D:\ E:\ (na Família Microsoft);
- /boot; /home; /usr; /tmp; /var; /var/temp; /tmp/; etc (na Família UNIX)

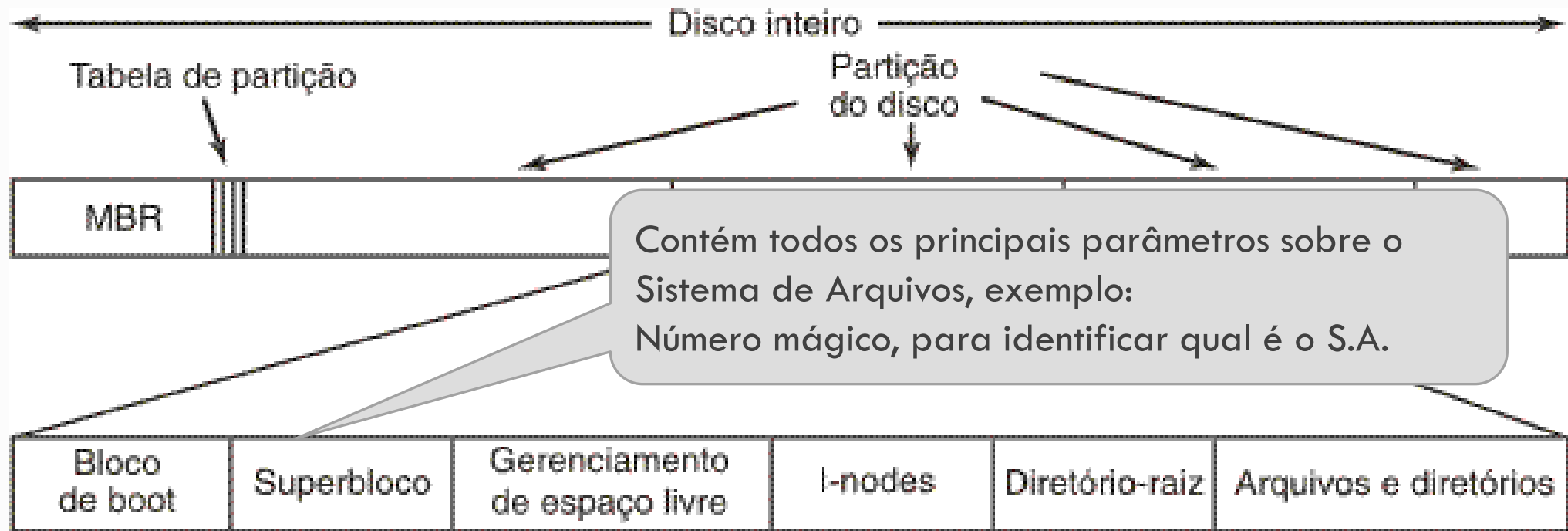
Mas é interessante reservar um bloco de boot no espaço reservado para cada partição, pois futuramente aquela partição poderá receber a instalação de um S.O..

Implementação do Sistema de Arquivos



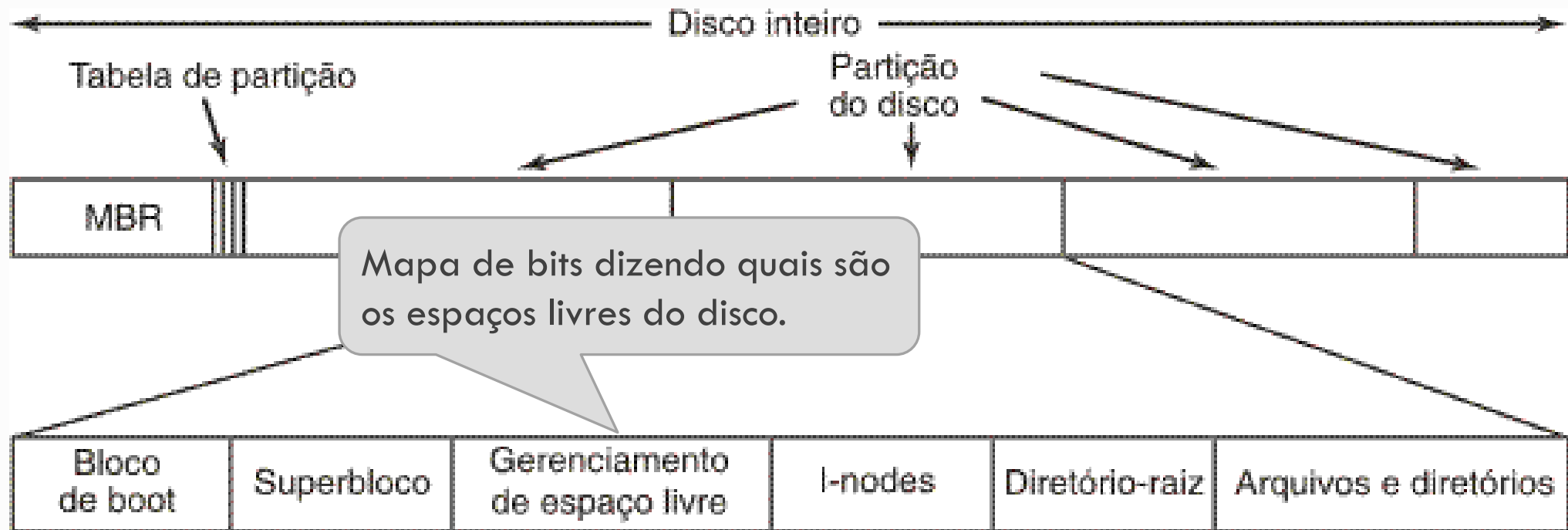
Um possível *layout* de Sistema de Arquivos

