Aula 18

Sistema de Arquivos - Implementação

Prof. Sérgio Vale Aguiar Campos
scampos@dcc.ufmg.br

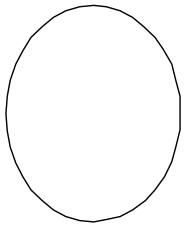
Visão Lógica X Implementação

- O que é um arquivo?

O usuário enxerga uma coleção de dados.

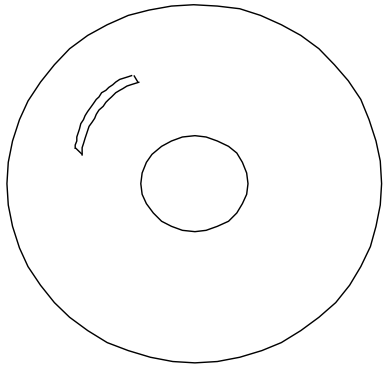
O sistema enxerga trilhas, setores e cilindros

Usuário:



corrupcao_gov_lula.txt

Sistema:

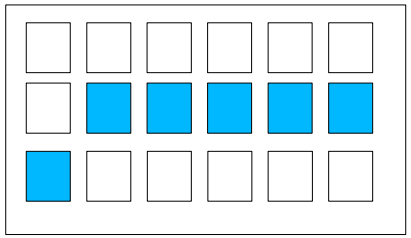


Alocação de Espaço em Disco

- Gerência de espaço em disco se parece com o quê?
  - Gerência de memória, claro.
  Só que aqui, otimizar o espaço utilizado é mais importante que otimizar o tempo de acesso.

- Três métodos básicos existem:
  - Alocação contígua
  - Alocação encadeada
  - Alocação indexada

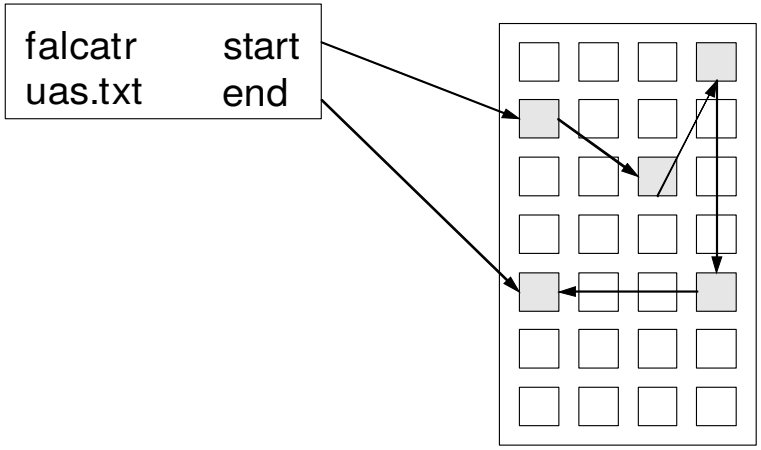
Alocação Contígua



- Vantagens:
  - fácil de implementar e acessar o arquivo
  - rápido acesso
- Desvantagens:
  - Fragmentação; como aumentar um arquivo?
    - Solução: Defrag/Compactação. Pode ser usado aqui porque o tempo não é tão crítico. Mas mesmo assim atrapalha.
- Não é muito usado.

Alocação Encadeada

- O arquivo passa a ser uma lista de blocos:



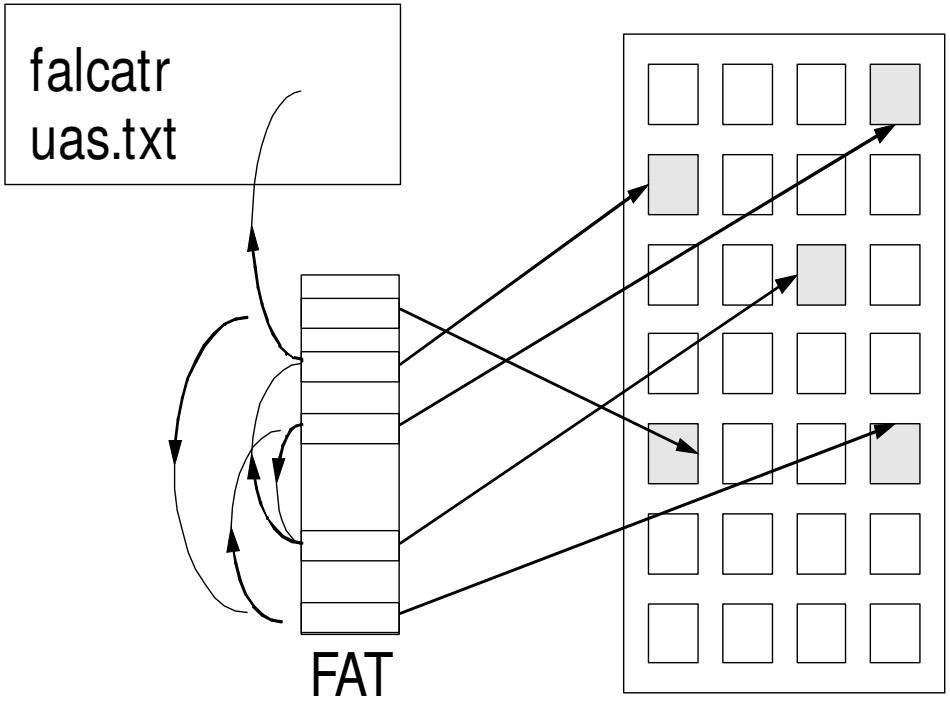
- Exemplo: Pick System – lista duplamente encadeada
  - vantagens?
  - desvantagens?

Alocação Encadeada

- Vantagens:
  - sem fragmentação
  - simples de aumentar um arquivo, assim como criar e apagar
- Desvantagens:
- - E os acessos não-seqüenciais?
- - Confiabilidade: e se um link se perder?
    - No sistema Pick isso se chamava **GFE** – Group Format Error
    - Se ao chegar no trabalho pela manhã houvesse um GFE na tela podia-se saber que o dia inteiro estava perdido em uma **caçada de links**...

Alocação Encadeada – FAT

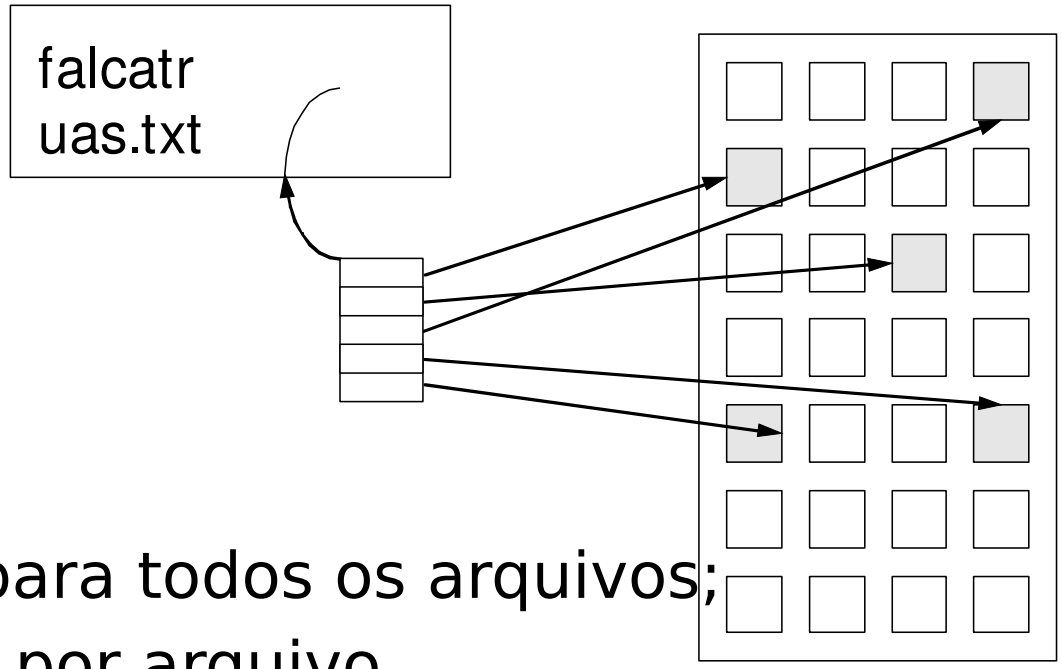
- E se guardar a lista encadeada **separada**?



