Sistemas Operacionais
Arquivos

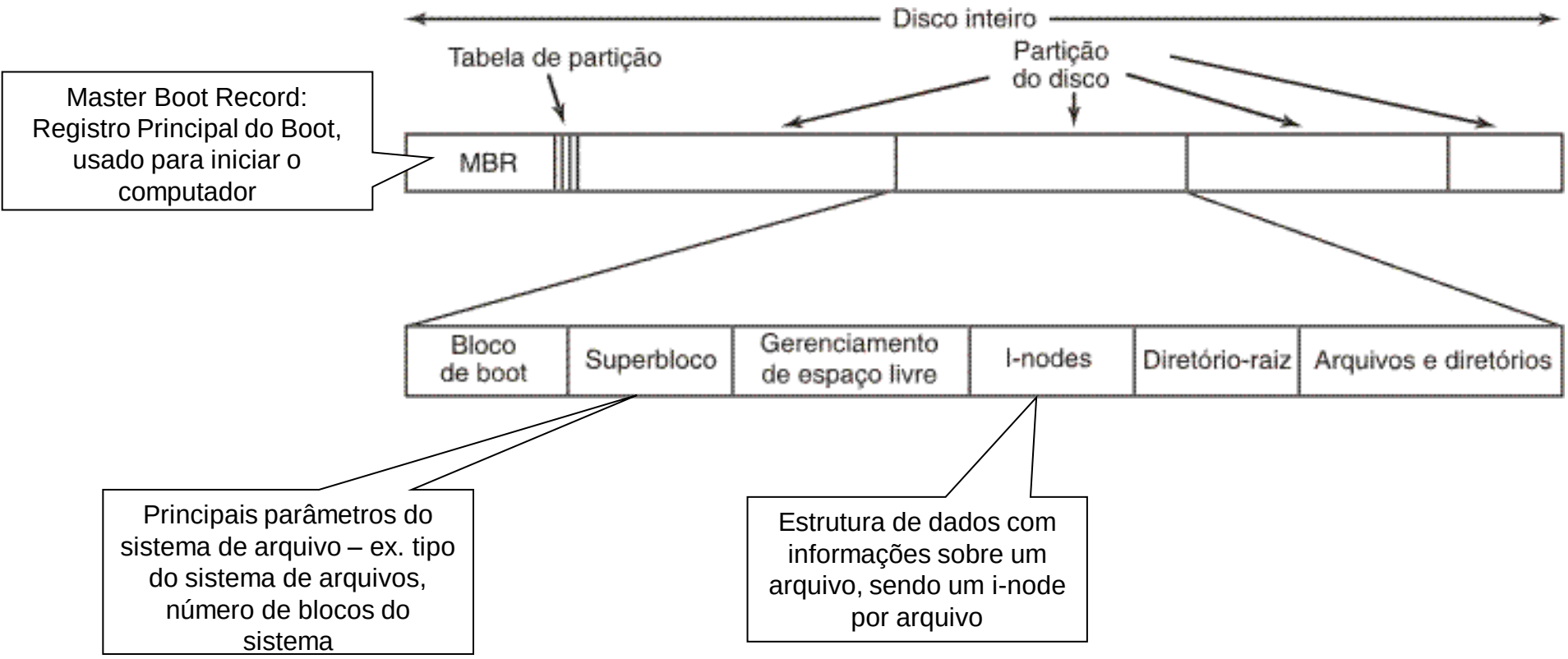
Carlos Ferraz (cagf@cin.ufpe.br)
Jorge Cavalcanti Fonsêca (jcbf@cin.ufpe.br)

Copyright

Carlos Ferraz – Cin/UFPE

Implementação do Sistema de Arquivos

Sistemas de arquivos são armazenados em disco



Um possível layout de sistema de arquivo

Alocação de Espaço em Disco

- A criação de arquivos exige:
  - que o sistema operacional tenha controle de quais áreas ou blocos no disco estão livres

- Este controle é realizado através de uma estrutura (geralmente lista ou tabela) de dados que armazenam informações e possibilitam ao sistema de arquivos gerenciar o espaço livre

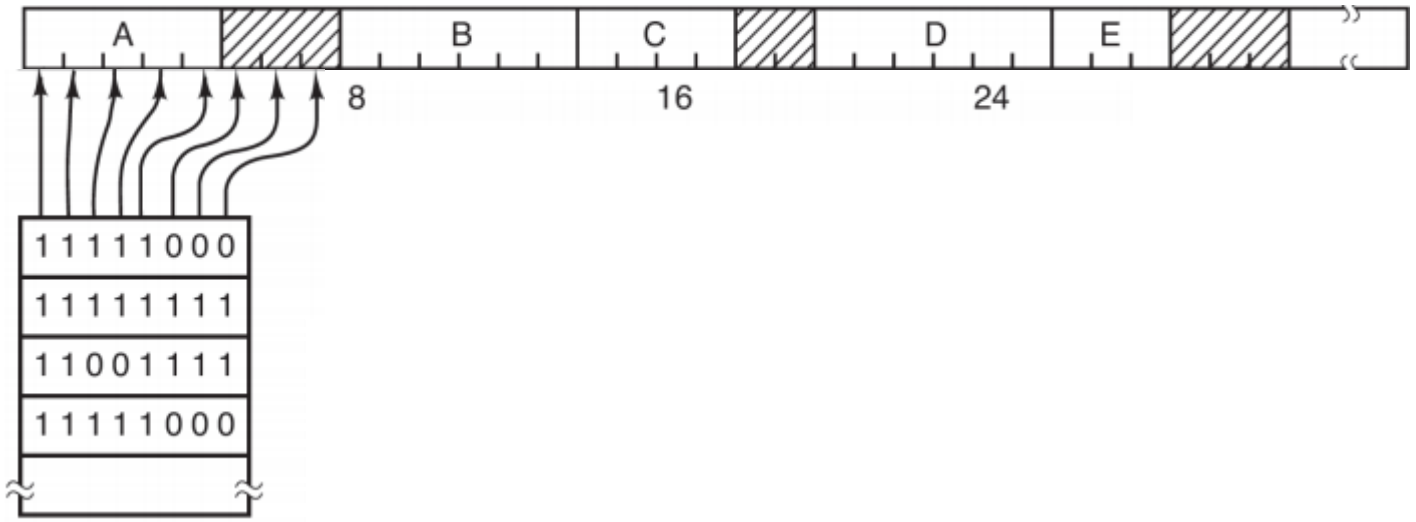
Alocação de Espaço em Disco

- A forma mais simples de implementar uma estrutura de espaços livres é através de uma tabela chamada **mapa de bits** (*bit map*)
  - cada entrada da tabela é associada a um bloco e representado por um bit, que pode assumir valor igual a 0 (bloco livre) ou 1 (bloco alocado).

- Esta estrutura gera um gasto excessivo de memória já que para cada bloco deve existir uma entrada na tabela.

