Sistemas de Arquivos - Interface

MCTA026-13 - Sistemas Operacionais

Emilio Francesquini
e.francesquini@ufabc.edu.br

2019.Q1

Centro de Matemática, Computação e Cognição
Universidade Federal do ABC

UFABC

Disclaimer

UFABC

- Estes slides foram preparados para o curso de **Sistemas Operacionais na UFABC**.
- Este material pode ser usado livremente desde que sejam mantidos, além deste aviso, os créditos aos autores e instituições.
- Este material foi baseado nas ilustrações e texto preparados por Silberschatz, Galvin e Gagne autores do livro texto [SGG] e detentores do copyright.

Interface do Sistema de Arquivos

Interface do Sistema de Arquivos

UFABC

- Conceito de arquivo
- Métodos de acesso
- Estrutura do disco e diretórios
- Proteção

Objetivos desta Seção

UFABC

- Explicar o funcionamento de sistemas de arquivos
- Descrever as interfaces de acesso a sistemas de arquivos
- Discutir *trade-offs* de implementação incluindo
  - Métodos de acesso
  - Compartilhamento de arquivos
  - Travas de arquivos
  - Estruturas de diretórios
- Explorar mecanismos de proteção dos sitemas de arquivos

O Arquivo

UFABC

- Pode ser visto como um espaço de endereçamento lógico contínuo
- Os arquivos podem ter tipos
  - Dados
    - numéricos
    - caracteres
    - binários
  - Programas
- Conteúdos (que podem ter muitos tipos) são definidos pelo seu criador
  - arquivo texto
  - arquivo de código fonte
  - arquivo executável
  - imagem
  - áudio
  - ...

Atributos de arquivos

UFABC

- Nome - A única informação gravada em um formato legível para humanos
- Identificador - Um identificador numérico e único que identifica o arquivo no sistema de arquivos
- Tipo - Utilizado por alguns sistemas de arquivos que precisam identificar o tipo do arquivo
- Localização - Ponteiro para a localização do arquivo no dispositivo (disco, SSD, ...)
- Tamanho - Tamanho atual do arquivo
- Proteção - Controla quem pode ler, escrever, executar, ...
- Data, hora e identificação de usuário - Armazena usuário criador/modificador; e datas e horas do último acesso, modificação, criação
  - Pode ser útil para proteção de dados, segurança e monitoramento de uso

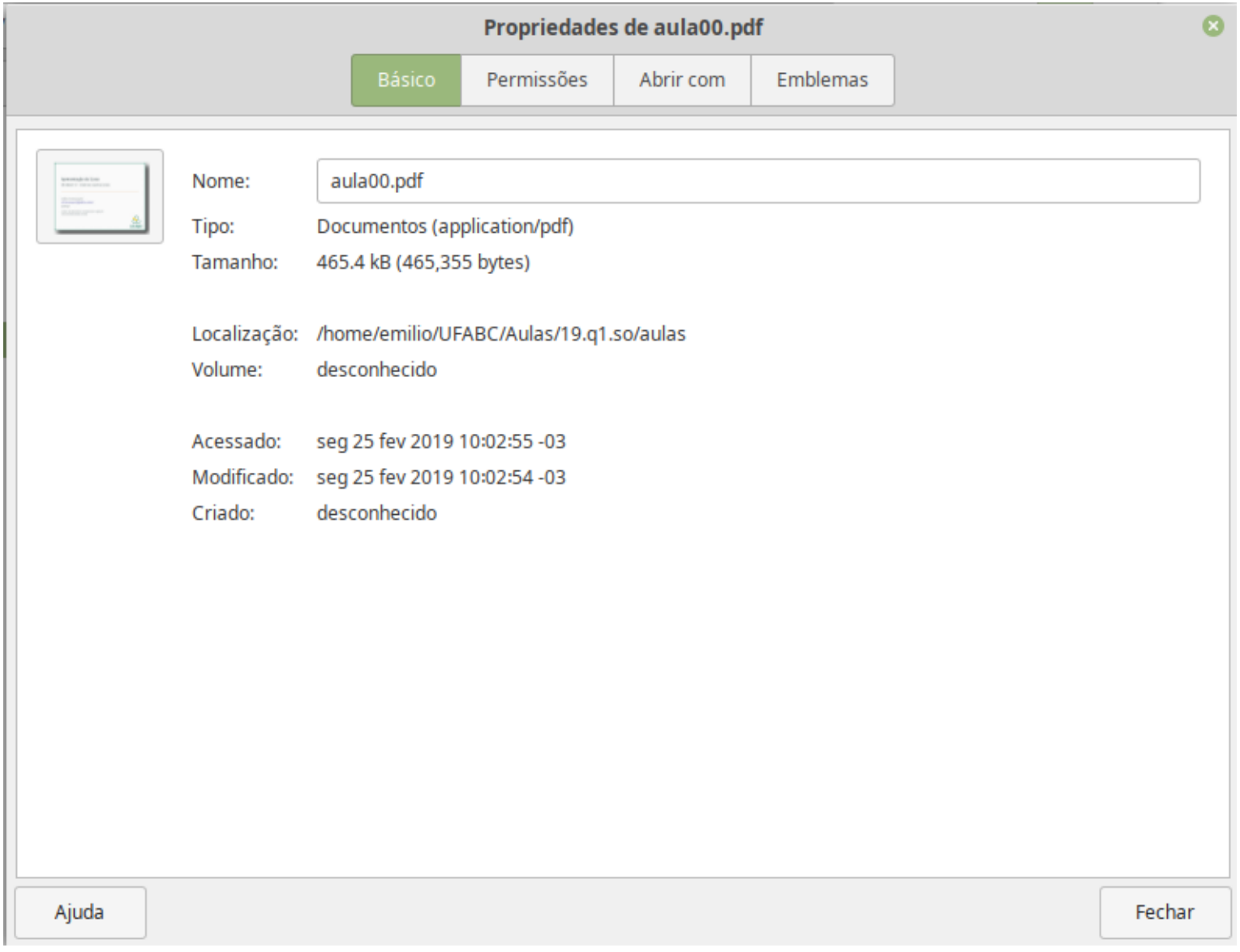
Atributos de arquivos

UFABC

- Alguns sistemas de arquivos incluem muitas variações como, por exemplo, *checksums*
- A informação é mantida em uma estrutura de diretórios (no dispositivo) que consiste de entradas chamadas de inodes para cada um dos arquivos presente no sistema

