GERÊNCIA DO SISTEMA DE ARQUIVOS

1. INTRODUÇÃO

O armazenamento e a recuperação de informações é uma atividade essencial para qualquer tipo de aplicação. Um processo deve ser capaz de ler e gravar de forma permanente grande volume de dados em dispositivos como fitas e discos, além de poder compartilhá-los com outros processos. A maneira pela qual o sistema operacional estrutura e organiza estas informações é através da implementação de arquivos.

Os arquivos são gerenciados pelo sistema operacional de maneira a facilitar o acesso dos usuários ao seu conteúdo. A parte do sistema responsável por essa gerência é denominada *sistema de arquivos*. O sistema de arquivos é a parte mais visível de um sistema operacional, pois a manipulação de arquivos é uma atividade frequentemente realizada pelos usuários, devendo sempre ocorrer de maneira uniforme, independente dos diferentes dispositivos de armazenamento.

2. ARQUIVOS

Um *arquivo* é constituído por informações logicamente relacionadas. Estas informações podem representar instruções ou dados. Um arquivo executável, por exemplo, contém instruções compreendidas pelo processador, enquanto um arquivo de dados pode ser estruturado livremente como um arquivo texto ou de forma mais rígida como um banco de dados relacional. Na realidade, um arquivo é um conjunto de registros definidos pelo sistema de arquivos, tornando seu conceito abstrato e generalista. A partir dessa definição, o conteúdo do arquivo pode ser manipulado seguindo conceitos preestabelecidos.

Os arquivos são armazenados pelo sistema operacional em diferentes dispositivos físicos, como fitas magnéticas, discos magnéticos e discos ópticos. O tipo de dispositivo no qual o arquivo é armazenado deve ser isolado pelo sistema operacional, de forma que exista uma independência entre os arquivos a serem manipulados e o meio de armazenamento.

Um arquivo é identificado por um nome, composto por uma sequência de caracteres. Em alguns sistemas de arquivos é feita distinção entre caracteres alfabéticos maiúsculos e minúsculos. Regras como extensão máxima do nome e quais são os caracteres válidos também podem variar.

Em alguns sistemas operacionais, a identificação de um arquivo é composta por duas partes separadas com um ponto. A parte após o ponto é denominada extensão do

arquivo e tem como finalidade identificar o conteúdo do arquivo. Assim é possível convencionar que uma extensão TXT identifica um arquivo texto, enquanto EXE indica um arquivo executável. Na tabela abaixo, são apresentadas algumas extensões de arquivos:

Extensão	*Descrição*
ARQUIVO. BAS	*Arquivo fonte em BASIC*
ARQUIVO.COB	*Arquivo fonte em COBOL*
ARQUIVO.EXE	*Arquivo executável*
ARQUIVO.OBJ	*Arquivo objeto*
ARQUIVO.TXT	*Arquivo texto*
ARQUIVO.PAS	*Arquivo fonte em PASCAL*

2.1.Organização de Arquivos

A *organização de arquivos* consiste em como os seus dados estão internamente armazenados. A estrutura dos dados pode variar em função do tipo de informação contida no arquivo. Arquivos texto possuem propósitos completamente distintos de arquivos executáveis, consequentemente, estruturas diferentes podem adequar-se melhor a um tipo do que a outro.

No momento da criação de um arquivo, seu criador pode definir qual a organização adotada. Esta organização pode ser uma estrutura suportada pelo sistema operacional ou definida pela própria aplicação.

A forma mais simples de organização de arquivos é através de uma sequência não-estruturada de bytes (fig.1a). Neste tipo de organização, o sistema de arquivos não impõe nenhuma estrutura lógica para os dados. A aplicação deve definir toda a organização, estando livre para estabelecer seus próprios critérios. A grande vantagem deste modelo é a flexibilidade para criar diferentes estruturas de dados, porém todo o controle de acesso ao arquivo é de inteira responsabilidade da aplicação.

Alguns sistemas operacionais possuem diferentes organizações de arquivos. Neste caso, cada arquivo criado deve seguir um modelo suportado pelo sistema de arquivos. As organizações mais conhecidas e implementadas são a sequencial, relativa e indexada (fig. 1b). Nestes tipos de organização, podemos visualizar um arquivo como um conjunto de registros. Os registros podem ser classificados em registros de tamanho fixo, quando possuírem sempre o mesmo tamanho, ou registros de tamanho variável.

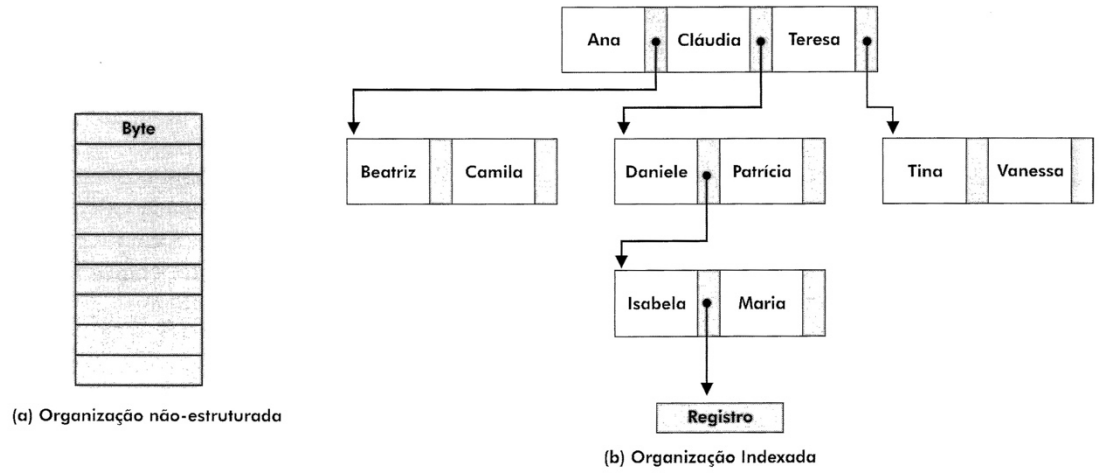


Figura 1: Organização de arquivos

2.2.Métodos de Acesso

Em função de como o arquivo está organizado, o sistema de arquivos pode recuperar registros de diferentes maneiras. Inicialmente, os primeiros sistemas operacionais só armazenavam arquivos em fitas magnéticas. Com isso, o acesso era restrito a leitura dos registros na ordem em que eram gravados, e a gravação de novos registros só era possível no final do arquivo. Este tipo de acesso, chamado de *acesso sequencial*, era próprio da fita magnética, que, como meio de armazenamento, possuía esta limitação.