Implementação do Sistema de Arquivos



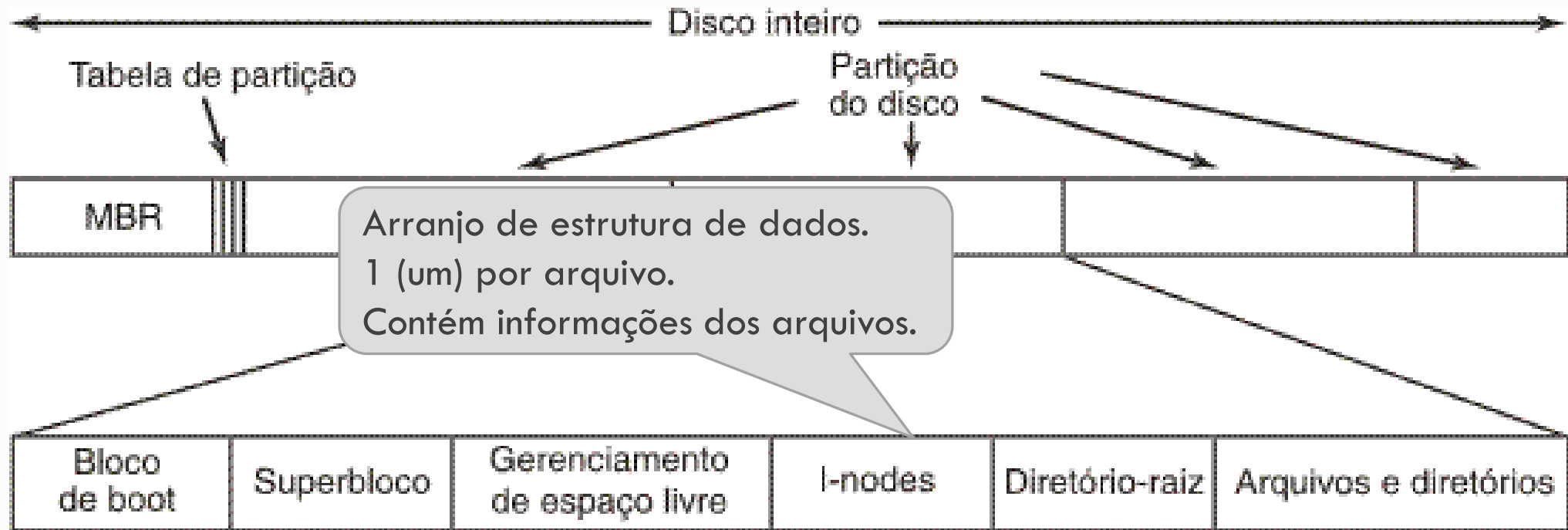
Um possível *layout* de Sistema de Arquivos

Implementação do Sistema de Arquivos



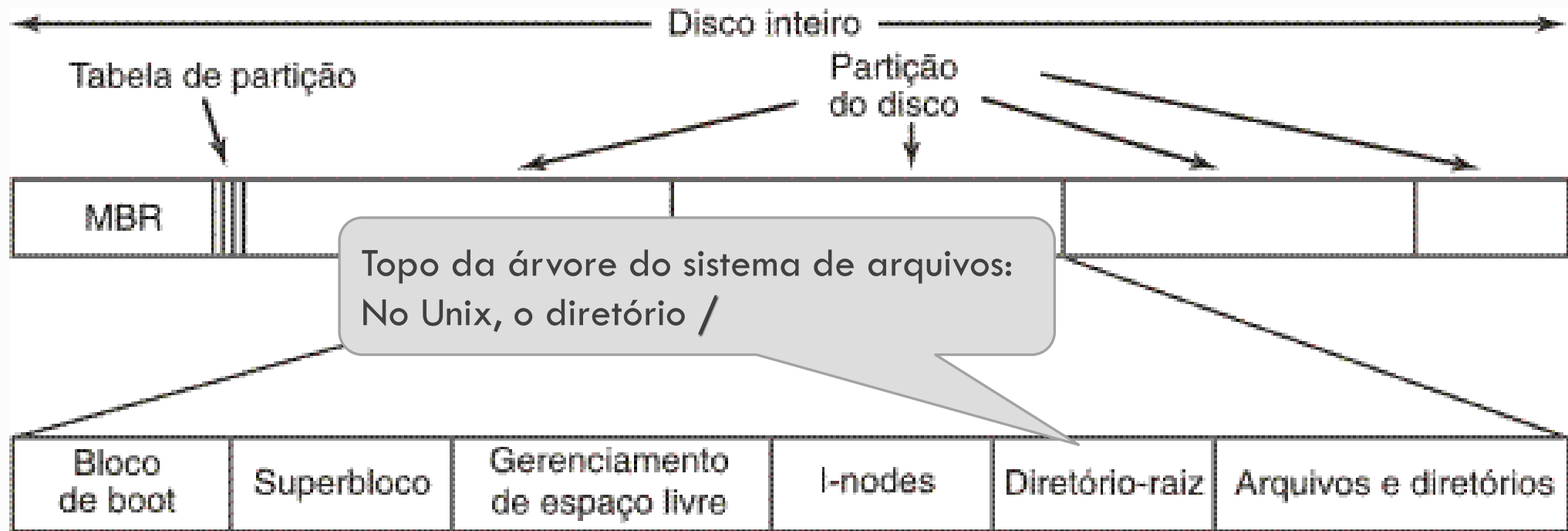
Um possível *layout* de Sistema de Arquivos

Implementação do Sistema de Arquivos



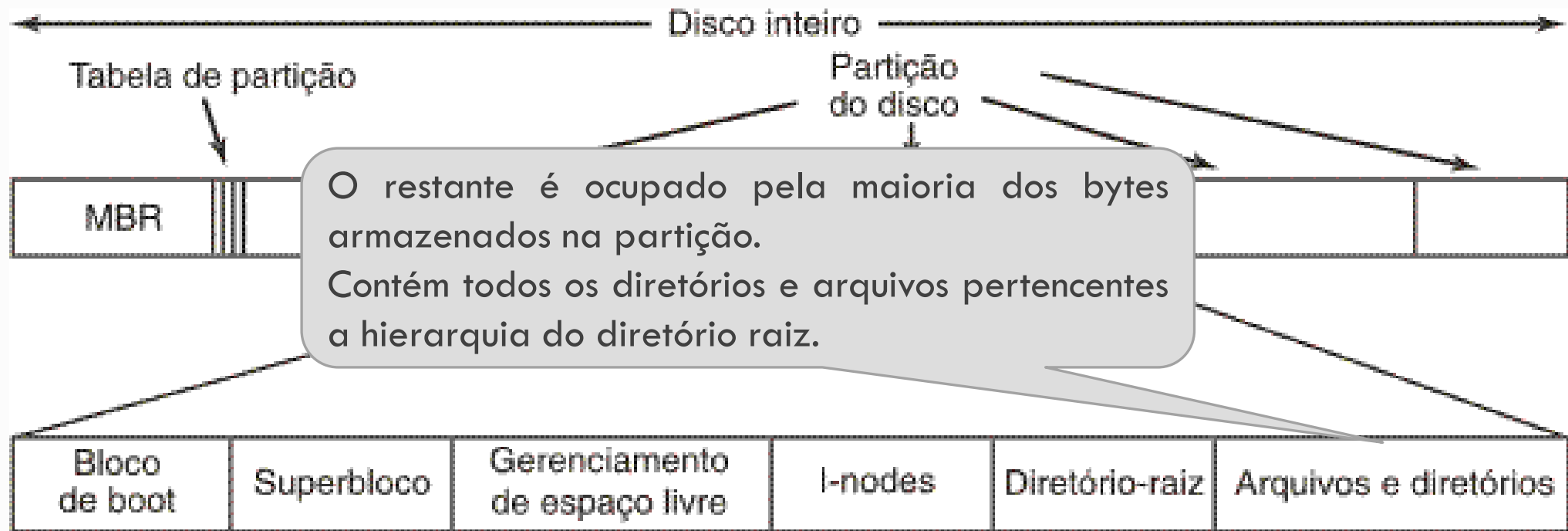
Um possível *layout* de Sistema de Arquivos