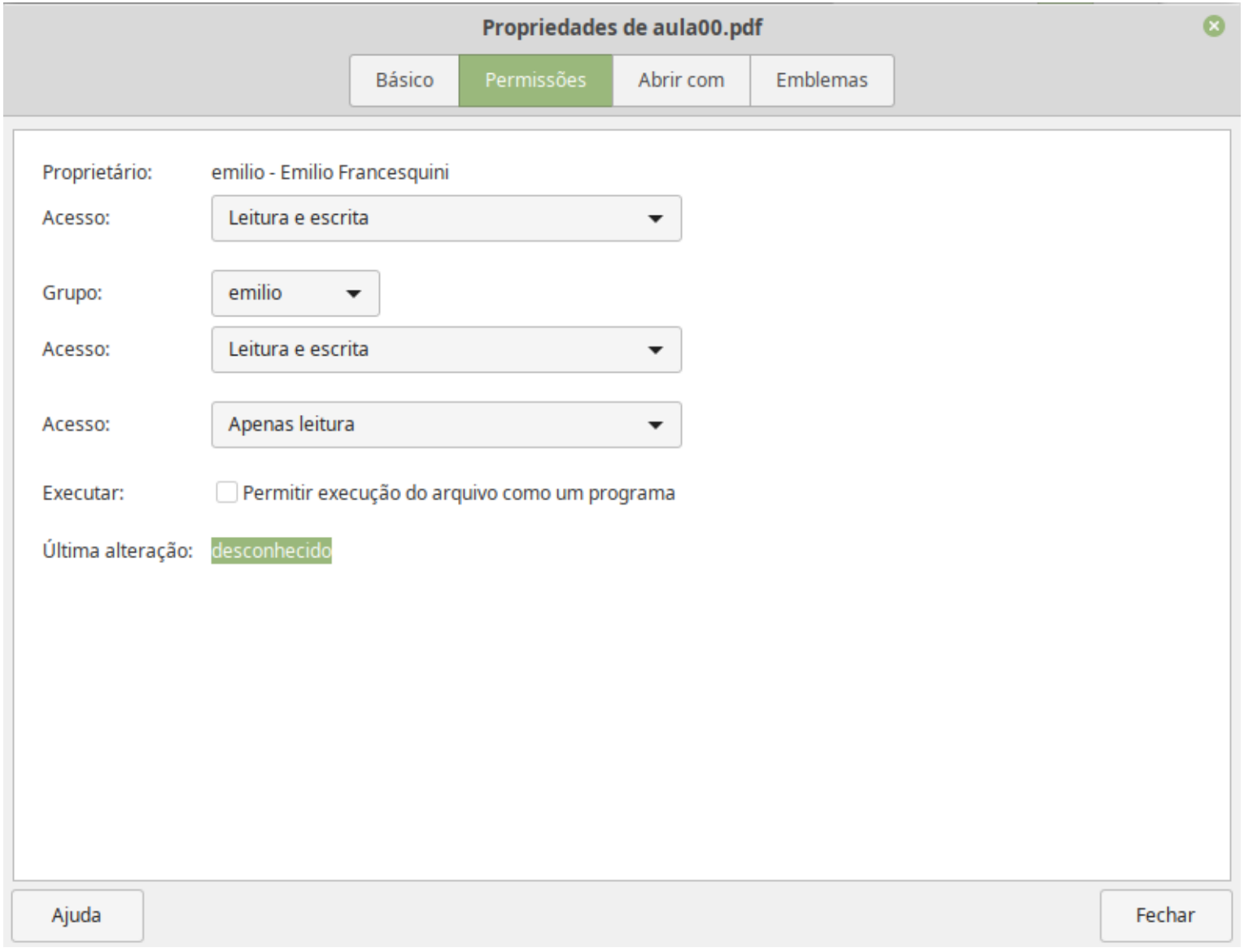
Janela com informações sobre arquivos (Cinnamon)

UFABC



Janela com informações sobre arquivos (Cinnamon)

UFABC



Operações com arquivos

UFABC

- **Criação**
- **Escrita** - na posição indicada pelo ponteiro de escrita
- **Leitura** - na posição indicada pelo ponteiro de leitura
- **Ajuste da localização dos ponteiros no arquivo** - seek
- **Exclusão**
- **Truncamento**
- **Abrir(F_i)** - Procura na estrutura de diretórios no disco pelo *inode* de F_i e move o conteúdo da entrada para a memória
- **Fechar(F_i)** - move o conteúdo do arquivo com a entrada *inode* de F_i da memória para a estrutura de diretórios no disco

Arquivos Abertos

UFABC

Diversos dados sobre arquivos abertos são mantidos pelo SO

- Tabela com os arquivos abertos - Mantém informação sobre todos os arquivos abertos
- Ponteiros por arquivo - Um ponteiro para cada última posição lida ou escrita e por processo que estiver com o arquivo aberto
- Contagem de arquivos abertos - Um contador com o número de vezes que um determinado arquivo foi aberto para permitir a limpeza da tabela de arquivos abertos quando o último utilizador fechá-lo
- Localização no disco do arquivo - Diversas operações de arquivos demandam modificações do dados contidos no arquivo. A informação para localizar o arquivo no disco é mantida na memória de forma que o sistema não precise relê-la para cada operação

Travas de arquivos

UFABC

- Alguns SOs permitem que o usuário trave o acesso ao arquivo
  - Evita leituras/escritas simultâneas
  - Shared lock - Parecido com uma trava de leitura (veremos mais adiante em detalhes) em que vários processos podem executar concorrentemente
  - Exclusive lock - Similar a uma trava de escrita
- O sistema operacional media o acesso ao arquivo de acordo com o estado da trava
- Política de acessos
  - Obrigatória/Mandatory - Acesso é negado dependendo das travas obtidas ou requisitadas
  - Informativa/Advisory - processos podem avaliar o status das travas e decidir como proceder

Tipos de arquivos - Nomes e extensões

UFABC

file type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, perl, asm	source code in various languages
batch	bat, sh	commands to the command interpreter
markup	xml, html, tex	textual data, documents
word processor	xml, rtf, docx	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	gif, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	rar, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, mp3, mp4, avi	binary file containing audio or A/V information