Slide aula memória
Gerenciando Memória Livre

- Mapa de Bits



- Tamanho da unidade de locação é inversamente proporcional ao tamanho do mapa de bits
- Unidade grande = muita memória desperdiçada
- Simples
- Problema para encontrar sequência de bits livres (busca muito lenta)

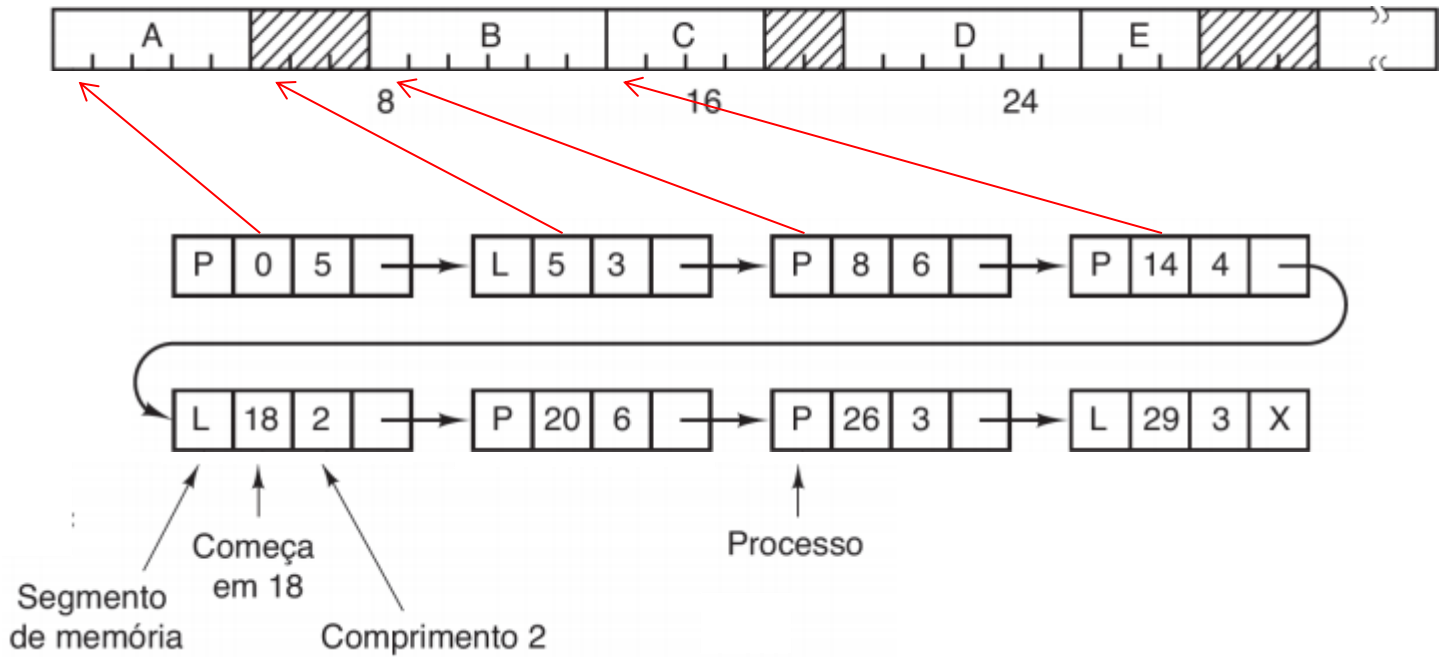
Alocação de Espaço em Disco

- Outra forma é realizar o controle por meio da ligação encadeada de todos os blocos livres e cada bloco deve possuir uma área reservada para armazenamento do endereço do próximo

- A partir do primeiro bloco pode-se ter acesso seqüencial aos demais de forma encadeada
  - Apresenta restrições se considerarmos que o algoritmo de busca de espaço livre sempre deve realizar uma pesquisa seqüencial na lista

Slide Aula Memória
Gerenciando Memória Livre

- Lista Encadeada
  - Lista com posições de memória livre



Alocação de Espaço em Disco

- Outra solução leva em conta que blocos contíguos são geralmente alocados ou liberados simultaneamente
  - com base neste conceito é possível manter uma **tabela** com o endereço do primeiro bloco de cada segmento e o número de blocos livres contíguos que se seguem

Bloco	Contador
1	1
5	3
12	1
15	1
17	1

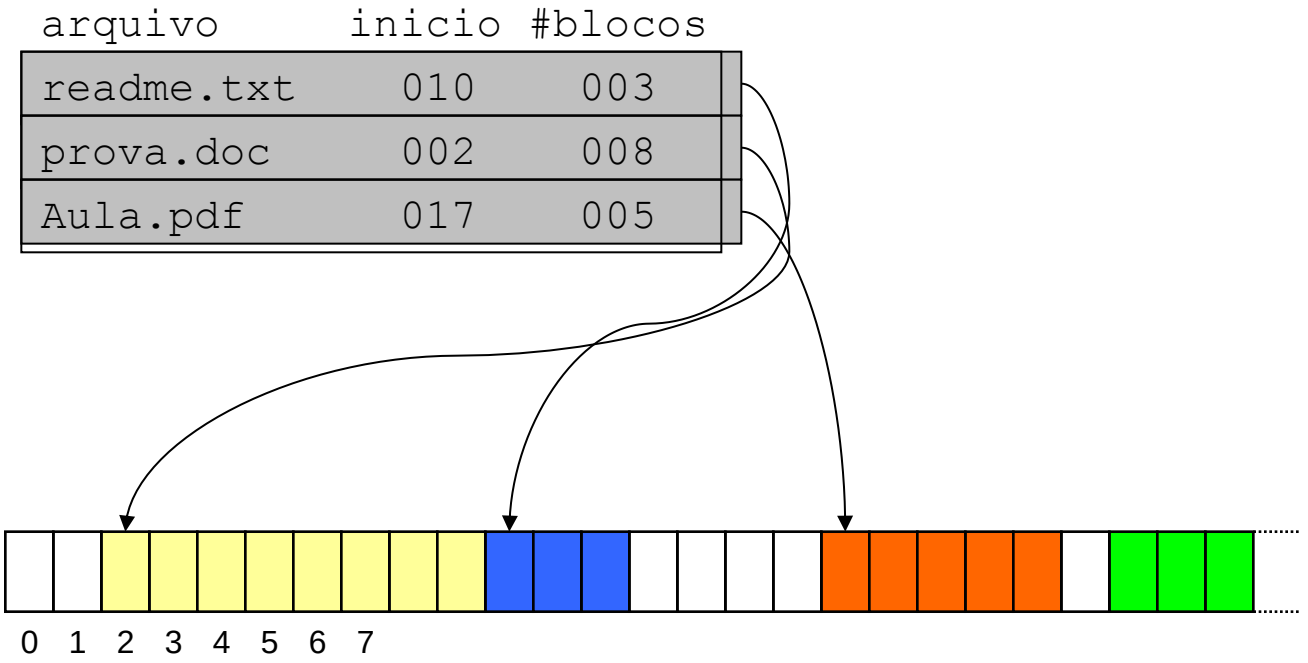
Tabela de Blocos Livres

Alocação Contígua de um Arquivo

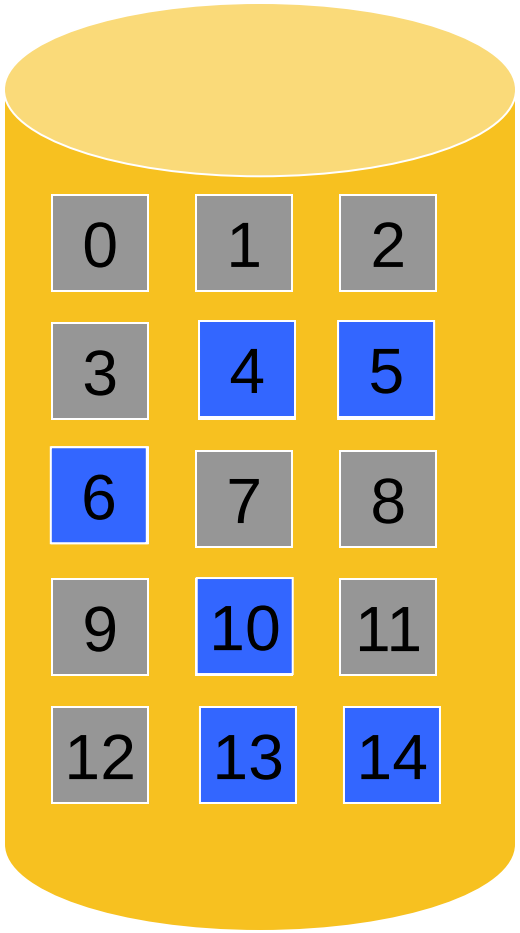
- Consiste em armazenar um **arquivo** em blocos seqüencialmente dispostos
  - o sistema localiza um arquivo através do endereço do primeiro bloco e da sua extensão em blocos

- O acesso é bastante simples tanto para a forma seqüencial tanto para a direta
  - o principal problema é a alocação de novos arquivos nos espaços livres, pois para colocar **n** blocos é necessário que se tenha uma cadeia com **n** blocos dispostos seqüencialmente no disco

Alocação Contígua



Alocação Contígua



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

Alocação Contígua

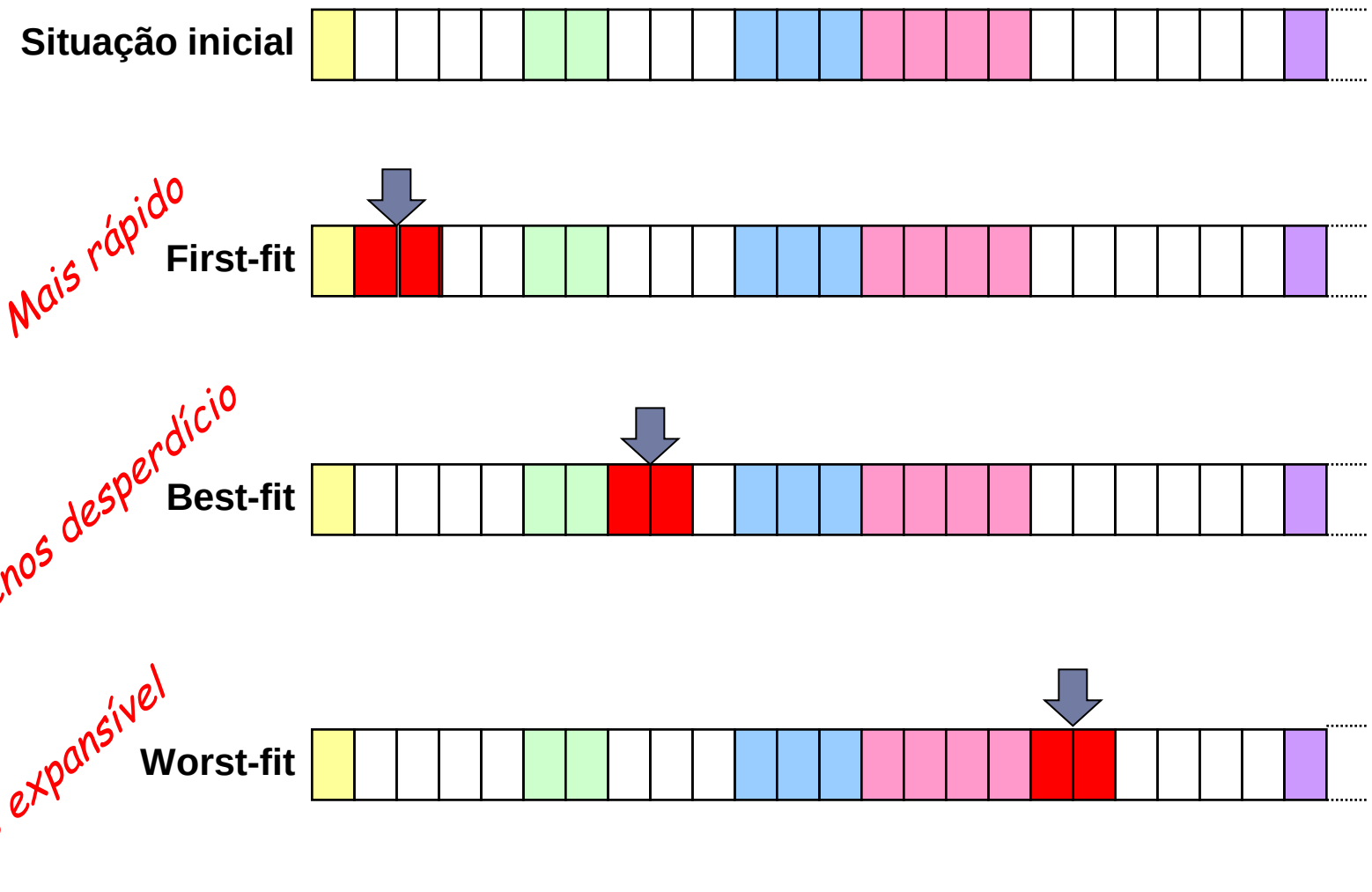
- Existem alguns problemas como:
  - determinar o espaço necessário para um arquivo quando é criado e depois existir a necessidade de extensão ➔ esta é uma operação complexa
  - a pré-alocação seria uma solução, mas pode levar a parte do espaço alocado a permanecer ociosa por um longo período de tempo...
- Quando o sistema operacional deseja alocar espaço para um novo arquivo, pode existir mais de um segmento livre disponível com o tamanho exigido e é necessário que alguma **estratégia de alocação** seja adotada para selecionar qual segmento deve ser escolhido

Esse tipo de alocação é válido em algum cenário?

Alocação Contígua

- As três principais estratégias:
  - ***First-fit***:
    - o primeiro segmento livre com tamanho suficiente para alocar o arquivo é selecionado. A busca na lista é seqüencial, sendo interrompida tão logo se encontre um segmento adequado.
  - ***Best-fit***
    - seleciona o menor segmento livre disponível com tamanho suficiente para armazenar o arquivo. A busca em toda a lista se faz necessária para a seleção do segmento, a não ser que a lista esteja ordenada por tamanho.
  - ***Worst-fit***:
    - o maior segmento é alocado e a busca por toda a lista se faz necessária, a menos que exista uma ordenação por tamanho.

Alocando um arquivo com 2 blocos



Alocação Contígua

- Independente da estratégia utilizada, a alocação apresenta um problema chamado **fragmentação de espaços livres**
  - o problema pode se tornar crítico quando um disco possuir blocos livres disponíveis, porém sem um segmento contíguo onde o arquivo possa ser alocado

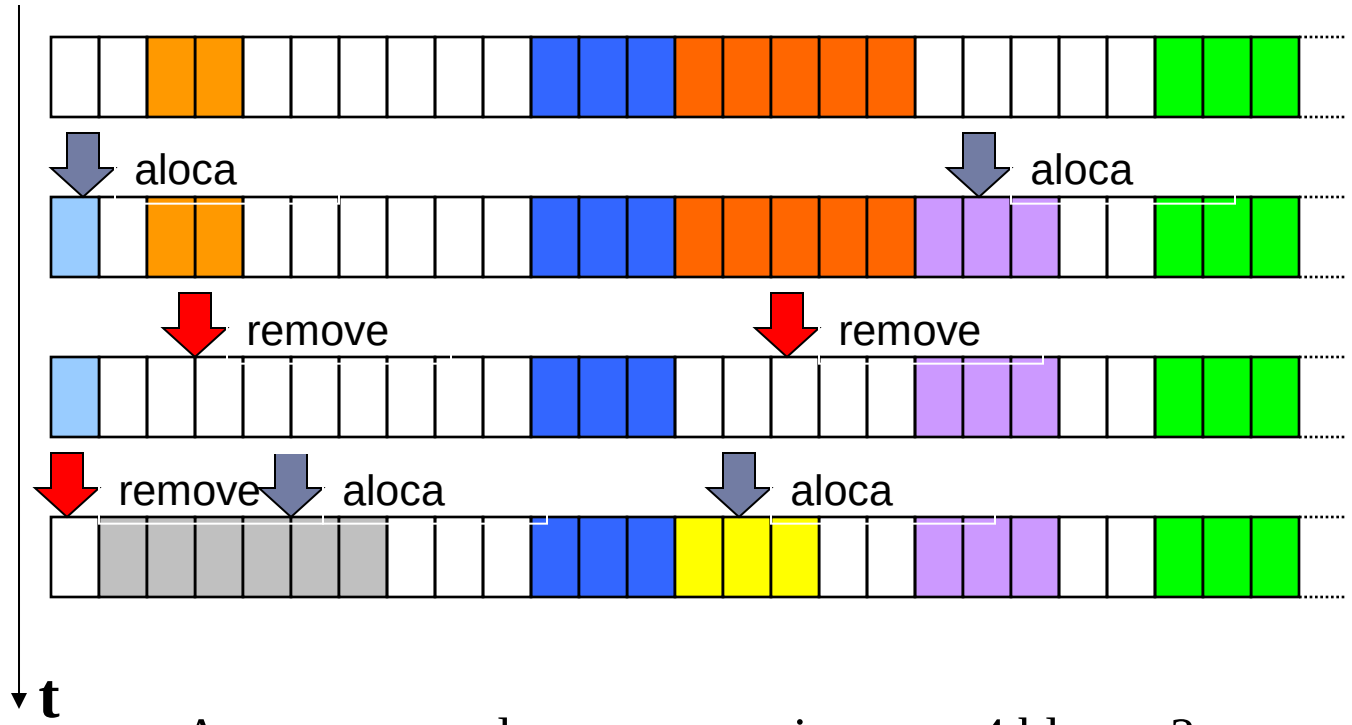
- Deve ser feita a desfragmentação periodicamente para reorganizar os arquivos no disco, a fim de que exista um único segmento de blocos livres
  - há um grande consumo de tempo neste processo e tem efeito temporário

Fragmentação externa

- Espaços vazios **entre** blocos de arquivos
- À medida que o sistema evolui:
  - arquivos são criados e removidos
  - mais espaços vazios aparecem
  - os espaços vazios ficam menores

➔ Alocar novos arquivos torna-se difícil!!!

Evolução da fragmentação

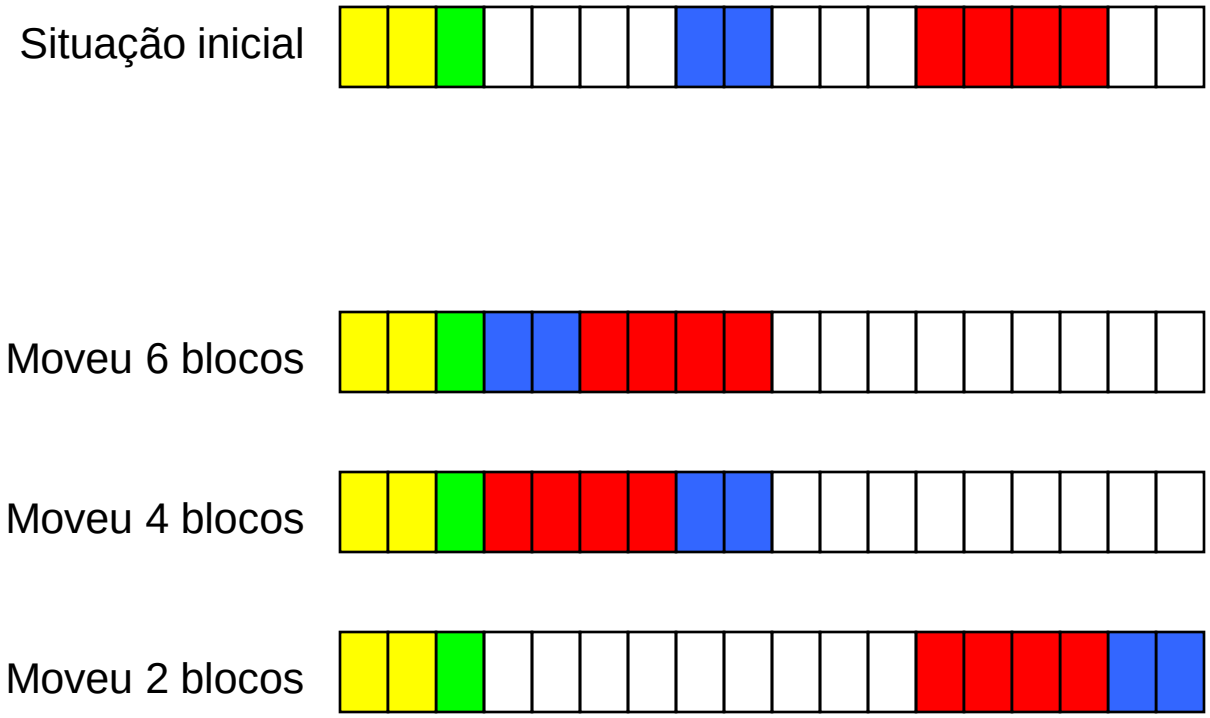


Agora, como alocar um arquivo com 4 blocos ?

Desfragmentação

- Mover arquivos para reagrupar os fragmentos em espaços maiores
- Visa permitir alocar arquivos maiores
- Deve ser feita periodicamente
- Uso de algoritmos para minimizar movimentação de arquivos (rapidez)

Estratégias de desfragmentação

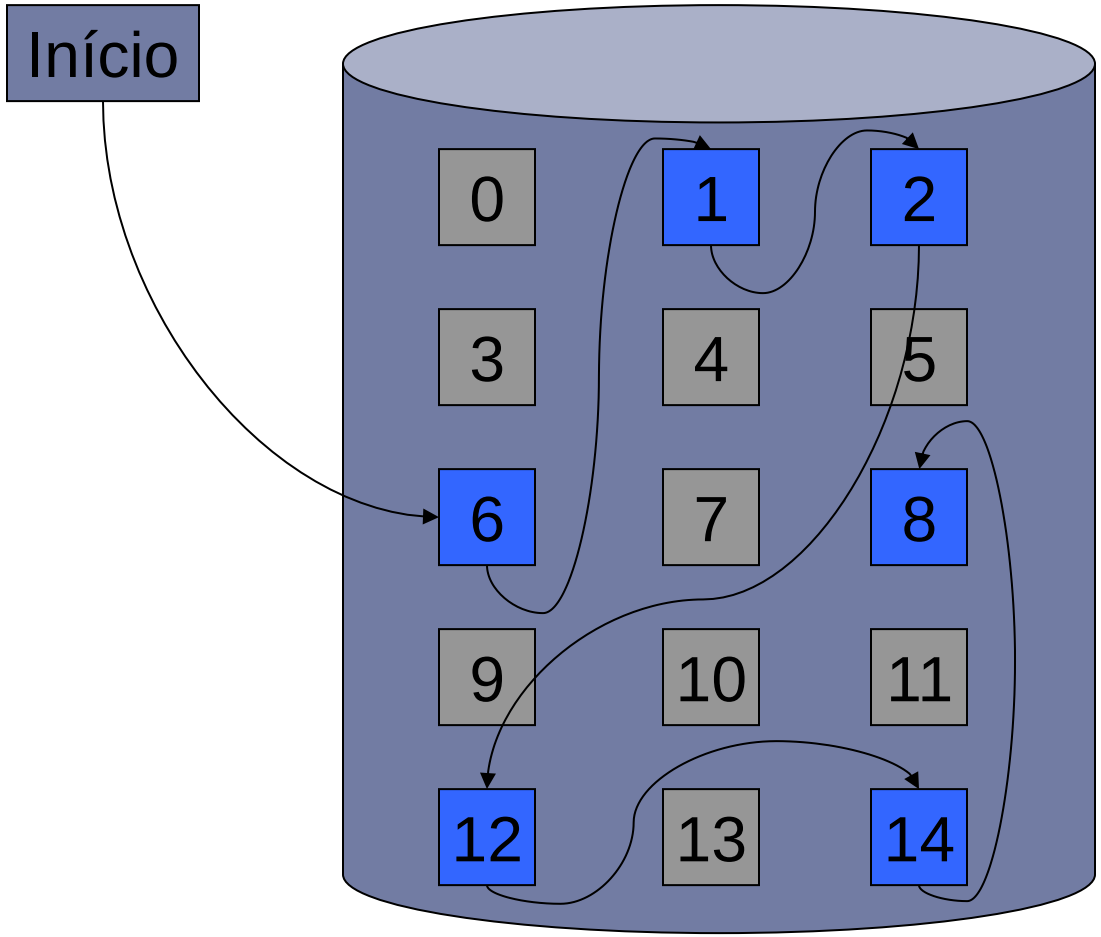


Alocação Encadeada de um Arquivo

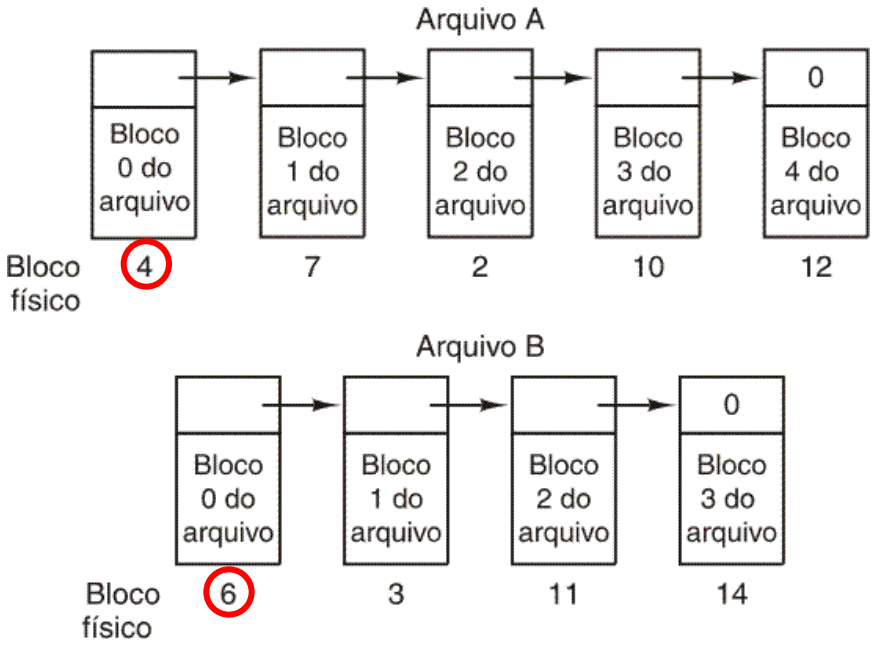
- O arquivo é organizado como um conjunto de blocos ligados no disco, independente de sua localização física e cada um deve possuir um ponteiro para o bloco seguinte

- Esta alocação só permite acesso seqüencial e desperdiça espaço nos blocos com armazenamento de ponteiros

Alocação Encadeada



Alocação Encadeada



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

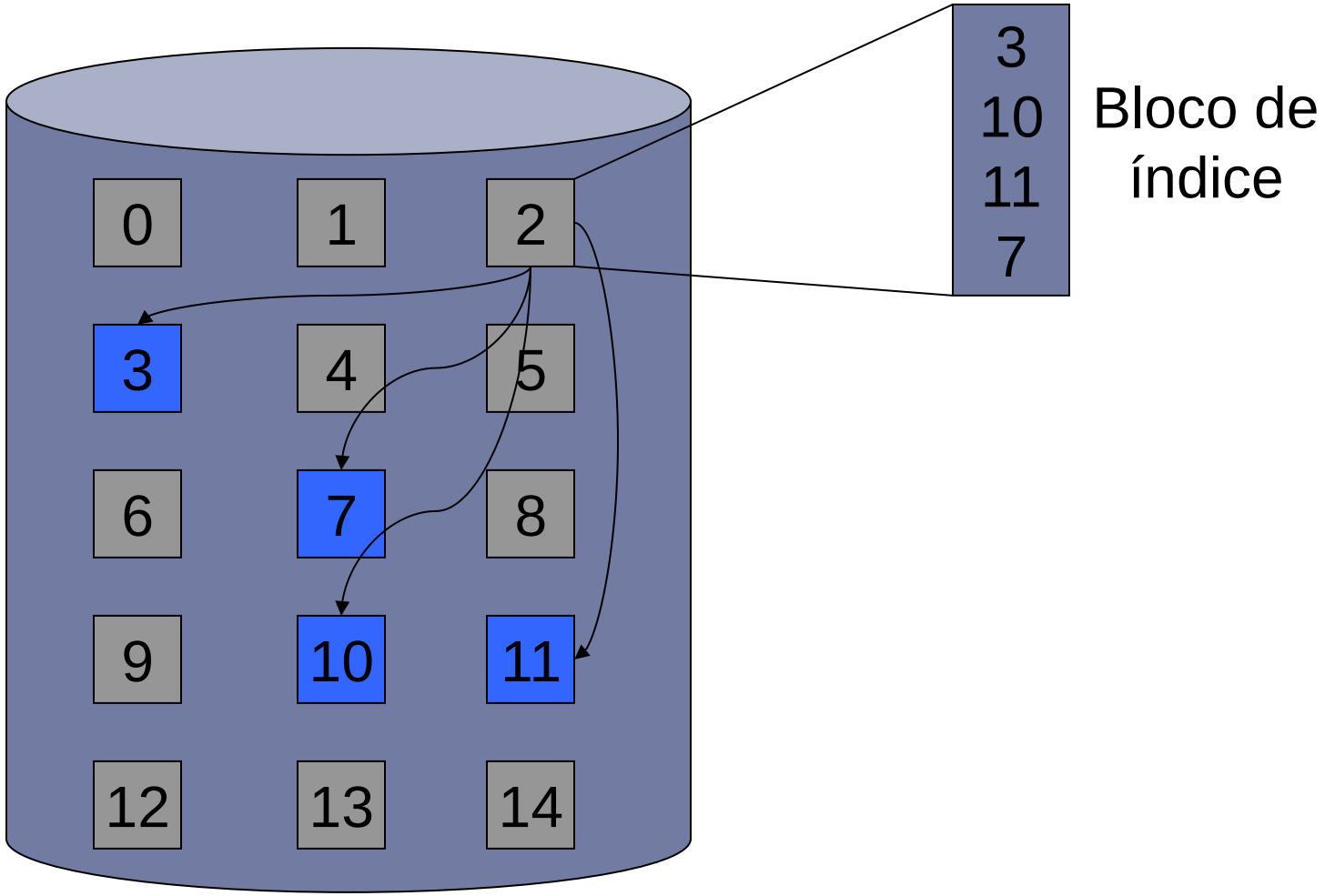
FAT
(File Allocation Table)
Tabela na memória

Alocação Indexada

- O princípio desta técnica é manter os ponteiros de todos os blocos de arquivos em uma única estrutura denominada *bloco de índice*

- *Qual a principal vantagem?*
  - Além de permitir o **acesso direto** aos blocos do arquivo, não utiliza informações de controle nos blocos de dados como existe na alocação encadeada
  - Não precisa ter a tabela completa em memória o tempo todo. Apenas, quando o arquivo é aberto o i-node é carregado.