Implementação do Sistema de Arquivos



Um possível *layout* de Sistema de Arquivos

Implementação do Sistema de Arquivos

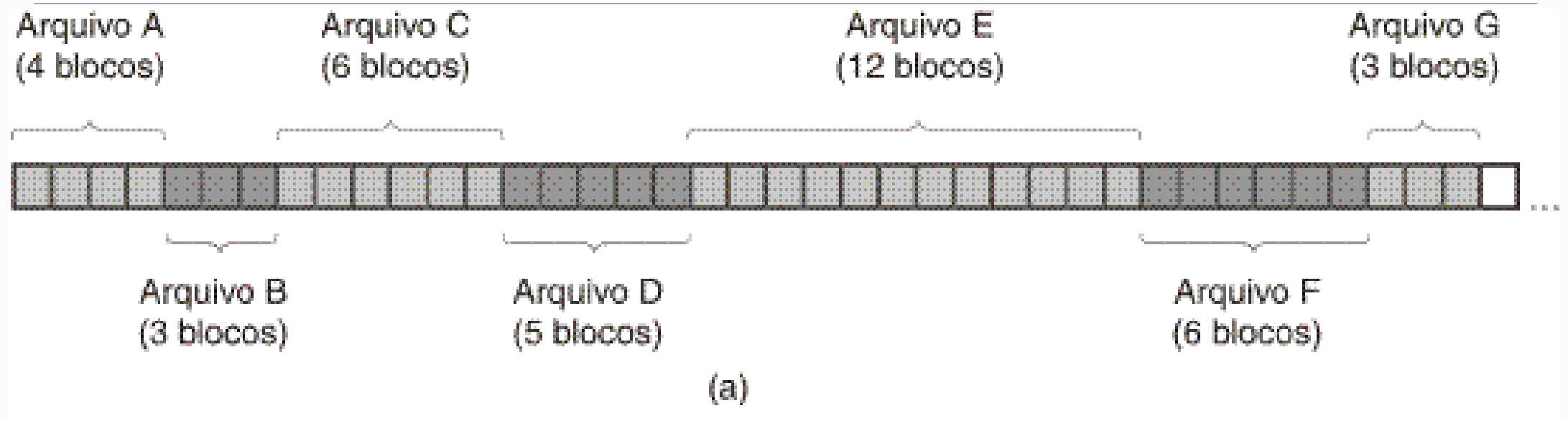


Um possível *layout* de Sistema de Arquivos

Implementação de Arquivos

Alocação Contígua

Alocação Contígua

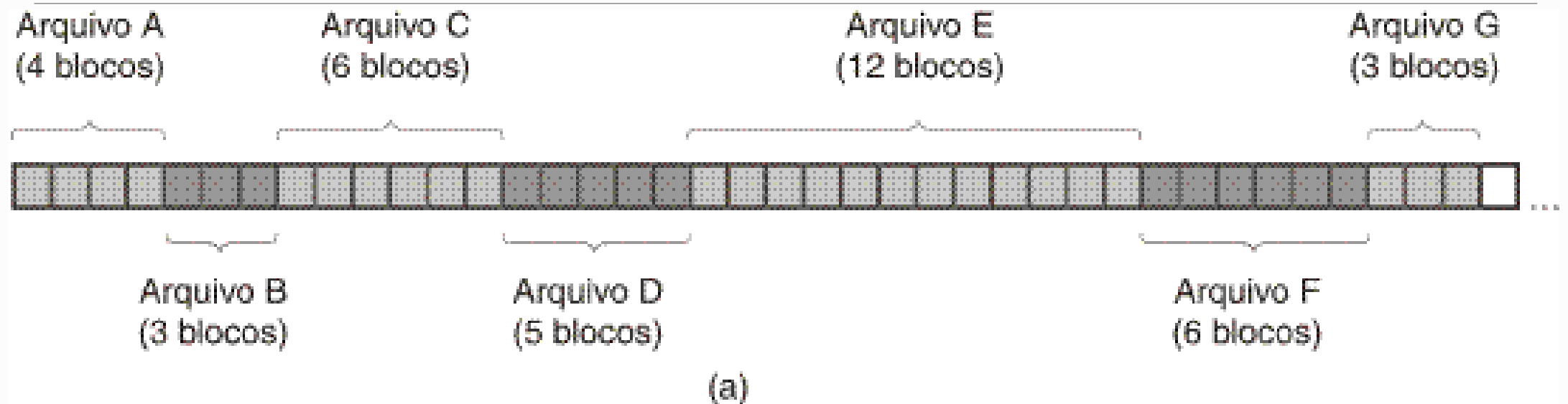


7 arquivos alocados sequencialmente no disco.

O algoritmo para armazenamento é muito simples.

Sempre alocar o próximo arquivo depois da última posição ocupada.

Alocação Contígua

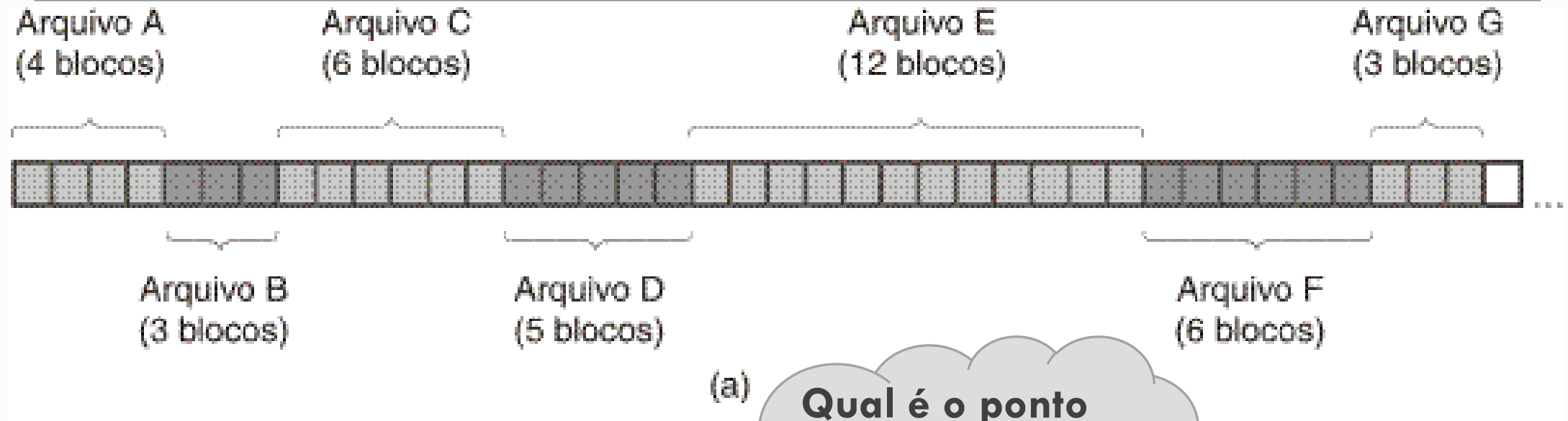


O desempenho de leitura é excelente.