Estruturas de arquivos

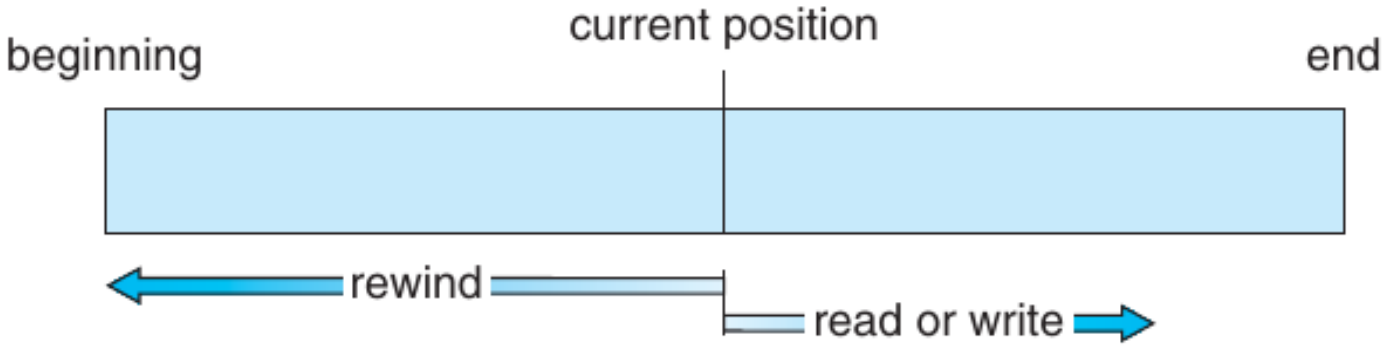
UFABC

- Nenhuma - sequencia de palavras, bytes
- Estrutura simples
  - Linhas
  - Colunas de largura fixa
  - Colunas de largura variável (CSV, TSV, ...)
- Estruturas complexas
  - Documento formatado
  - Arquivo executável
- Quem decide o tratamento?
  - SO
  - Programa

Métodos de acesso - Sequencial

UFABC

- Estrutura geral



- Operações
  - `read_next()` - Lê a próxima posição do arquivo e automaticamente avança o ponteiro
  - `write_next()` - Adiciona ao final do arquivo e avança para o fim (após o que acabou de ser escrito)
  - `reset` - Volta ao início do arquivo

Métodos de acesso - Direto

UFABC

- O arquivo é composto de diversos registros lógicos de tamanho fixo que permitem aos programas ler e escrever registros de maneira rápida em ordem arbitrária
- O arquivo é visto como uma sequência numerada de blocos (ou registros).
  - Por exemplo, uma aplicação pode ler o bloco 14, depois o bloco 53 e finalmente escrever no bloco 42
- Operações
  - `read(n)` - Leitura do bloco `n`
  - `write(n) - Escrita do bloco ~n`
- Números de blocos relativos permitem o SO decidir onde o arquivo deve ser armazenado
  - Mais sobre isso no capítulo 12

`cp` é um ponteiro para o próximo bloco a ser lido/escrito

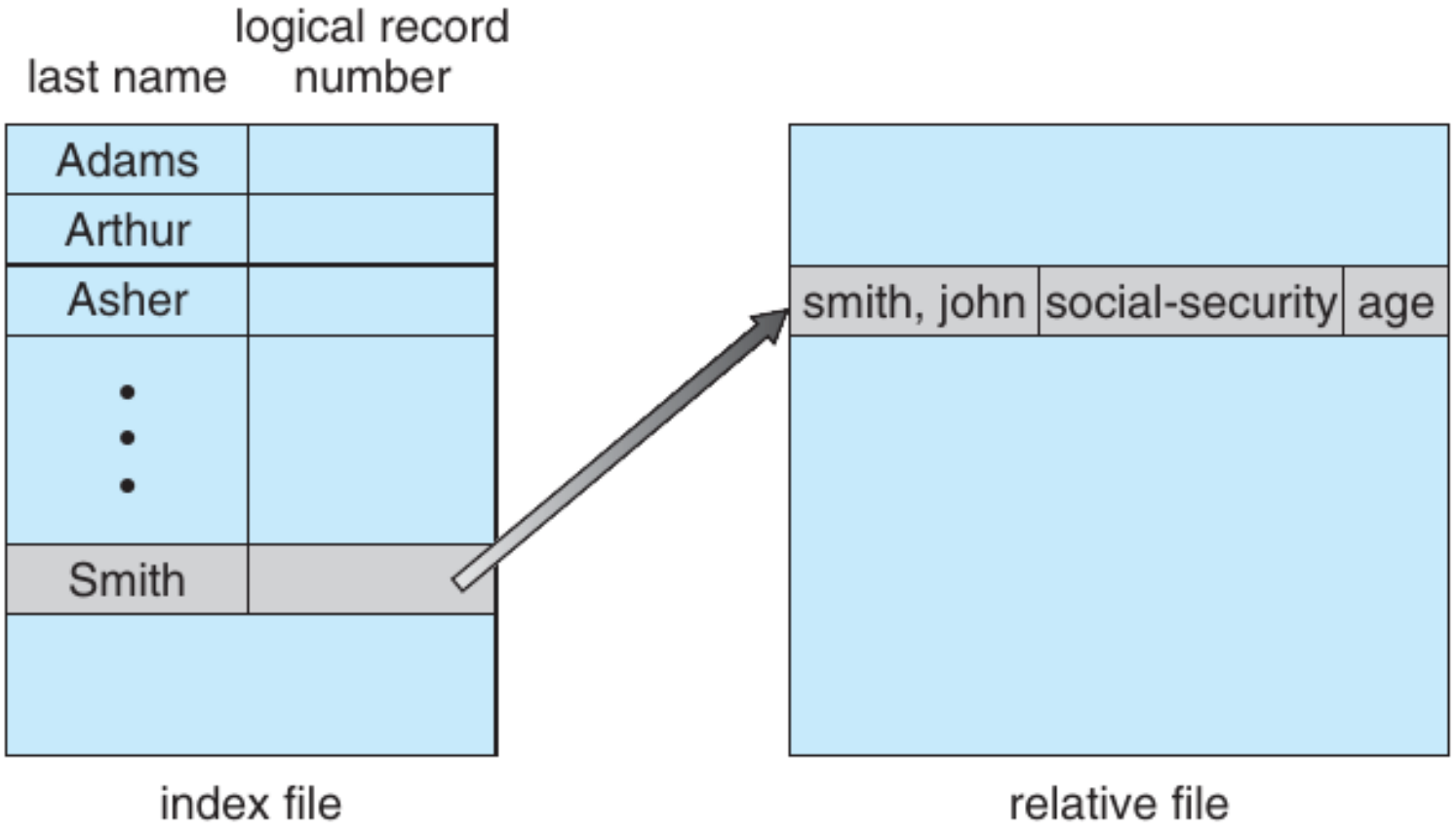
sequential access	implementation for direct access
reset	cp = 0;
read_next	read cp ; cp = cp + 1;
write_next	write cp; cp = cp + 1;