- Vantagens:
  - Diminui a probabilidade de erros
- Desvantagens:
  - É preciso **cache** a FAT – eficiência
  - Usado no DOS, causa má impressão.

Alocação Indexada

- E se a lista encadeada for “compactada”?



- Diferenças:
  - uma FAT para todos os arquivos;
  - um índice por arquivo.
- Acesso é mais eficiente que FAT? Direto ou seqüencial?

Alocação Indexada

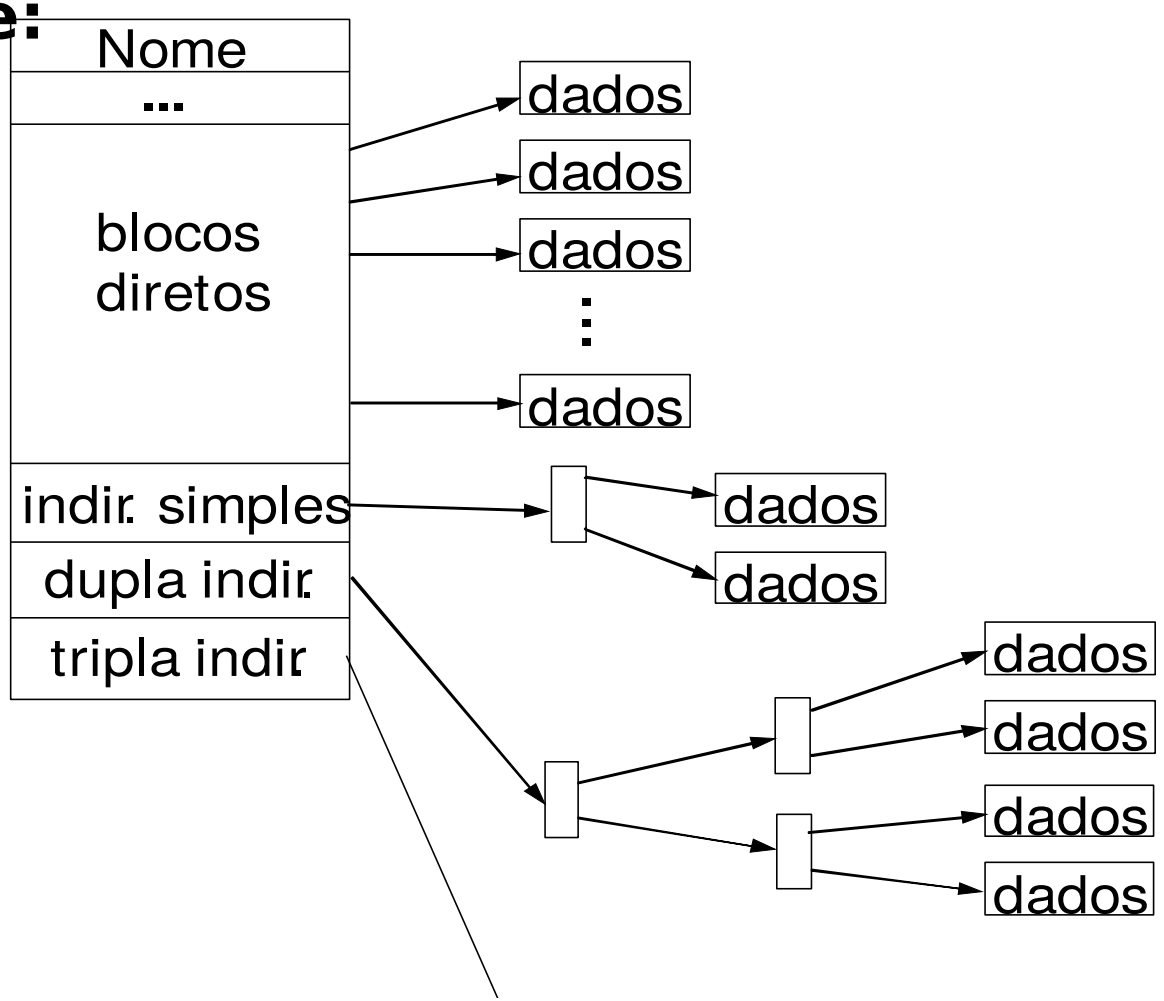
- Problema:
  - Usa-se ao menos um índice inteiro, mesmo que o arquivo use só um bloco.
- - Qual o tamanho ideal de índice?
    - - Muito grande, desperdício;
      - Muito pequeno, não cabem arquivos grandes.
      - Mas na verdade o tamanho ideal é um bloco de disco
- E aí???