Com o advento dos discos magnéticos, foi possível a introdução de métodos de acesso mais eficientes. O primeiro a surgir foi o *acesso direto*, que permite a leitura/gravação de um registro diretamente na sua posição. Este método é realizado através do número do registro, que é a sua posição relativa ao início do arquivo. No acesso direto não existe restrição à ordem em que os registros são lidos ou gravados, sendo sempre necessária a especificação do número do registro. É importante notar que o acesso direto somente é possível quando o arquivo é definido com registros de tamanho fixo.

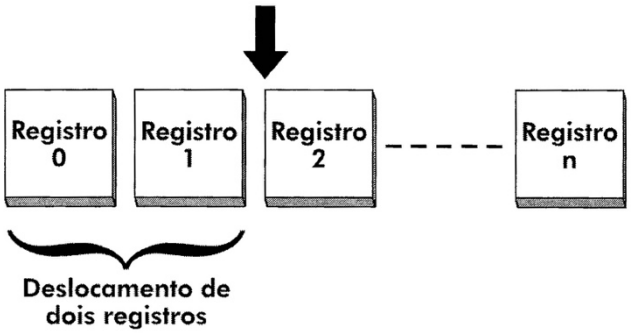


Figura 2: Acesso direto

O acesso direto pode ser combinado com o acesso sequencial. Com isso é possível acessar diretamente um registro qualquer de um arquivo e, a partir deste, acessar sequencialmente os demais.

Um método de acesso mais sofisticado, que tem como base o acesso direto, é o chamado *acesso indexado* ou *acesso por chave*. Para este acesso, o arquivo deve possuir uma área de índice onde existam ponteiros para os diversos registros. Sempre que a aplicação desejar acessar um registro, deverá ser especificada uma chave através da qual o sistema pesquisará na área de índice o ponteiro correspondente. A partir desta informação é realizado um acesso direto ao registro desejado.

2.3.Operações de Entrada/Saída

O sistema de arquivos disponibiliza um conjunto de rotinas que permite às aplicações realizarem operações de E/S, como tradução de nomes em endereços, leitura e gravação de dados e criação/eliminação de arquivos. Na realidade, as rotinas de E/S têm como função disponibilizar uma interface simples e uniforme entre a aplicação e os diversos dispositivos. A figura 3 ilustra a comunicação entre aplicação e dispositivos de maneira simplificada.

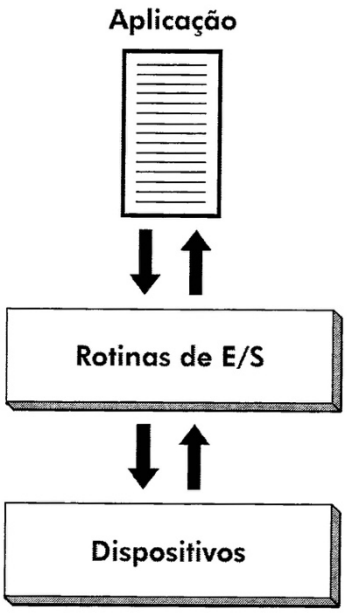


Figura 3: Operações de entrada/saída

A tabela a seguir apresenta algumas destas rotinas encontradas na maioria das implementações de sistemas de arquivos:

Rotina	*Descrição*
CREATE	*Criação de arquivo*
OPEN	*Abertura de um arquivo*
READ	*Leitura de um arquivo*
WRITE	*Gravação em um arquivo*
CLOSE	*Fechamento de um arquivo*
DELETE	*Eliminação de um arquivo*

2.4.Atributos

Cada arquivo possui informações de controle denominadas *atributos*. Os atributos variam dependendo do sistema de arquivos, porém alguns, como tamanho do arquivo, proteção, identificação do criador e data de criação estão presentes em quase todos os sistemas.

Alguns atributos especificados na criação do arquivo não podem ser modificados em função de sua própria natureza, como organização e data/hora de criação. Outros são alterados pelo próprio sistema operacional, como tamanho e data/hora do último backup realizado. Existem ainda atributos que podem ser modificados pelo próprio usuário, como proteção do arquivo, tamanho máximo e senha de acesso. Na tabela abaixo, são apresentados os principais atributos presentes nos sistemas de arquivos:

Atributos	*Descrição*
Tamanho	*Especifica o tamanho do arquivo*
Proteção	*Código de proteção de acesso*
Proprietário	*Identifica o criador do arquivo*
Criação	*Data e hora de criação do arquivo*
Senha	*Senha necessária para acessar o arquivo*
Organização	*Indica a organização lógica dos registros*
Backup	*Data e hora do último backup realizado*

3. DIRETÓRIOS

A estrutura de *diretórios* é como o sistema organiza logicamente os diversos arquivos contidos em um disco. O diretório é uma estrutura de dados que contém entradas

associadas aos arquivos onde cada entrada armazena informações como localização física, nome, organização e demais atributos.

Quando um arquivo é aberto, o sistema operacional procura a sua entrada na estrutura de diretórios, armazenando as informações sobre atributos e localização do arquivo em uma tabela mantida na memória principal. Esta tabela contém todos os arquivos abertos, sendo fundamental para aumentar o desempenho das operações com arquivos. É importante que ao término do uso de arquivos, este seja fechado, ou seja, que se libere o espaço na tabela de arquivos abertos.

A implementação mais simples de uma estrutura de diretórios é chamado de nível único (single-level directory). Neste caso, somente existe um único diretório contendo todos os arquivos do disco. Este modelo é bastante limitado, já que não permite que usuários criem arquivos com o mesmo nome, o que ocasionaria um conflito no acesso aos arquivos.

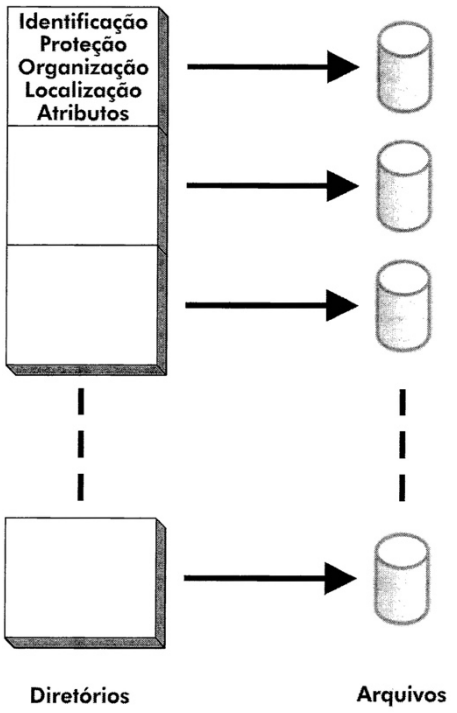


Figura 4: Estrutura de diretórios de nível único

Como o sistema de nível único é bastante limitado, uma evolução do modelo foi a implementação de uma estrutura onde para cada usuário existiria um diretório particular denominado User File Directory (UFD). Com esta implementação, cada usuário passa a poder criar arquivos com qualquer nome, sem a preocupação de conhecer os demais arquivos do disco.

Para que o sistema possa localizar arquivos nessa estrutura, deve haver um nível de diretório adicional para controlar os diretórios individuais dos usuários. Este nível,

denominado Master File Directory (MFD), é indexado pelo nome do usuário, onde cada entrada aponta para o diretório pessoal.

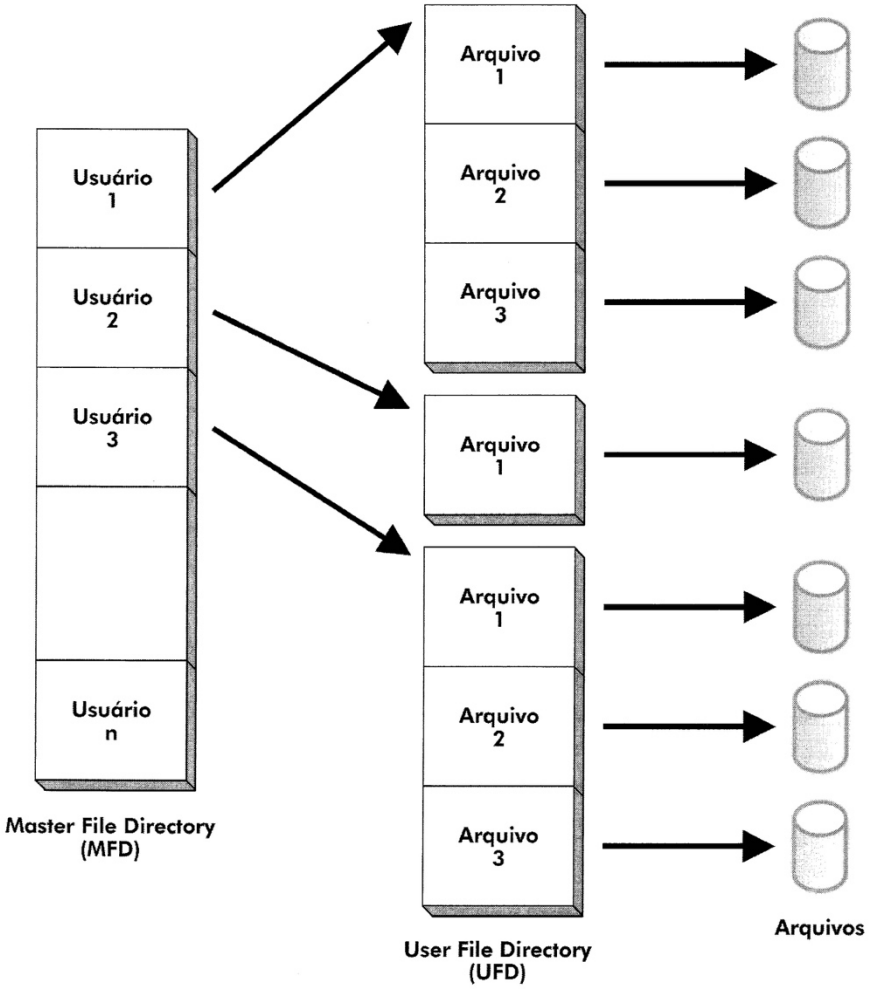


Figura 5: Estrutura de diretórios com dois níveis

A estrutura de diretórios com dois níveis é análoga a uma estrutura de dados em árvore, onde o MFD é a raiz, os galhos são os UFD e os arquivos são as folhas. Neste tipo de estrutura, quando se referencia um arquivo, é necessário especificar, além do seu nome, o diretório onde se localiza. Esta referência é chamada de path (caminho). Como exemplo, caso o usuário PAULO necessite acessar um arquivo próprio chamado DOCUMENTO.TXT, este pode ser referenciado como /PAULO/DOCUMENTO.TXT. Cada sistema de arquivos possui sua própria sintaxe para especificação de diretórios e arquivos.

Sob o ponto de vista do usuário, a organização dos seus arquivos em um único diretório não permite uma organização adequada. A extensão do modelo de dois níveis para um de múltiplos níveis permitiu que os arquivos fossem logicamente melhor organizados. Este novo modelo, chamado estrutura de diretórios em árvore (tree-structured directory), é adotado pela maioria dos sistemas.

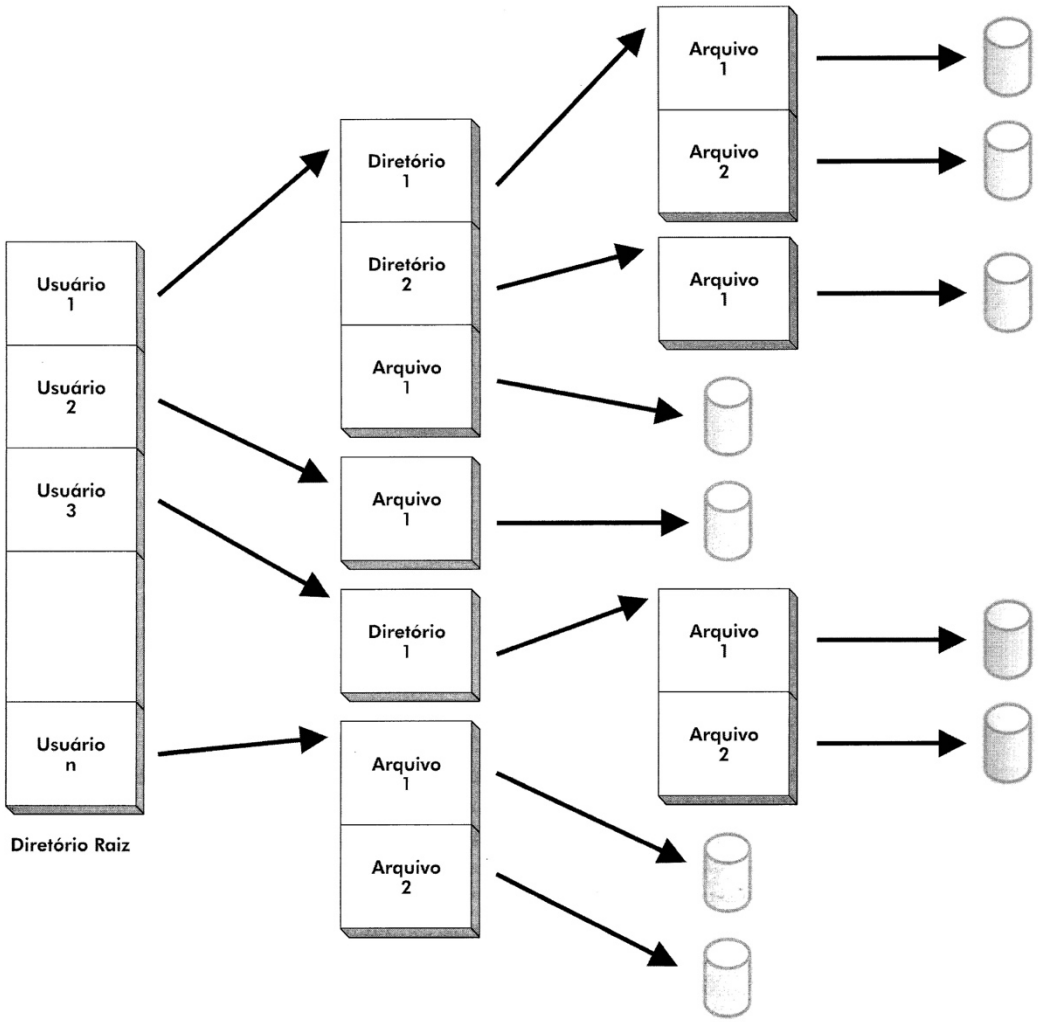


Figura 6: Estrutura de diretórios em árvore

Na estrutura em árvore, cada usuário pode criar diversos níveis de diretórios, também chamados de subdiretórios. Cada diretório pode conter arquivos ou outros diretórios. O número de níveis de uma estrutura em árvore é dependente do sistema de arquivos de cada sistema operacional. No OpenVMS são possíveis oito níveis de diretórios.

Um arquivo, nesta estrutura em árvore, pode ser especificado unicamente através de um path absoluto, descrevendo todos os diretórios percorridos a partir da raiz (MFD) até o diretório no qual o arquivo está ligado. Na figura 7, o path absoluto do arquivo SOMA.EXE é /PAULO/PROGRAMAS. Na maioria dos sistemas, os diretórios também são tratados como arquivos, possuindo identificação e atributos, como proteção, identificador do criador e data de criação.

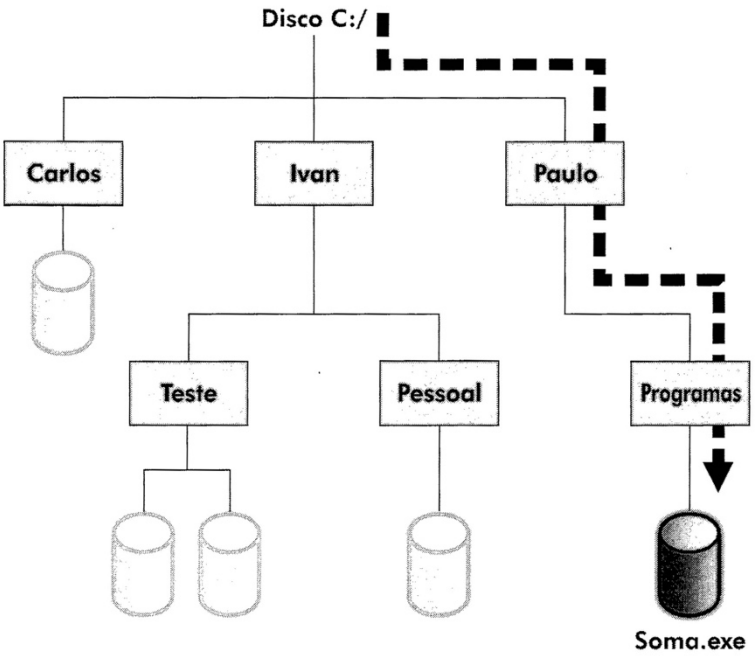


Figura 7: Path de um arquivo

4. GERÊNCIA DE ESPAÇO LIVRE EM DISCO

A criação de arquivos em disco exige que o sistema operacional tenha o controle de quais áreas ou blocos no disco estão livres. Este controle é realizado utilizando-se alguma estrutura de dados que armazena informações que possibilitam ao sistema de arquivos gerenciar o espaço livre do disco. Nesta estrutura, geralmente uma lista ou tabela, é possível identificar blocos livres que poderão ser alocados a um novo arquivo. Neste caso, o espaço é removido da estrutura para que não seja reutilizado. No momento em que um arquivo é eliminado, todos os seus blocos são liberados para a lista de espaços livres.

A forma mais simples de implementar uma estrutura de espaços livres é através de uma tabela denominada mapa de bits (bit map). Cada entrada na tabela é associada a um bloco do disco representado por um bit, podendo assumir valor igual a 0 (indicando bloco livre) ou 1 (indicando bloco alocado). Na figura 8a, podemos observar um exemplo desta implementação que apresenta como principal problema um excessivo gasto de memória, já que para cada bloco do disco deve existir uma entrada na tabela.

Uma segunda maneira de realizar este controle é com uma estrutura de lista encadeada de todos os blocos livres do disco. Para que isto seja possível, cada bloco possui uma área reservada para armazenamento do endereço do próximo bloco. A partir do primeiro bloco livre é, então, possível o acesso sequencial aos demais de forma encadeada (figura 8b). Este esquema apresenta algumas restrições se considerarmos que, além do espaço utilizado no bloco com informação de controle, o algoritmo de busca de espaço livre sempre deve realizar uma pesquisa sequencial na lista.

Uma outra solução leva em consideração que blocos contíguos são geralmente alocados ou liberados simultaneamente. Podemos, desta forma, enxergar o disco como um conjunto de segmentos de blocos livres. 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. Esta técnica de gerência de espaço livre é conhecida como tabela de blocos livres (figura 8c).

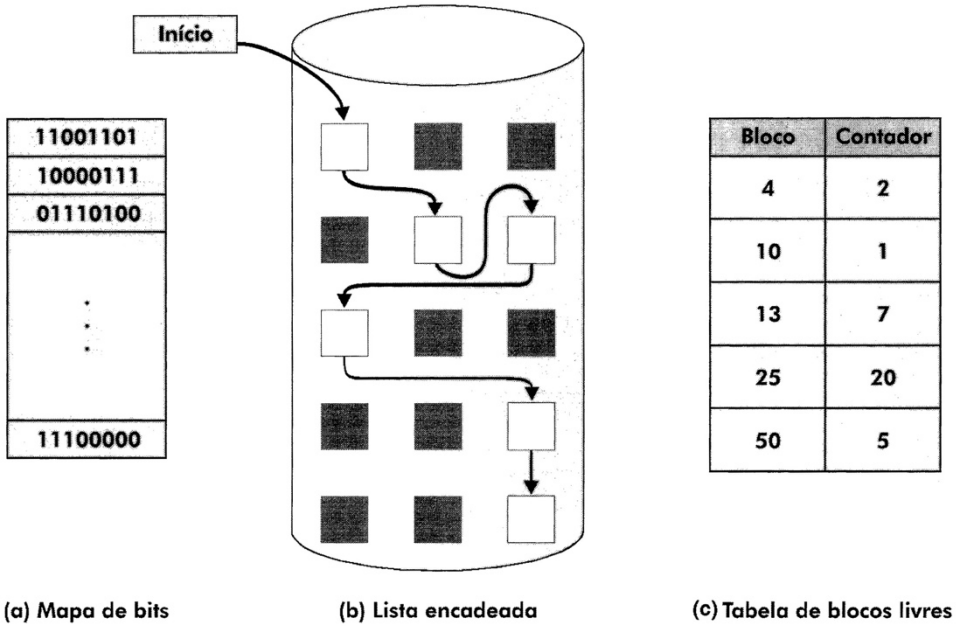


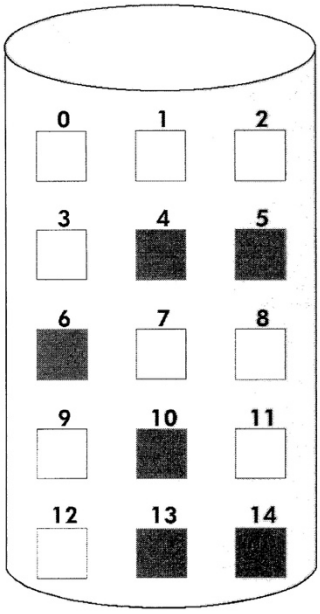
Figura 8: Alocação de espaço em disco

5. GERÊNCIA DE ALOCAÇÃO DE ESPAÇO EM DISCO

Da mesma forma que o sistema operacional gerencia os espaços livres no disco, a gerência dos espaços alocados aos arquivos é de fundamental importância em um sistema de arquivos. A seguir as principais técnicas são apresentadas.

5.1.Alocação Contígua

A *alocação contígua* consiste em armazenar um arquivo em blocos sequencialmente dispostos no disco. Neste tipo de alocação, o sistema localiza um arquivo através do endereço do primeiro bloco e da sua extensão em blocos.



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

Figura 9: Alocação contígua

O acesso a arquivos dispostos contiguamente no disco é bastante simples tanto para a forma sequencial quanto para a direta. Seu principal problema é a alocação de espaço livre para novos arquivos. Caso um arquivo deva ser criado com determinado tamanho, é necessário existir uma quantidade suficiente de blocos contíguos no disco para realizar a alocação.

Para este tipo de alocação, podemos enxergar o disco como um grande vetor, onde os elementos podem ser considerados segmentos com tamanhos diferentes de blocos contíguos. Estes segmentos estão dispostos alternadamente entre segmentos ocupados (blocos alocados) e segmentos livres (blocos livres). No momento em que o sistema operacional deseja alocar espaço para armazenar um novo arquivo, pode existir mais de um segmento livre disponível com o tamanho exigido. Neste caso, é necessário que alguma estratégia de alocação seja adotada para selecionar qual o segmento na lista de blocos livres deve ser escolhido. Analisaremos a seguir as principais estratégias:

- First-fit

Neste caso, o primeiro segmento livre com tamanho suficiente para alocar o arquivo é selecionado. A busca na lista é sequencial, sendo interrompida tão logo se localize um segmento com tamanho adequado.

- Best-fit

A alocação best-fit seleciona o menor segmento livre disponível com tamanho suficiente para armazenar o arquivo. A busca em toda lista se faz necessária para a seleção do segmento, a não ser que a lista esteja ordenada por tamanho.

Worst-fit

Neste caso, o maior segmento é alocado. Mais uma vez a busca em toda a lista se faz necessária, a menos que exista uma ordenação por tamanho.

Independente da estratégia utilizada, a alocação contígua apresenta um problema chamado fragmentação dos espaços livres. Como os arquivos são criados e eliminados frequentemente, os segmentos livres vão se fragmentando em pequenos pedaços por todo o disco. O problema pode tornar-se crítico quando um disco possui blocos livres disponíveis, porém não existe um segmento contíguo em que o arquivo possa ser alocado.

O problema da fragmentação pode ser contornado através de rotinas que reorganizem todos os arquivos no disco de maneira que só exista um único segmento de blocos livres. Este procedimento, denominado desfragmentação, geralmente utiliza uma área de trabalho no próprio disco ou em fita magnética. Existe um grande consumo de tempo neste tipo de operação. É importante também ressaltar que a desfragmentação é um procedimento com efeito temporário e deve, portanto, ser realizada periodicamente.

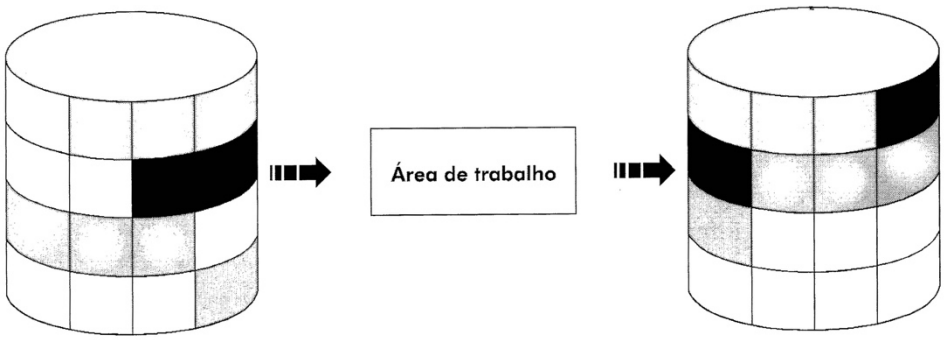


Figura 10: Desfragmentação

Podemos concluir que a alocação contígua apresenta alguns inconvenientes, sendo o principal problema a determinação do espaço em disco necessário a um arquivo. Nem sempre, no momento da criação de um arquivo, é possível determinar qual o seu tamanho em definitivo, podendo posteriormente existir a necessidade de extensão. Por

esta operação ser complexa na alocação contígua, a pré-alocação de espaço é uma solução que, apesar de resolver o problema, pode ocasionar que parte do espaço alocado permaneça ocioso por um longo período de tempo.

5.2.Alocação Encadeada

Na *alocação encadeada,* um arquivo pode ser organizado como um conjunto de blocos ligados logicamente no disco, independente da sua localização física. Cada bloco deve possuir um ponteiro para o bloco seguinte do arquivo e assim sucessivamente.

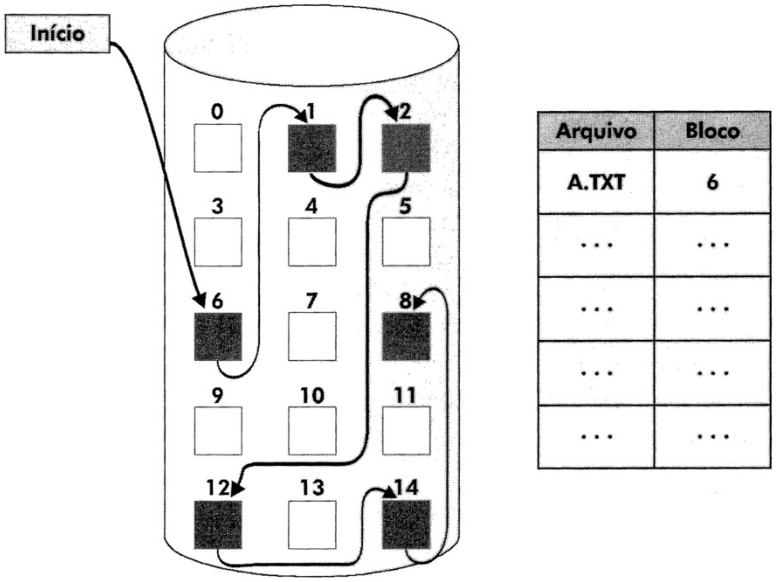


Figura 11: Alocação encadeada

A fragmentação dos espaços livres, apresentada anteriormente, não ocasiona nenhum problema na alocação encadeada, pois os blocos livres alocados para um arquivo não precisam necessariamente estar contíguos. O que ocorre neste método é a fragmentação de arquivos, que é a quebra do arquivo em diversos pedaços denominados extents.

A fragmentação resulta no aumento do tempo de acesso aos arquivos, pois o mecanismo de leitura/gravação do disco deve se deslocar diversas vezes sobre sua superfície para acessar cada extent (excessivo tempo de seek). Para otimizar o tempo das operações de E/S neste tipo de sistema, é importante que o disco seja periodicamente desfragmentado. Apesar de ter propósitos diferentes, o procedimento de desfragmentação é idêntico ao já apresentado na alocação contígua.

A alocação encadeada só permite que se realize acesso sequencial aos blocos dos arquivos. Isto constitui uma das principais desvantagens desta técnica, já que não é possível o acesso direto aos blocos. Além disso, essa técnica desperdiça espaço nos blocos com o armazenamento de ponteiros.

5.3.Alocação Indexada

A *alocação indexada* soluciona uma das principais limitações da alocação encadeada, que é a impossibilidade do acesso direto aos blocos dos arquivos. O princípio desta técnica é manter os ponteiros de todos os blocos do arquivo em uma única estrutura denominada bloco de índice.

A alocação indexada, além de permitir o acesso direto aos blocos do arquivo, não utiliza informações de controle nos blocos de dados, como existente na alocação encadeada.

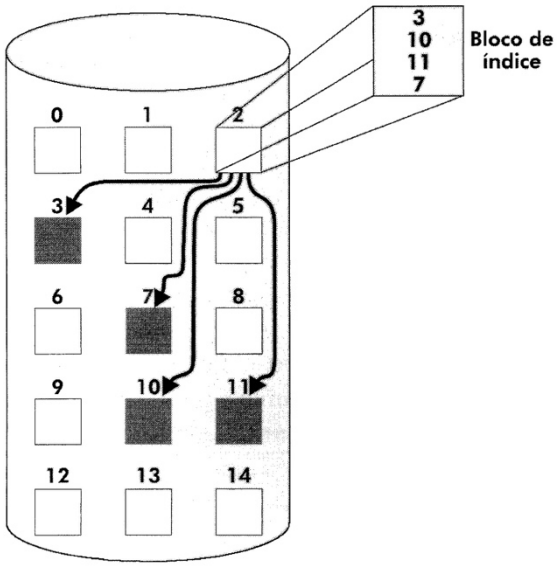


Figura 12: Alocação indexada

6. PROTEÇÃO DE ACESSO

Considerando que os meios de armazenamento são compartilhados entre diversos usuários, é de fundamental importância que mecanismos de proteção sejam implementados para garantir a proteção individual de arquivos e diretórios. Qualquer sistema de arquivos deve possuir mecanismos próprios para proteger o acesso às informações gravadas em discos e fitas, além de possibilitar o compartilhamento de arquivos entre usuários, quando desejado.

Em geral, o tipo de acesso a arquivos é implementado mediante a concessão ou não dos diferentes acessos que podem ser realizados, como leitura (read), gravação (write), execução (execute) e eliminação (delete). O acesso de leitura está relacionado com qualquer tipo de operação em que o arquivo possa ser visualizado, como a exibição do seu conteúdo, edição através de um editor de textos ou até mesmo a cópia para criação de um novo arquivo. O acesso de gravação está relacionado à alteração no conteúdo do

arquivo, como inclusão ou alteração de registros. O acesso de execução só tem sentido quando associado a arquivos executáveis ou arquivos de comandos, indicando o direito de execução do arquivo. Finalmente, o acesso de eliminação expressa a permissão para eliminação do arquivo.

O controle de acesso à operações realizadas com diretórios possui diferenças em relação às operações com arquivos. Controle da criação/eliminação de arquivos nos diretórios, visualização do seu conteúdo e eliminação do próprio diretório são operações que também devem ser protegidas.

Existem diferentes mecanismos e níveis de proteção, cada qual com suas vantagens e desvantagens – para cada tipo de sistema, um modelo é mais adequado do que outro. A seguir temos três diferentes mecanismos de proteção, presentes na maioria dos sistemas de arquivos.

6.1.Senha de Acesso

A associação de uma *senha de acesso* a um arquivo é um princípio bastante simples. O controle de acesso se resume ao usuário ter o conhecimento da senha e, consequentemente, ter a liberação do acesso ao arquivo concedida pelo sistema.

Como cada arquivo possui apenas uma senha, o acesso é liberado ou não na sua totalidade. Isso significa que não é possível determinar quais tipos de operação podem ou não ser concedidas. Outra desvantagem deste método é a dificuldade de compartilhamento de arquivos, pois, além do dono do arquivo, todos os demais usuários teriam que conhecer a senha de acesso.

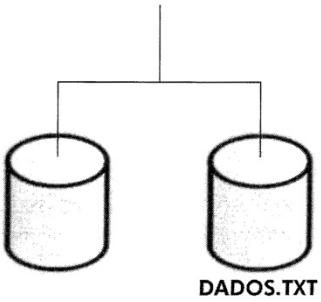
6.2.Grupos de Usuário

A proteção baseada em *grupos de usuários* é implementada por diversos sistemas operacionais. Este tipo de proteção tem como princípio a associação de cada usuário do sistema a um grupo. Os grupos de usuários são organizados logicamente com o objetivo de compartilhar arquivos e diretórios – os usuários que desejam compartilhar arquivos entre si devem pertencer a um mesmo grupo.

Esse mecanismo implementa três níveis de proteção ao arquivo: owner (dono), group (grupo) e all (todos). Na criação do arquivo, o usuário especifica se o arquivo deve ser acessado somente pelo seu criador, pelos usuários do grupo ao qual ele pertence ou por todos os usuários do sistema. Nessa especificação é necessário associar o tipo de acesso (leitura, escrita, execução e eliminação) aos três níveis de proteção.

Vejamos um exemplo na figura 13, onde a proteção do arquivo DADOS.TXT libera completamente o acesso para o dono, permite somente o compartilhamento para leitura dentro do grupo e não libera qualquer tipo de acesso para os demais usuários do sistema.

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



Nível de Proteção	Tipo de Acesso
Owner	Leitura Escrita Execução Eliminação
Group	Leitura
All	—

Figura 13: Proteção por grupos de usuários

6.3.Lista de Controle de Acesso

A *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. Nesse caso, quando um usuário tenta acessar um arquivo, o sistema operacional verifica se a lista de controle autoriza a operação desejada.

O tamanho dessa estrutura de dados pode ser bastante extenso se considerarmos que um arquivo pode ter seu acesso compartilhado por diversos usuários. Além deste fato, existe um overhead adicional, se comparado com o mecanismo de proteção por grupo de usuários, devido à pesquisa sequencial que o sistema deverá realizar na lista sempre que um acesso for solicitado.

Em determinados sistemas de arquivos é possível encontrar tanto o mecanismo de proteção por grupos de usuários quanto o de lista de controle de acesso, oferecendo, desta forma, uma maior flexibilidade ao mecanismo de proteção de arquivos e diretórios.

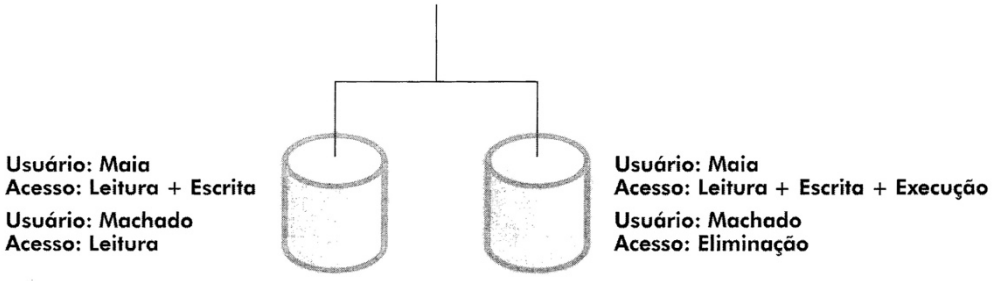


Figura 14: Lista de controle de acessos

7. COMPRESSÃO DE DADOS

Compressão de dados é uma técnica utilizada para economizar espaço nos arquivos. Os três métodos de compressão de dados nas bases de dados são: caracteres repetidos, termos repetidos e compressão de front end.

Os *registros com caracteres repetidos* podem ser abreviados. Por exemplo, dados em um campo de tamanho fixo podem incluir um pequeno nome seguido de vários caracteres vazios; esse campo pode ser substituído por outro de tamanho variável e um código especial para indicar quantos caracteres vazios foram truncados.

Digamos que a cadeia de caracteres ADAMS tenha o seguinte aspecto quando é armazenada em um campo de 15 caracteres (*v* representa um caractere vazio):

ADAMSvvvvvvvvvv

Ao ser codificada, ela toma o seguinte aspecto:

ADAMSv10

Números com muitos zeros também podem ser encurtados por meio de um código para indicar quantos zeros devem ser acrescentados para recompor o número original. Por exemplo, se a entrada original for

300000000

a entrada codificada será:

3#8

Termos repetidos também podem ser comprimidos. Um método possível utiliza símbolos para representar as palavras mais comuns em uma base de dados. Por exemplo, em uma base de dados de alunos universitários, palavras como aluno, curso, professor, sala de aula, nota e departamento podem ser substituídos por um único caractere. Evidentemente, o sistema deve ser capaz de distinguir entre dados comprimidos e não comprimidos.

A *compressão de front-end* é utilizada nos sistema de gerenciamento de bases de dados para a compressão de índices. Por exemplo, uma base de dados de alunos onde os nomes são relacionados em ordem alfabética poderia ser comprimida conforme a tabela abaixo:

Lista original	**Lista comprimida**
Smith, Betty	Smith, Betty
Smith, Gino	7Gino
Smith, Donald	7Donald
Smithberger, John	5berger, John
Smithbren, Ali	6ren, Ali
Smithco, Rachel	5co, Rachel
Smither, Kevin	5er, Kevin
Smithers, Renny	7s, Renny
Snyder, Katherine	1nyder, Katherine

A compressão de dados implica uma compensação: ganha-se espaço de armazenagem mas perde-se tempo de processamento. Não se esqueça: para todos os

métodos de compressão de dados, o sistema deve ser capaz de distinguir entre dados comprimidos e dados originais.

8. IMPLEMENTAÇÃO DE CACHES

O acesso a disco é bastante lento se comparado ao acesso à memória principal, devido à arquitetura dos discos magnéticos. Este é o principal motivo das operações de E/S com discos serem um problema para o desempenho do sistema.

Com o objetivo de minimizar este problema, a maioria dos sistemas de arquivos implementa uma técnica denominada *buffer cache*. Neste esquema, o sistema operacional reserva uma área da memória para que se tornem disponíveis cache utilizados em operações de acesso ao disco. Quando uma operação é realizada, seja leitura ou gravação, o sistema verifica se a informação desejada se encontra no buffer cache. Em caso positivo, não é necessário o acesso ao disco. Caso o bloco requisitado não se encontre no cache, a operação de E/S é realizada e o cache é atualizado. Como existe uma limitação no tamanho do cache, cada sistema adota políticas para substituição de blocos como a FIFO ou a LRU.

Apesar de esta implementação melhorar o desempenho do sistema, aspectos de segurança devem ser levados em consideração. No caso de blocos de dados permanecerem por um longo período de tempo na memória principal, a ocorrência de problemas de energia pode ocasionar a perda de tarefas já realizadas e consideradas já salvas em disco.

Existem duas maneiras distintas de tratar este problema. No primeiro caso, o sistema operacional possui uma rotina que executa periodicamente em um intervalo de tempo, atualizando em disco todos os blocos modificados do cache. Uma segunda alternativa é, toda vez que um bloco do cache for modificado, que seja realizada imediatamente uma atualização no disco (write-through caches).

Analisando comparativamente as duas técnicas, podemos concluir que a primeira implica menor quantidade de operações de E/S, porém o risco de perda de dados é maior. Apesar de tal probabilidade ser pequena, pode ocorrer que dados atualizados de um arquivo e ainda no cache sejam perdidos no caso de faltar energia. Isto já não aconteceria nos caches do tipo write-through, em função do seu próprio funcionamento, porém o aumento considerável nas operações de E/S tornam este método menos eficiente. Atualmente, a maioria dos sistemas operacionais utiliza a primeira técnica de otimização.