Outros tipos de acesso

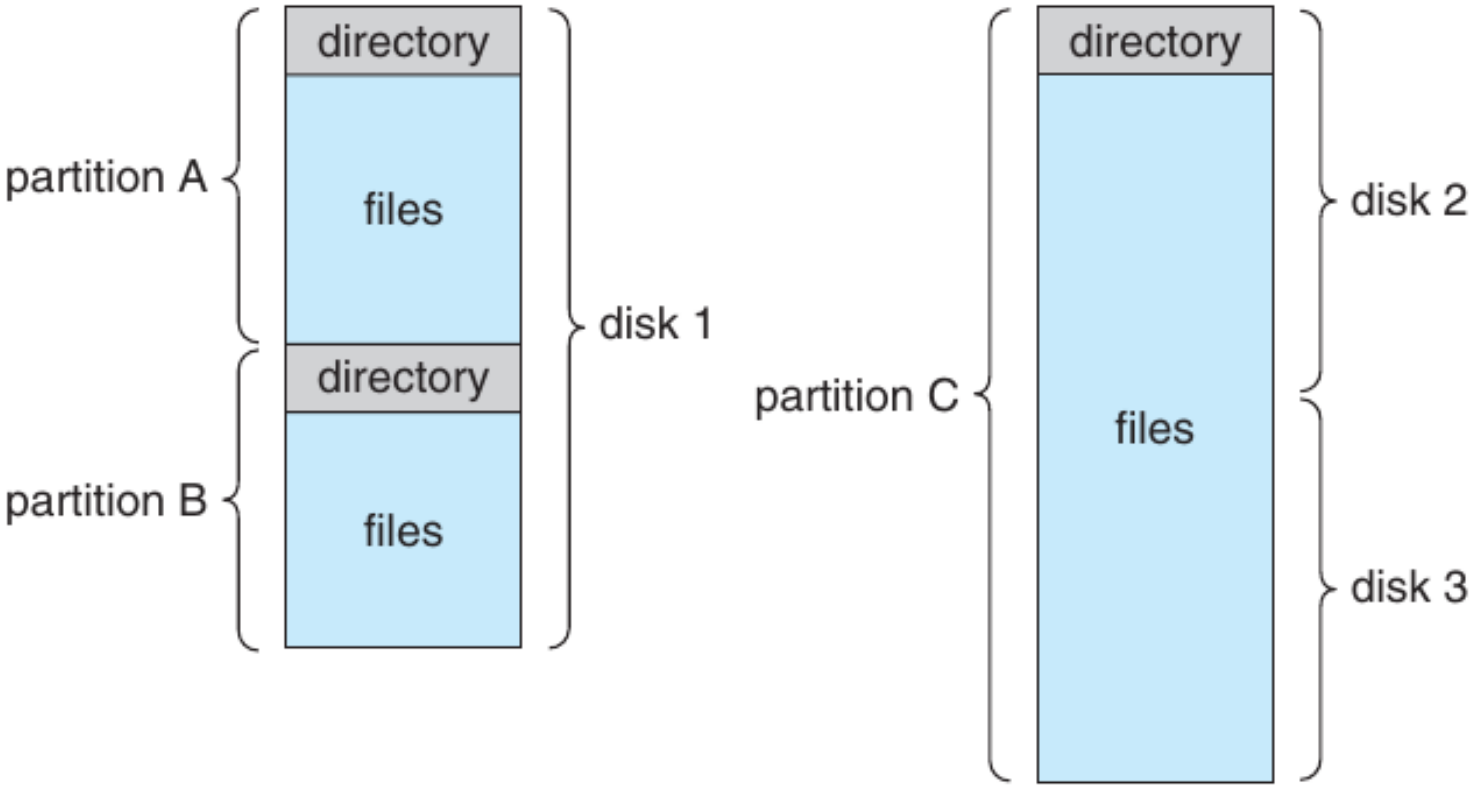
UFABC

- Se necessário podem ser construídos utilizando os métodos base apresentados
- Em geral envolvem a criação de um arquivo de índice
  - O índice é normalmente mantido na memória por motivos de eficiência
  - Se o índice for grande demais, mantém-se um índice do índice na memória
- *IBM indexed sequential-access method* (ISAM)
  - Pequeno índice geral que aponta para blocos em disco com índices secundários
  - Arquivos são mantidos em ordem baseada em uma chave definida
  - Feito completamente pelo SO
- O SO VMS provê um índice de arquivos relativos, por exemplo

Exemplo de índice de arquivos relativos



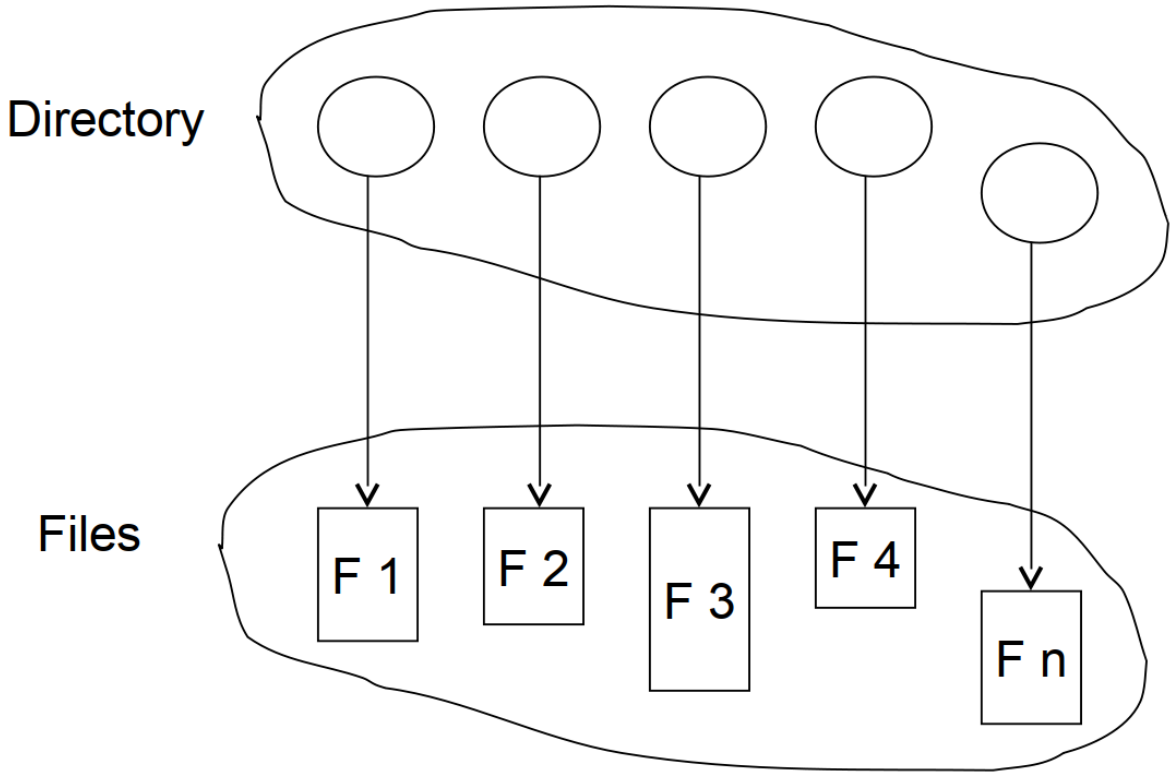
Estrutura de diretório



Estrutura de diretório

UFABC

Uma coleção de nós contendo informações sobre os arquivos



Tanto a estrutura de diretórios quanto os arquivos são armazenadas no disco

Operações em um Diretório

UFABC

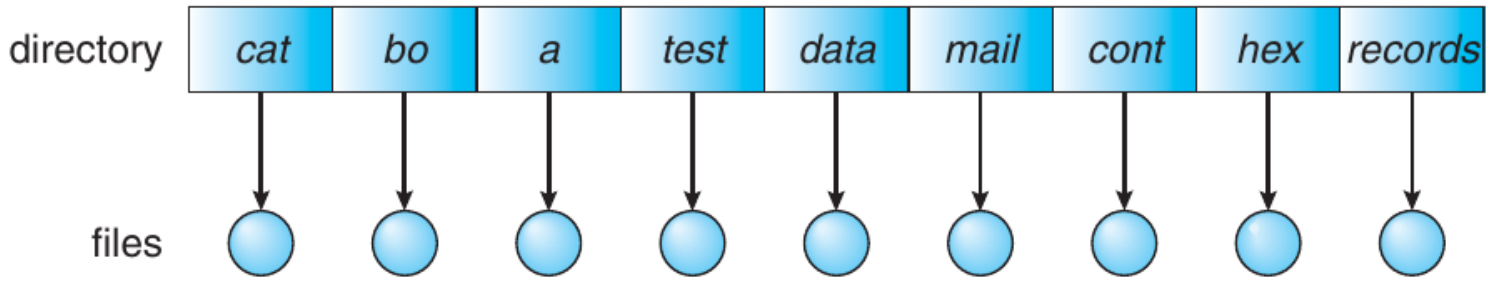
- Procurar um arquivo
- Criar um arquivo
- Apagar um arquivo
- Listar o conteúdo de um diretório
- Renomear um arquivo
- Percorrer o sistema de arquivos

Organização de Diretório

UFABC

- O diretório é organizado logicamente para obter
  - **Eficiência** - localizar um aquivo rapidamente
  - **Nomes** - sistema conveniente aos usuários
    - Dois usuários podem ter um nome igual para seus arquivos
    - O mesmo arquivo pode ter vários nomes diferentes
  - **Agrupamento** - agrupamento lógico de vários arquivos com propriedades semelhantes (ex. todos os programas Java, todos os jogos, ...)

Diretório de um Único Nível

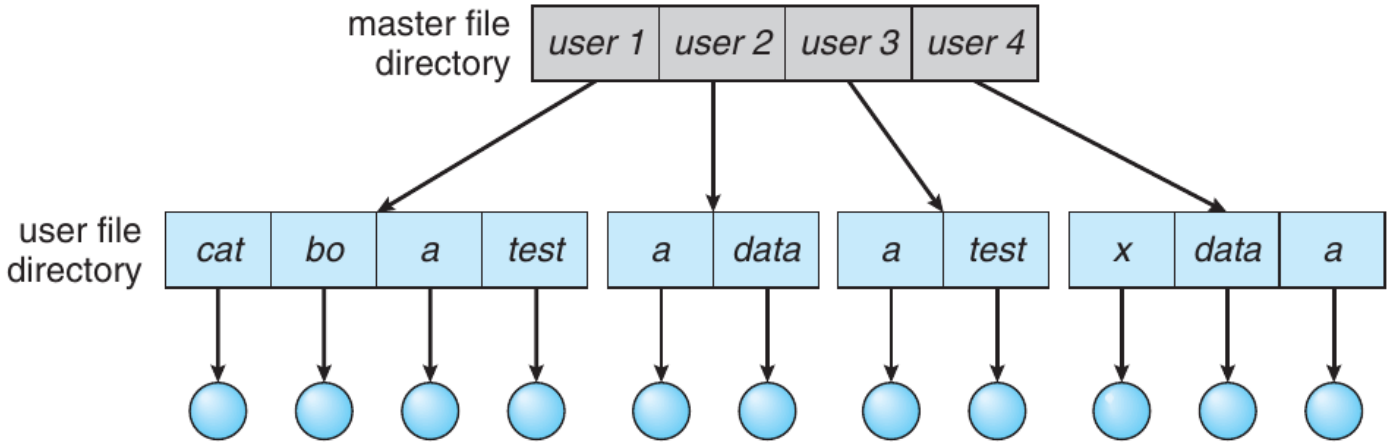


- Problema de nomes
- Problema de agrupamento

Diretório de Dois Níveis

UFABC

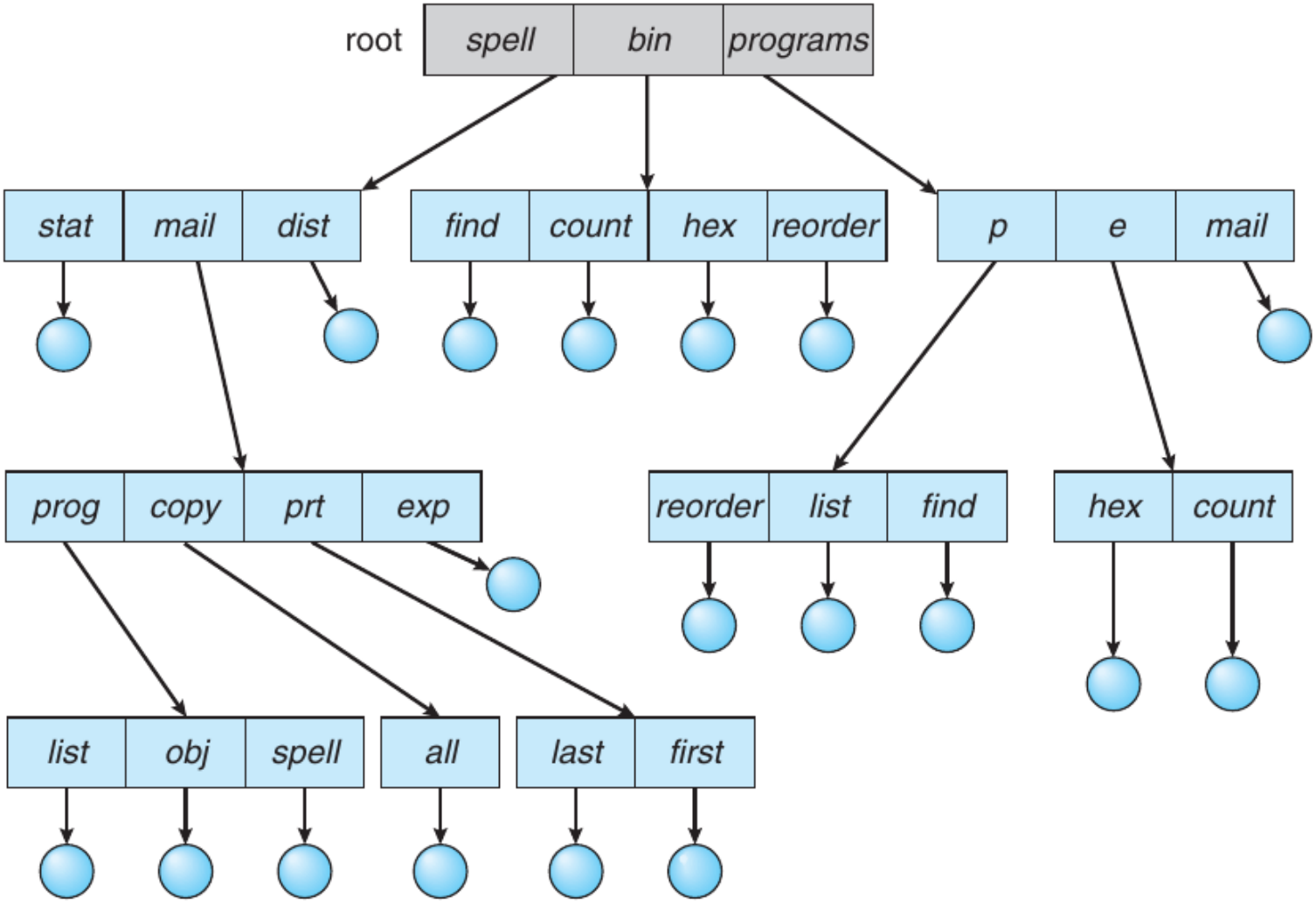
- Um diretório separado para cada usuário



- Nome é um caminho
- Permite que dois usuários tenham arquivos com os mesmos nomes
- Busca eficiente
- Não permite agrupamentos

Diretório Baseado em Árvores

UFABC



Diretório Baseado em Árvores

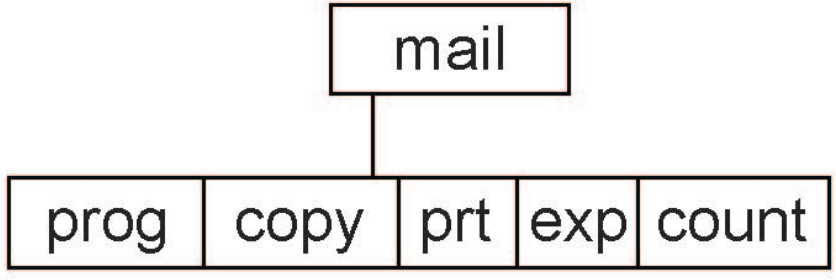
UFABC

- Busca eficiente
- Permite agrupamentos
- Diretório atual/Diretório de trabalho
  - `cd /home/emilio/provas`
  - `cat prova1.txt`

Diretório Baseado em Árvores

UFABC

- Caminho absoluto ou relativo
- A criação/exclusão/abertura de arquivos é feita no diretório de trabalho
- Exemplos
  - Apagando um arquivo
    - `rm <nome-do-arquivo>`
  - Criando um diretório
    - `mkdir <nome-do-diretório>`
  - Suponha que o diretório de trabalho atual é "`/mail`". Então o comando:
    - `mkdir count`