Alocação Indexada

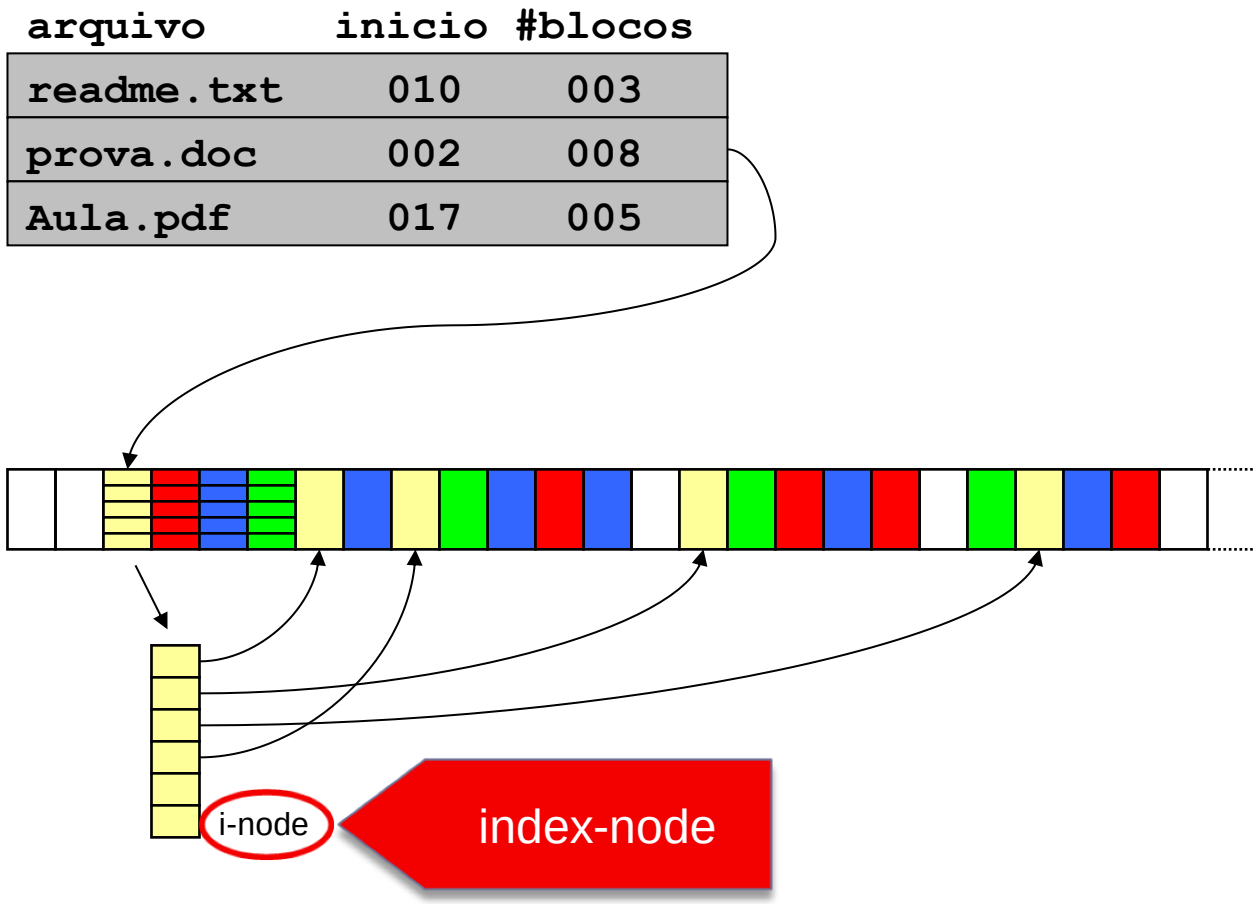


Alocação Indexada

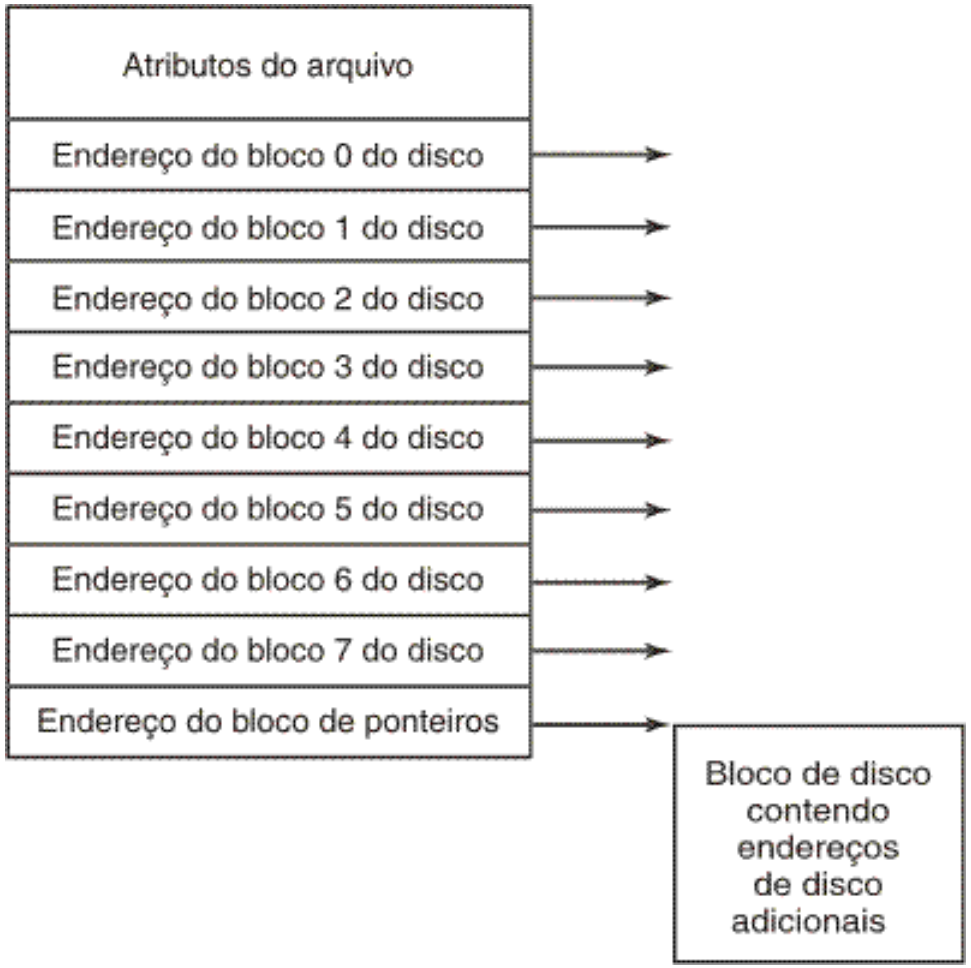
Associa a cada arquivo uma estrutura de dados conhecida como *I-Node*

Relaciona atributos e endereços em disco dos blocos de arquivos

Dado o i-node, é possível encontrar todos os blocos.