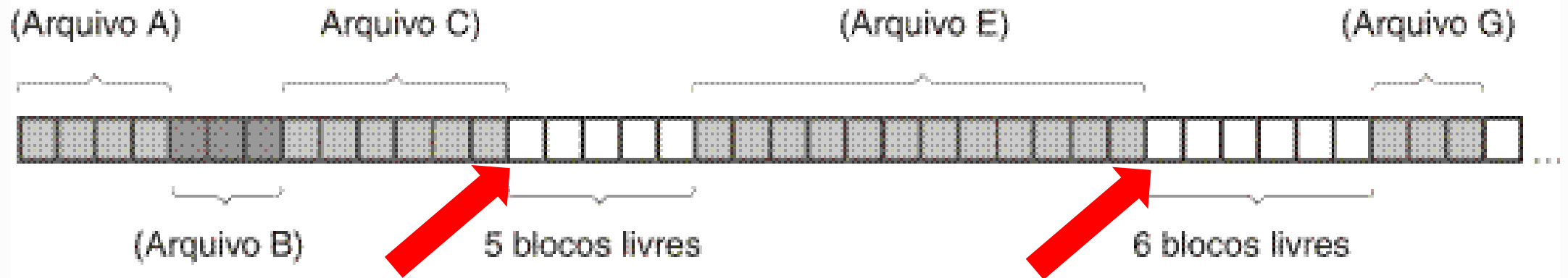
É preciso saber apenas: (a) O endereço do disco do primeiro bloco e; (b) o número de blocos do arquivo.

A chamada de sistema `seek` posiciona o cursor de leitura na posição correta em tempo $O(1)$.

Alocação Contígua

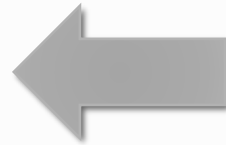


Alocação Contígua

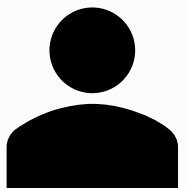


(b)

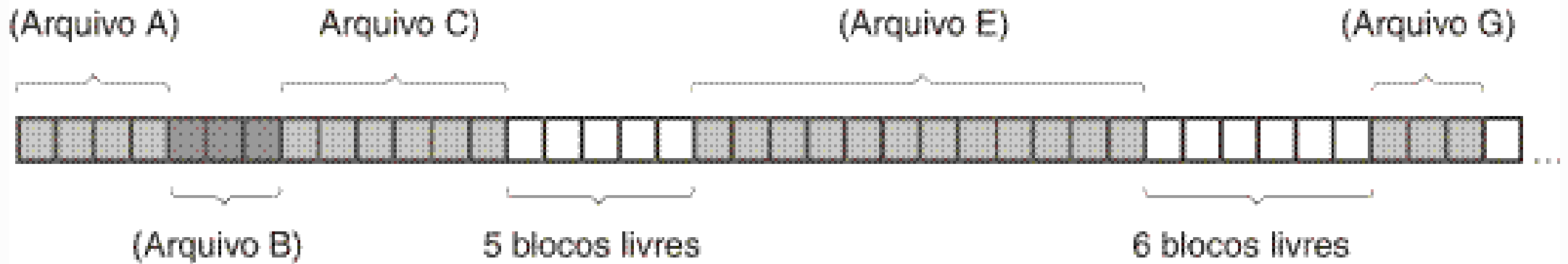
**Fragmentação
de Disco**



**Qual é o ponto
fraco da
alocação
contígua?**

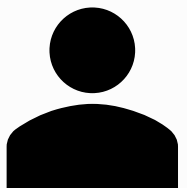


Alocação Contígua

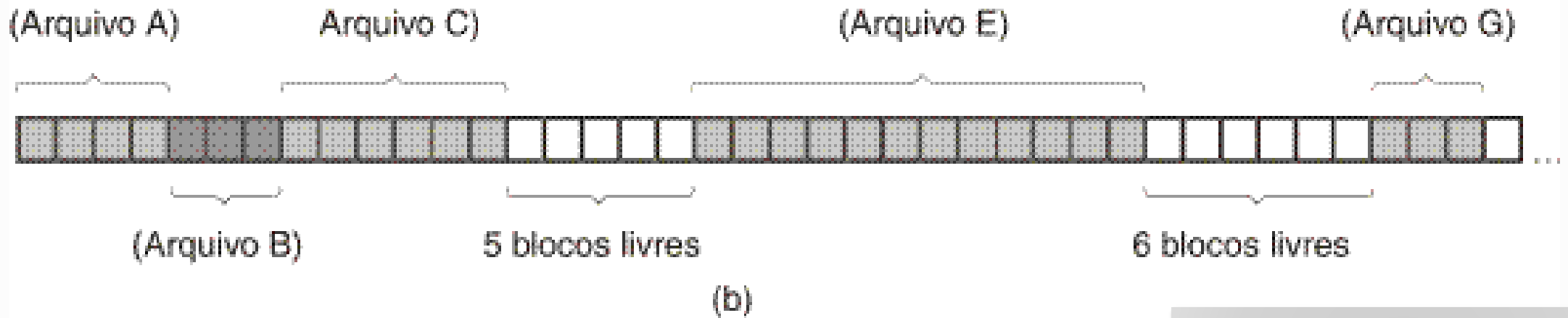


Obviamente, ao se apagar um arquivo, o disco não pode ser desfragmentado imediatamente.

Qual seria a complexidade desta operação?



Alocação Contígua



Mantendo as lacunas, ao se criar um novo arquivo, é necessário saber o seu tamanho para indicar em qual lacuna ele será alocado.

Qual política é melhor?

Deixando a maior lacuna possível?

Deixando a menor lacuna possível?

Alocação Contígua

A alocação contígua foi largamente utilizada em antigos S.O.s, por sua simplicidade na implementação.

Deixou de ser utilizada porque era necessário saber o tamanho do arquivo na hora de sua criação.

Mas, com o surgimento das mídias ópticas somente para escrita (CD-ROM, DVD, Blu-rays), a alocação contígua passou a ser novamente uma boa ideia.

Implementação de Arquivos

Alocação por Lista Encadeada

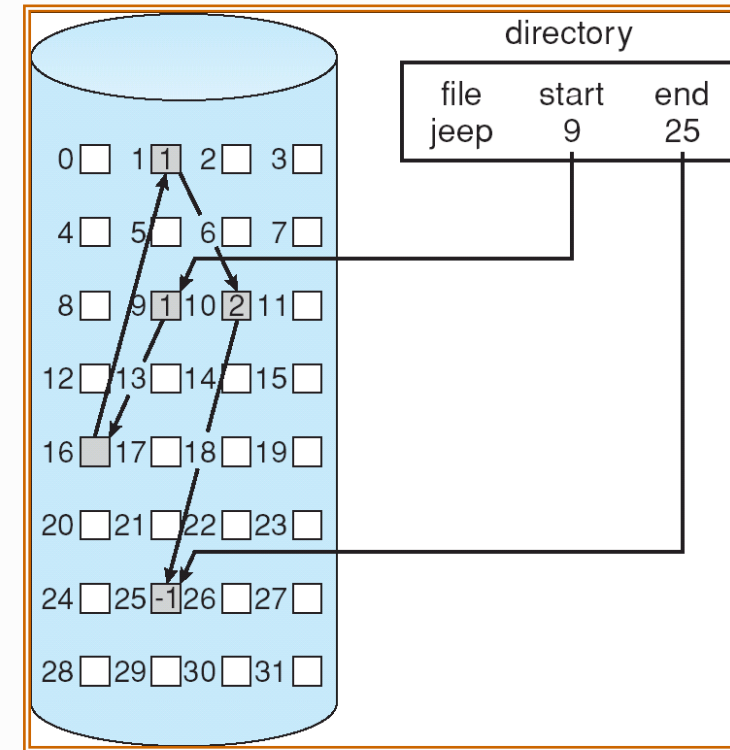
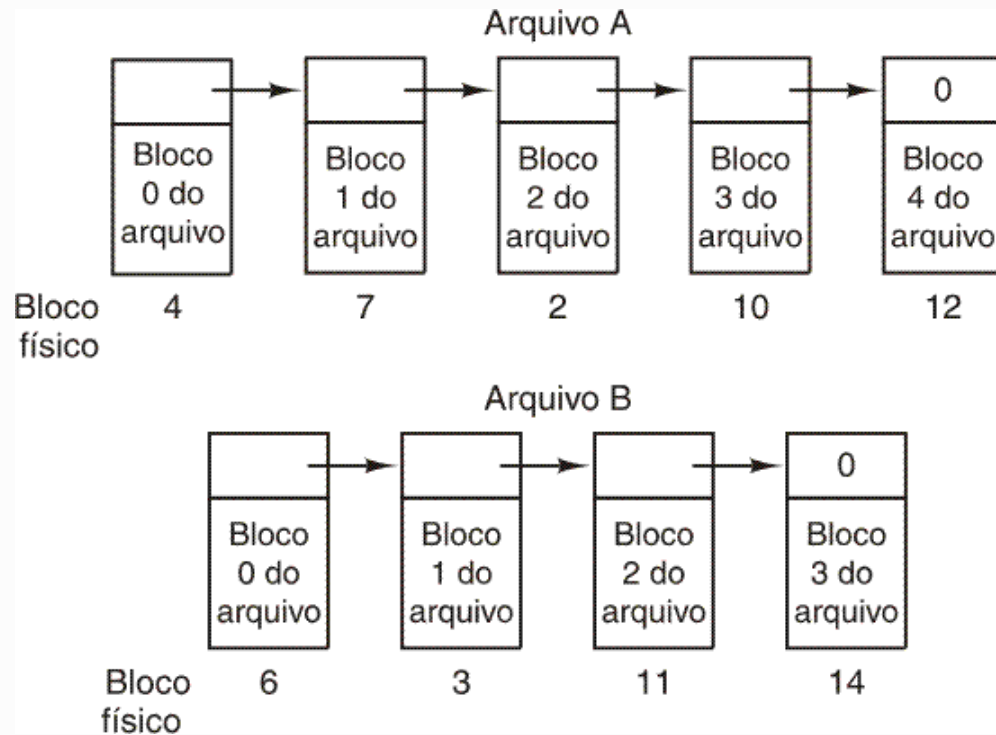
Alocação por Lista Encadeada