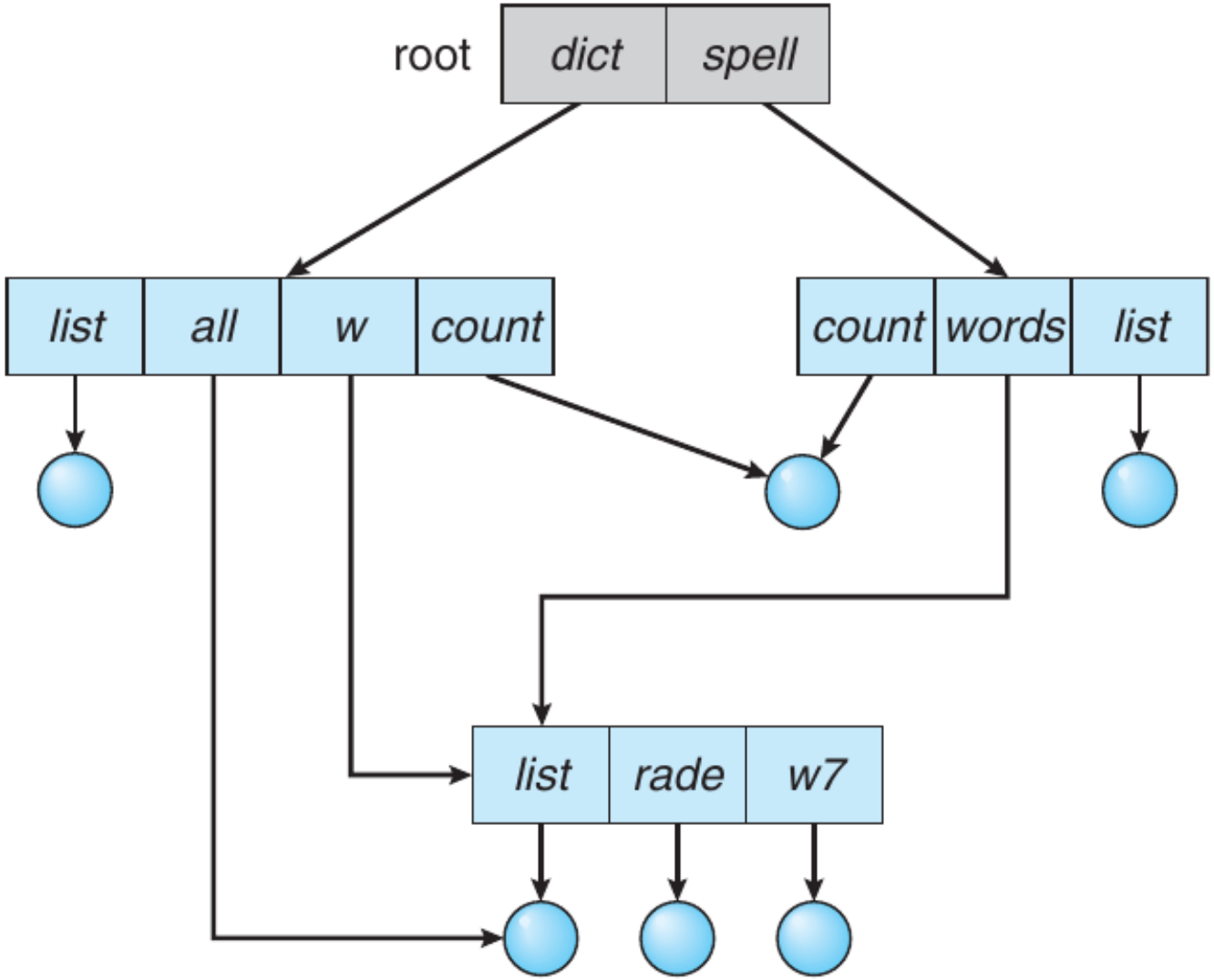


    - Apagar o diretório "`mail`"→ apagar a subárvore enraizada por "`mail`"

Diretório Baseado em Grafos Acíclicos

UFABC

Permitem o uso de subdiretórios compartilhados



Diretório Baseado em Grafos Acíclicos

UFABC

- Permite que arquivos e diretórios tenham dois nomes (caminhos)
  - Aliasing
  - Na realidade apenas um arquivo existe. Logo qualquer mudança feita é refletida para usuários utilizando qualquer um dos nomes
- Como arquivos e diretórios compartilhados são implementados?
  - Pode-se duplicar a informação do *inode* do arquivo em questão em todos os diretórios que compartilham o arquivo de maneira que todos apontem para o mesmo local.
    - Desvantagem: caso a informação sobre o arquivo seja atualizada então é preciso atualizar todas as réplicas
  - Criar um novo tipo de entrada no diretório
    - Link - nome alternativo (ponteiro) a um arquivo (ou diretório) já existente

Diretório Baseado em Grafos Acíclicos - Links

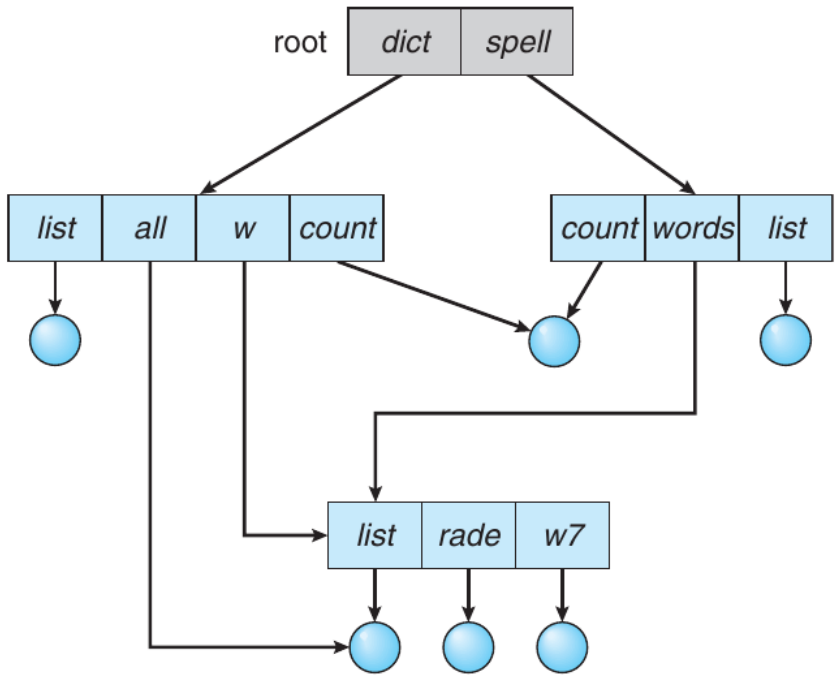
UFABC

- Um link pode ser implementado como sendo um caminho **absoluto** ou **relativo**
- Quando uma referência a um arquivo é feita e um link é encontrado, então o nome do arquivo real está incluído na informação do link
  - Um link pode apontar para outros links
- O processo de traduzir um link para o nome real do arquivo é chamado de resolução
- Links são normalmente identificáveis pelo formato nas entradas de diretório (ou por terem um tipo especial em sistemas que dão suporte a tipos) e são, efetivamente, ponteiros indiretos
- O SO (e aplicações) devem estar cientes da existência de links quando percorrem o sistema de arquivos para preservar a estrutura acíclica do sistema