i-node

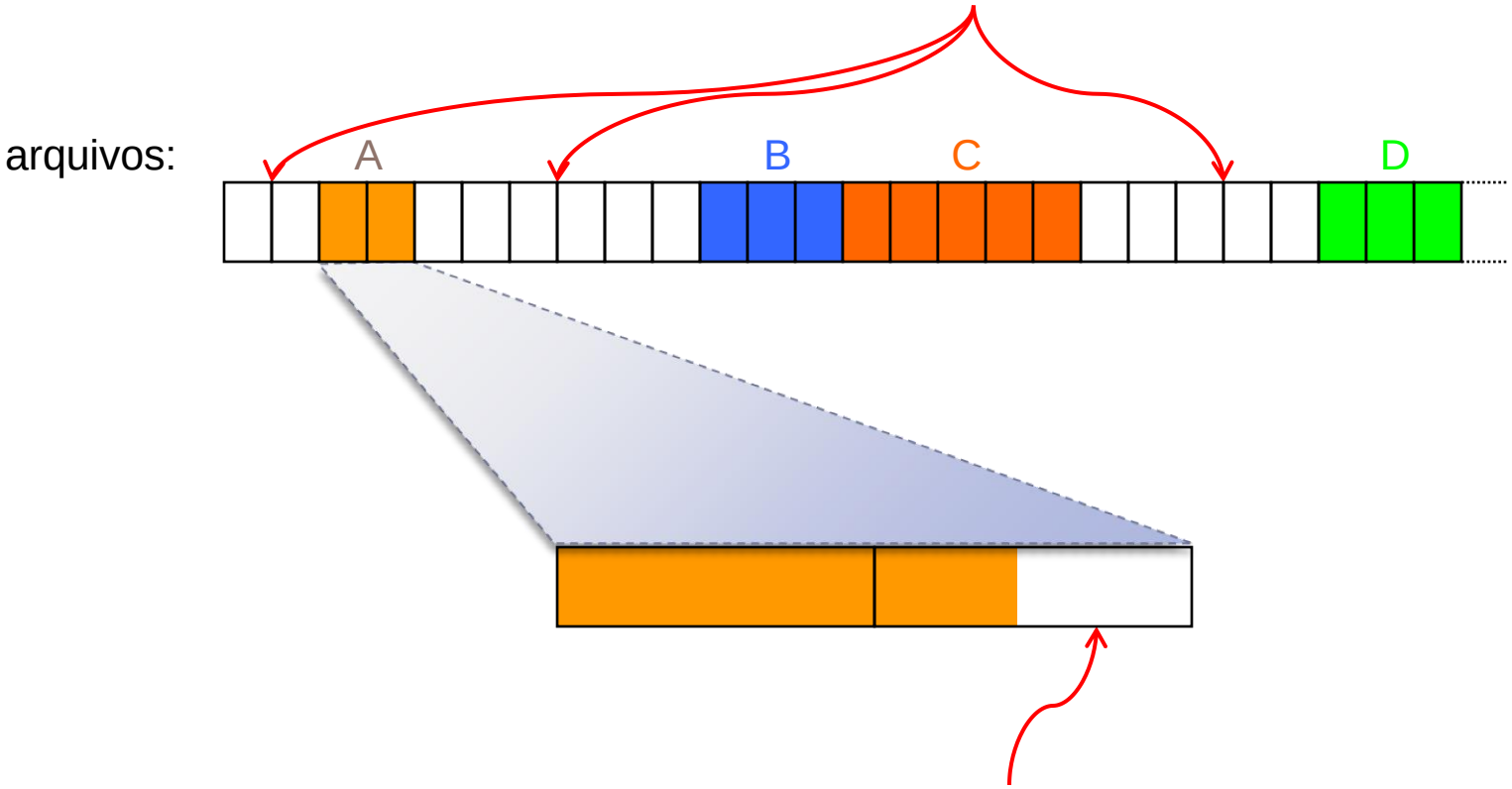

Atributos do arquivo
Endereço do bloco 0 do disco
Endereço do bloco 1 do disco
Endereço do bloco 2 do disco
Endereço do bloco 3 do disco
Endereço do bloco 4 do disco
Endereço do bloco 5 do disco
Endereço do bloco 6 do disco
Endereço do bloco 7 do disco
Endereço do bloco de ponteiros
Bloco de disco contendo endereços de disco adicionais

Fragmentação interna

- Arquivos são alocados em blocos:
  - Os blocos têm tamanho fixo
  - Entre 512 bytes e 8 Kbytes
  - Um bloco não pode ser alocado parcialmente
- Se usarmos blocos de 4096 bytes
  - um arquivo de 5700 bytes ocupará 2 blocos
  - 2492 bytes serão perdidos no último bloco
- Em média, perde-se 1/2 bloco por arquivo

E qual a diferença entre Fragmentação interna e externa?

Fragmentação externa: espaços vazios **entre** blocos de **arquivos**



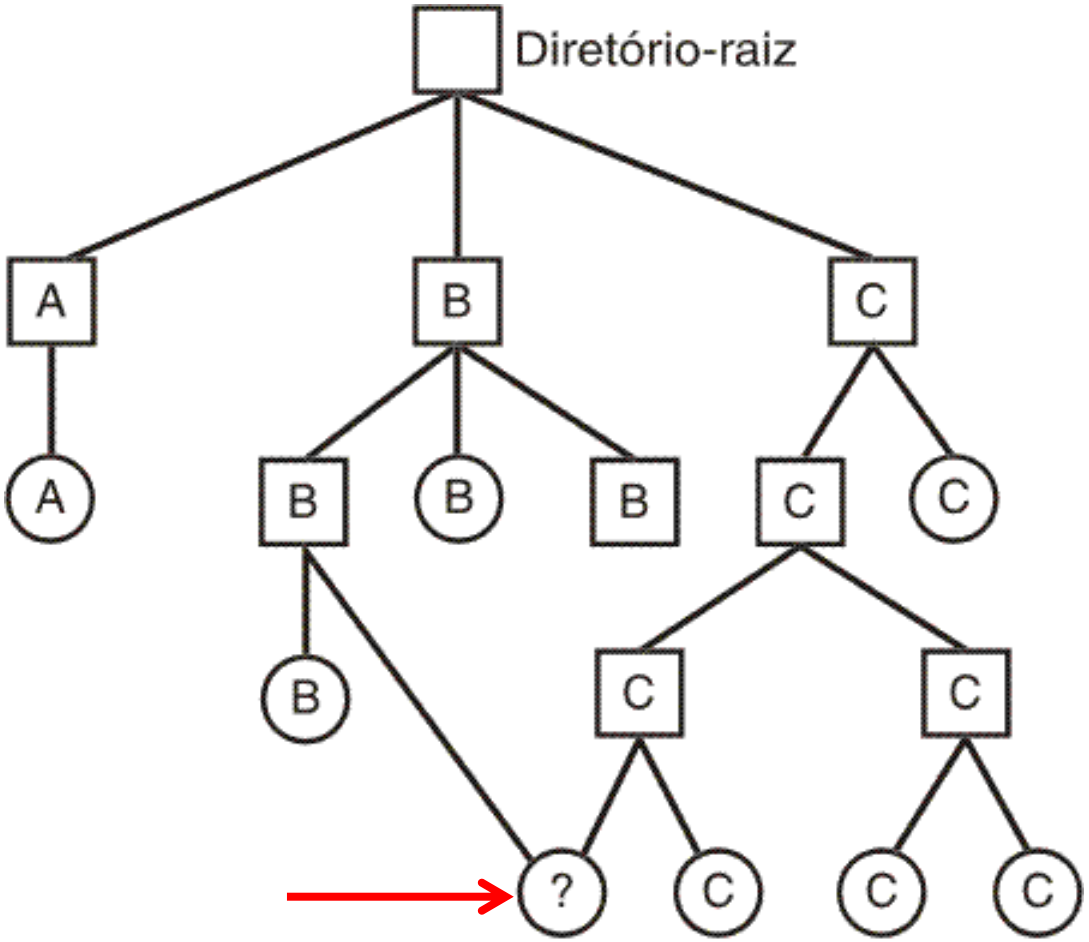
Fragmentação interna: uso incompleto do espaço do último bloco de **um arquivo**

Tamanho dos blocos

- A escolha do tamanho dos blocos é importante para a eficiência do sistema

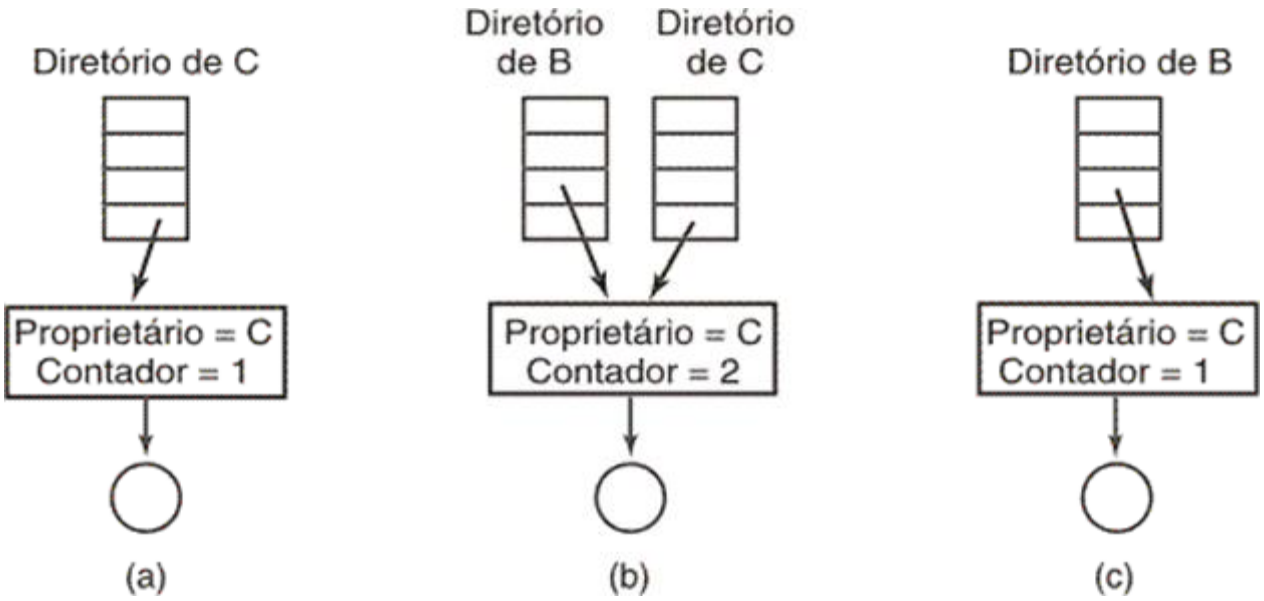
- Blocos pequenos:
  - menor perda por fragmentação interna
  - mais blocos por arquivo: maior custo de gerência
- Blocos grandes:
  - maior perda por fragmentação interna
  - menos blocos por arquivo: menor custo de gerência

Arquivos Compartilhados (1)



Arquivo compartilhado

Arquivos Compartilhados (2)



(a) Situação antes da ligação

(b) Depois de a ligação ser criada

(c) Depois de o proprietário (C) remover o arquivo (i-node deixado intacto para evitar erro, já que B não é o proprietário – e continua na conta de alocação de C…)

Gerenciamento do Espaço em Disco

Considerações relevantes:

- Tamanho do bloco: eficiência
- Monitoramento de blocos livres (ex. mapas de bits)
- Cotas de usuários

Proteção de Acesso

- Considerando que os meios de armazenamento são **compartilhados**, é necessário ter mecanismos de proteção par garantir a proteção de arquivos e diretórios

- Qualquer sistema de arquivos deve possuir mecanismos próprios para proteger o acesso às informações gravadas

- O tipo de acesso é mediante concessão ou não de acessos que podem ser realizados como:
  - **leitura** (read)
  - **gravação** (write)
  - **execução** (execute)
  - **eliminação** (delete)

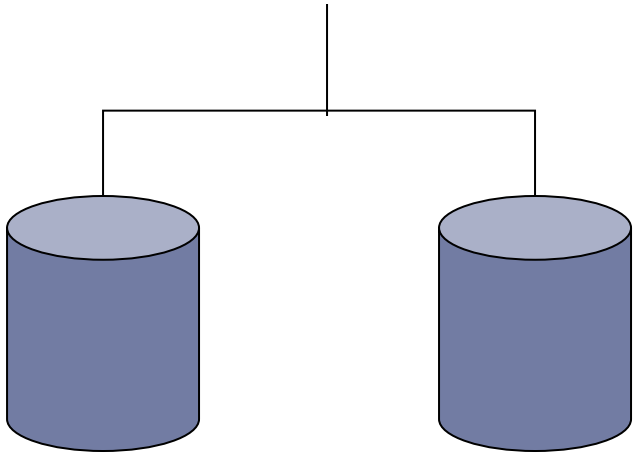
Grupos de Usuários

- Tem como princípio a associação de cada usuário do sistema a um grupo. Os usuários são organizados com o objetivo de compartilhar arquivos entre si

- Implementa três tipos de proteção: *owner* (dono), *group* (grupo) e *all* (todos)
  - na criação do arquivo é especificado quem e o tipo de acesso aos três níveis de proteção

- Em geral, somente o dono ou usuários privilegiados é que podem modificar a proteção dos arquivos

Proteção por Grupo de Usuários



dados.txt

Nível de proteção	Tipo de Acesso
Owner	Leitura Escrita Execução Eliminação
Group	Leitura
All	--

Proteção por Grupo de Usuários

Permissão	Binário	Decimal
---	000	0
--x	001	1
-w-	010	2
-wx	011	3
r--	100	4
r-x	101	5
rw-	110	6
rwx	111	7

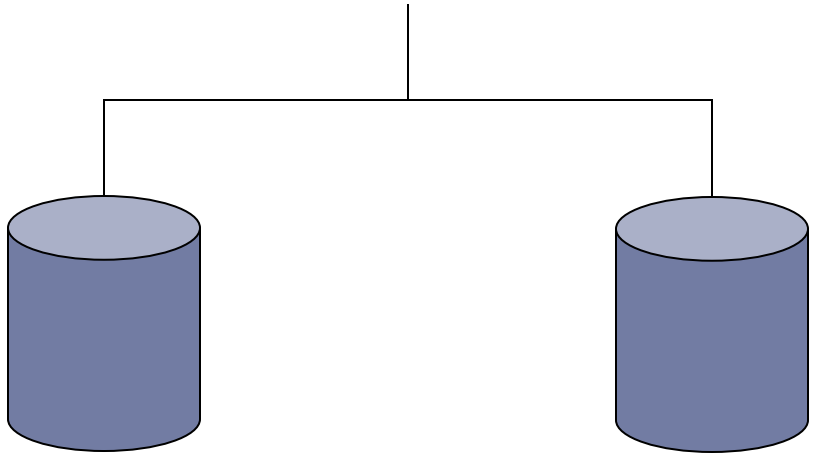
Lista de Controle de Acesso

- Access Control List – ACL consiste em uma lista associada a cada arquivo onde são especificados quais os usuários e os tipos de acesso permitidos
- O tamanho desta estrutura pode ser bastante extenso se um arquivo tiver seu acesso compartilhado por diversos usuários
- Existe um *overhead* adicional devido à pesquisa seqüencial que o sistema deverá realizar na lista sempre que solicitado
- É possível ter tanto a proteção por grupos de usuários quanto pela lista de acesso
  - maior flexibilidade ao mecanismo de proteção

Lista de Controle de Acesso

Usuário: cagf
Acesso: leitura + escrita

Usuário: ll3
Acesso: leitura



Usuário: cagf
Acesso: leitura + escrita + execução

Usuário: ll3
Acesso: eliminação

Implementação de Caches

- O acesso a disco é bastante lento comparado à memória principal e este é o fator para que as operações de E/S sejam consideradas um problema para o desempenho do sistema

- Com o objetivo de minimizar este problema, a maioria dos sistemas operacionais implementa a técnica de *buffer cache*, onde o sistema reserva uma área na memória para que se tornem disponíveis *caches* utilizados em operações de acesso a disco

- Quando uma operação é realizada, o sistema procura na *cache* a informação e, caso não encontre, busca no disco e depois atualiza a *buffer cache*

Implementação de Caches

- Como existe limite para o tamanho da *cache*, o sistema adota políticas de substituição como o FIFO (First in First out) ou a LRU (Least Recently Used)
- No caso de dados (blocos) permanecerem por um longo tempo na memória, a ocorrência de problemas de energia pode resultar na perda de tarefas já executadas e consideradas salvas em disco
- Existem duas maneiras de tratar este problema:

o sistema pode possuir uma rotina que executa, de tempos em tempos, atualizações em disco de todos os blocos modificados na *cache*

toda vez que um bloco da *cache* for modificado, realizar uma atualização no disco (*write-through caches*)

Sistemas Operacionais
Arquivos

Carlos Ferraz (cagf@cin.ufpe.br)
Jorge Cavalcanti Fonsêca (jcbf@cin.ufpe.br)