O objetivo da alocação por lista encadeada é:

- Reduzir o processo de fragmentação gradual do disco;

Os arquivos não precisam estar necessariamente armazenados em uma sequência ininterrupta de blocos.

Alocação por Lista Encadeada



Armazenamento de um arquivo como uma lista encadeada de blocos de disco

Alocação por Lista Encadeada

Assim, os arquivos do tipo diretório precisam saber apenas o primeiro endereço de cada arquivo.

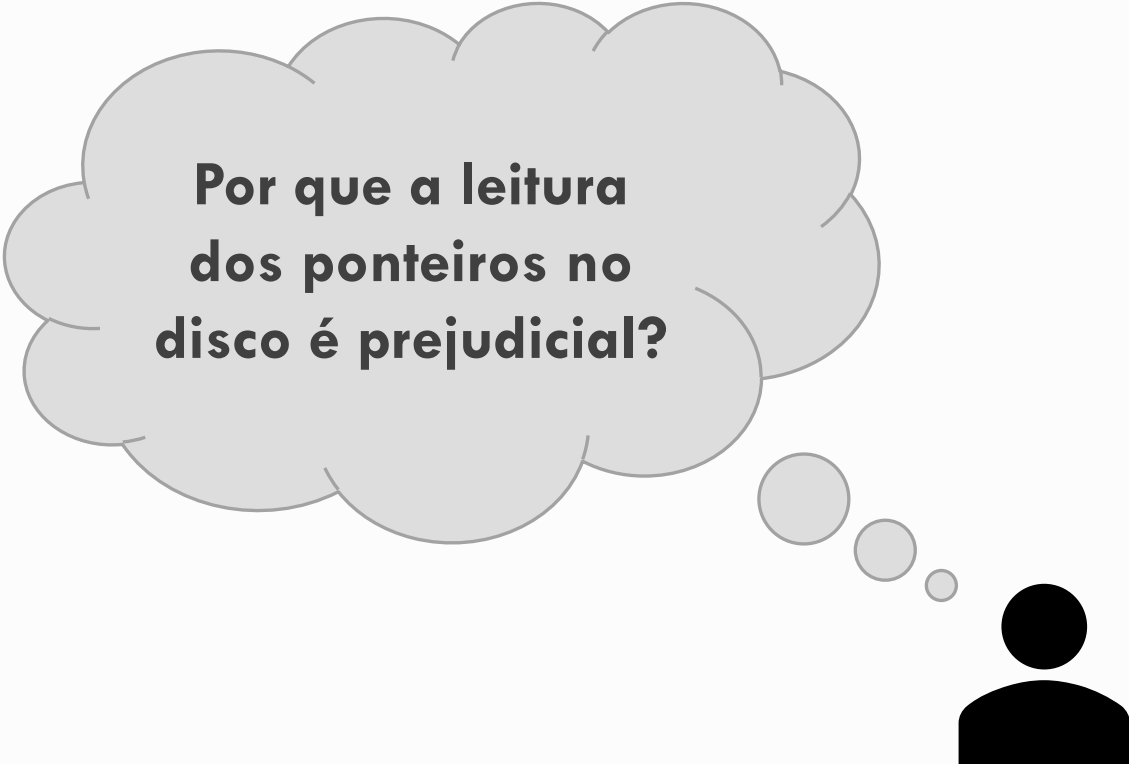
O restante do arquivo pode ser encontrado a partir do primeiro bloco.

Por outro lado, a leitura de todo arquivo utilizando acesso aleatório passa a ser mais lenta.

- Somente operações em disco!

Alocação por Lista Encadeada

Além disso, a quantidade de dados que um bloco pode armazenar deixa de ser potência de 2, pois se perde com a alocação do apontador para o próximo bloco.



**Por que a leitura
dos ponteiros no
disco é prejudicial?**

Alocação por Lista Encadeada

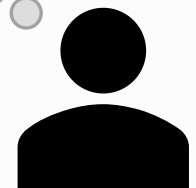
Imagine que você deseja ler apenas os últimos bytes de um arquivo.

E seu arquivo possui quase 500 MB.

O S.O. precisa (em disco), navegar por todos os blocos até localizar os últimos blocos do arquivo.



**Por que a leitura
dos ponteiros
no disco é
prejudicial?**



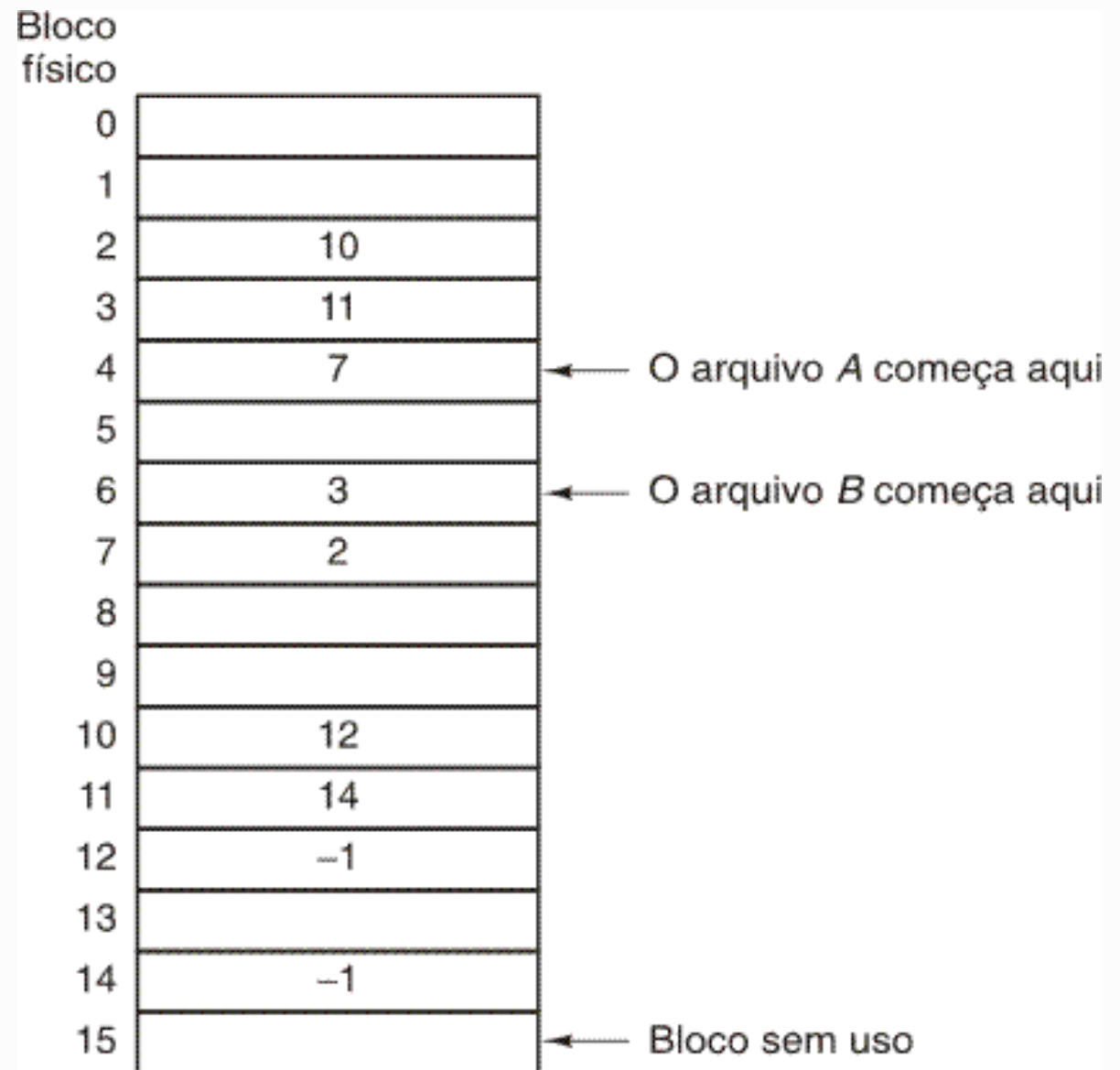
Alocação por Lista Encadeada

Para resolver o problema de velocidade no acesso aleatório, foi proposta a tabela de alocação de arquivos.

O acesso aos ponteiros é feito na memória principal.

FAT (*File Allocation Table*).

Alocação por Lista Encadeada



Alocação por lista encadeada usando uma tabela de alocação de arquivos em RAM

Alocação por Lista Encadeada

A complexidade continua sendo a mesma para o acesso aleatório: $O(n)$

Mas agora, todo acesso é feito diretamente na memória principal (RAM).

A principal desvantagem deste método é que toda a tabela deve estar em memória o tempo todo.

- Para um disco rígido de 20 GB, a tabela de alocação de arquivos (FAT) ocuparia até 80 MB da memória principal.

Próxima aula

Leitura:

Sistemas operacionais modernos

Implementação de Diretórios

Referências

Sistemas Operacionais Modernos. Tanenbaum, A. S. 2ª edição. 2003.

Sistemas Operacionais. Conceitos e Aplicações. A. Silberschatz; P. Galvin; G. Gagne. 2000.

Sistemas Operacionais – Projeto e Implementação. Tanenbaum, A. S. 2ª edição. 2000.

Slides Prof. Humberto Brandão