Diretório Baseado em Grafos Acíclicos - Links

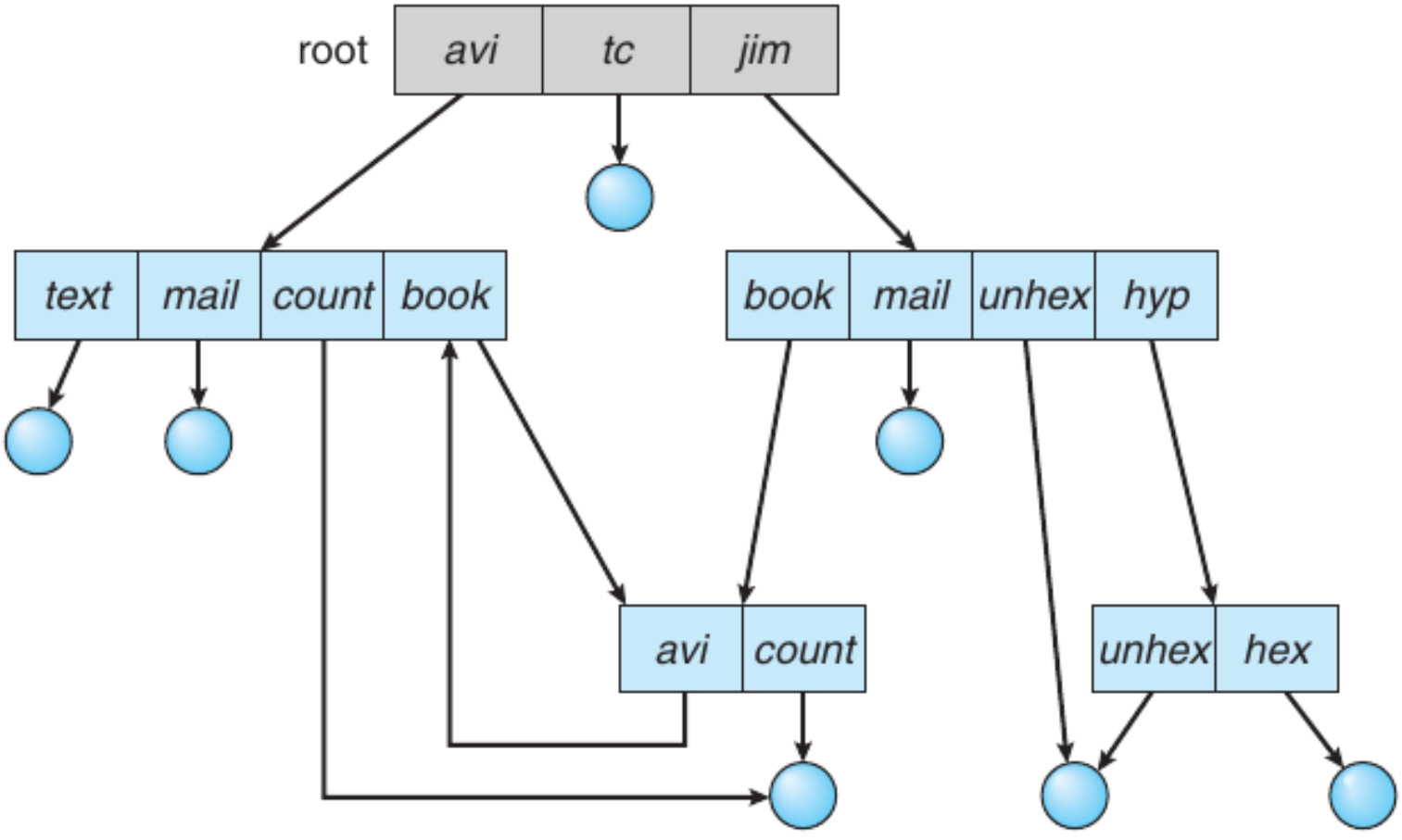
UFABC

- Se `dict/count` for apagado → ponteiro perdido (*dangling pointer*)
- Possíveis soluções
  - Backpointers - permitem apagar todos os ponteiros
    - Registros de tamanhos variados podem ser um problema
  - Backpointers com daisy-chain
  - Entrada com contadores



Diretório Baseado em Grafos Gerais

UFABC



UFABC

- Como garantir que não há ciclos?
  - Permita apenas links para arquivos, e não para subdiretórios
  - Garbage collection
  - Toda vez que um novo link for adicionado verificar se a inclusão cria um ciclo

UFABC

- O proprietário (*owner*) ou criador (*creator*) do arquivo deve poder controlar
  - O que pode ser feito...
  - ... e por quem
- Tipos de acessos
  - Leitura
  - Escrita
  - Execução
  - Concatenação (*append*)
  - Exclusão
  - Listar

UFABC

- Modo de acesso: leitura, escrita e execução
- Três classes de usuário no Linux:
  - *owner* (proprietário)
  - *group* (grupo)
  - *public///other* (público)
  - Para cada arquivo e subdiretório guarda-se 9 bits de proteção, 3 para o owner, 3 para o group e 3 para público

	Bits	RWX
Owner	7	111
Group	6	110
Public	1	001

UFABC

- Para criar um grupo é normalmente preciso pedir ao administrador do sistema
- Suponha que o grupo `G` acabou de ser criado e que deseja-se incluir o grupo ao diretório `game`
- `chgrp G game`

-Para ajustar os direitos de acesso do arquivo `game` use o comando `chmod`

- `chmod 761 game`
  - 7 owner
  - 6 group
  - 1 public

Exemplo: listando diretórios no Linux

UFABC

```
1  emilio@warp:~$ ls -l
2  total 29644
3  -rwxrwxr-x  1 emilio emilio     8512 Oct 14 21:36  a.out
4  -rw-rw-r--  1 emilio emilio       16 Dec  9 09:17  avaliacao_auto.txt
5  drwx------ 16 emilio emilio    53248 Jan 29 20:19  Downloads
6  -rw-rw-r--  1 emilio emilio 10169495 Apr  9  2018  ds5thedn.pdf
7  -rw-rw-r--  1 emilio emilio       10 Nov 26 17:33  lico
8  drwxr-xr-x  2 emilio emilio     4096 Dec 22  2017  Music
9  drwxrwxr-x  3 emilio emilio     4096 May  3  2018  NetBeansProjects
10 -rw-rw-r--  1 emilio emilio   115637 Dec  3 18:50  org-caldav-d54da94.el
11 -rw-rw-r--  1 emilio emilio        0 Dec  3 18:42  org-caldav-inbox.org
12 drwxrwxr-x  2 emilio emilio     4096 Apr 18  2018  personal
13 drwxr-xr-x  4 emilio emilio    12288 Feb 12 23:19  Pictures
14 drwxr-xr-x  2 emilio emilio     4096 Dec 22  2017  Public
15 drwxrwxr-x  3 emilio emilio     4096 Apr  3  2018  R
16 -rw-rw-r--  1 emilio emilio      578 Nov 26 17:14  teste.c
17 -rw-rw-r--  1 emilio emilio     3096 Dec  3 18:44  teste.el
18 drwxrwxr-x  4 emilio emilio     4096 Sep 11 17:10  texmf
19 drwxr-xr-x  3 emilio emilio     4096 Jan 29 20:23  Videos
20 dummy@warp:~$
```

Exemplo: Windows

UFABC