Alocação Indexada

- Solução:
  - Usar uma lista encadeada de índices. Uma FAT de índices? IAT?
  - Índices multiníveis: usar índices de índices.
  - Solução BSD Unix (misto):
    - Os primeiros blocos são guardados em um índice direto
    - Se o arquivo não couber, cria-se um índice para índices – 1 nível de indireção
    - Se ainda não couber, índice para índice de índices – 2 níveis de indireção
    - Até 3 níveis existem

Alocação Indexada

- Unix **inode:**



Fragmentação

Fragmentação externa foi resolvida com alocação encadeada e indexação. Mas e fragmentação interna?

- Depende do tamanho do bloco de dados
- Por exemplo, com alocação encadeada o tamanho da FAT é inversamente proporcional ao tamanho do bloco
  - Pois a FAT **tem** que conter um valor para cada bloco no disco
  - No DOS, para discos maiores que 1G o bloco tem que ser de 32K
  - Com isso pode-se **perder até 25%** do espaço para fragmentação interna!
- Com alocação indexada isso não é problema: não é preciso manter um índice do disco todo.
  - Por exemplo, no BSD Unix o bloco é de 4K

Implementação de Diretórios

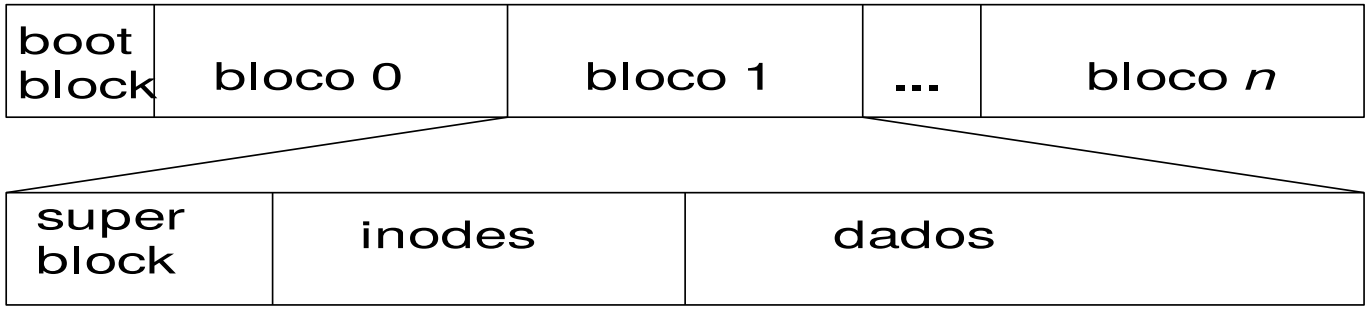
- Dado um nome de arquivo temos que achar seu "inode" (ou equivalente). Problema básico de busca em tabelas:
  - Lista linear
    - Fácil de implementar;
    - Difícil de usar
  - Tabela de hash
    - Mais eficiente.

Gerência de Espaço Livre

- Várias opções existem:
  - Bitmaps: Pode-se guardar um bitmap do disco no qual o bit 1 significa que o bloco correspondente está livre.
    - Problema: tem que ser “cached”, mas ocupa muito espaço na memória
  - Lista encadeada: Não ocupa espaço, mas é “meio” ineficiente para acessos não seqüenciais.
    - Só que o acesso à lista de blocos livres **é** sequencial.

Sistema de Arquivos de Linux/Unix

- Ponto inicial: um **superbloco** contendo: tamanho do bloco, número de blocos, lista de inodes etc.
  - replicado no disco, porque se o superbloco não puder ser lido, o sistema todo fica inacessível
  - usa um conjunto de blocos onde cada grupo de blocos é um mini sistema de arquivos:



- Motivo: tentar manter descrição e dados próximos

Sistema de Arquivos de Linux/Unix

- No Unix, inodes são pré-alocados e espalhados no disco:
  - aumenta o desempenho

- No Linux, os grupos de blocos fazem um papel semelhante

Sistema de Arquivos de Linux/Unix

- Diretórios no Linux:
  - Lista encadeada de nomes/inodes: