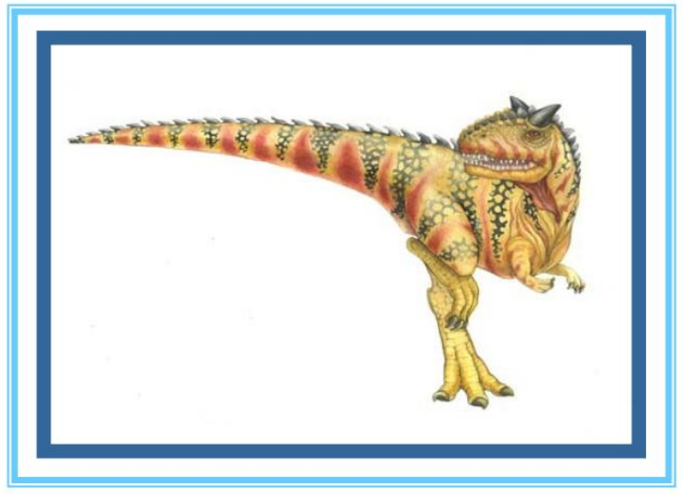
Capítulo 10: Sistema de arquivos Implementação



Capítulo 10: Implementação do sistema de arquivos

- ÿ Estrutura do sistema de arquivos
- ÿ Implementação do sistema de arquivos
- ÿ Implementação do diretório
- ÿ Métodos de alocação
- ÿ Gestão de Espaço Livre ÿ Eficiência e Desempenho
- ÿ Recuperação
- ÿ NFS
- ÿ Exemplo: Sistema de arquivos WAFL

Objetivos

ÿ Descrever os detalhes da implementação de sistemas de arquivos locais e estruturas de diretório

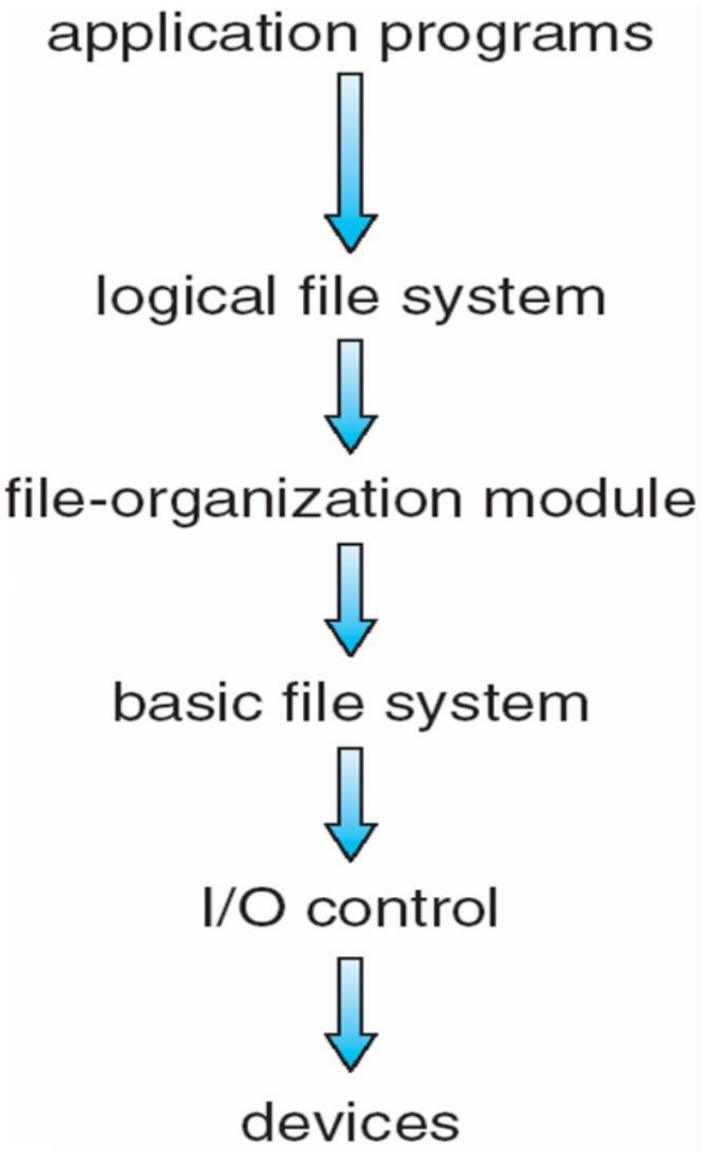
ÿ Descrever a implementação de sistemas de arquivos remotos

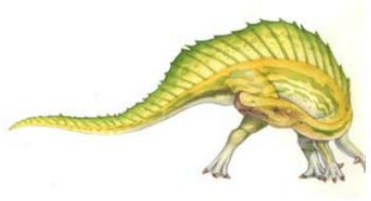
ÿ Discutir a alocação de blocos e algoritmos de blocos livres e compensações

Estrutura do sistema de arquivos

- Estrutura do arquivo
  - Unidade de armazenamento lógica
  - Coleta de informações relacionadas
- **O sistema de arquivos** reside no armazenamento secundário (discos)
  - Forneceu interface de usuário para armazenamento, mapeando lógica para física
  - Fornece acesso eficiente e conveniente ao disco, permitindo que os dados sejam armazenados, localizados e recuperados facilmente
- O disco fornece reescrita no local e acesso aleatório
  - Transferências de E/S realizadas em **blocos** de **setores** (geralmente 512 bytes)
- **Bloco de controle de arquivo** – estrutura de armazenamento que consiste em informações sobre um arquivo
- **O driver do dispositivo** controla o dispositivo físico
- Sistema de arquivos organizado em camadas

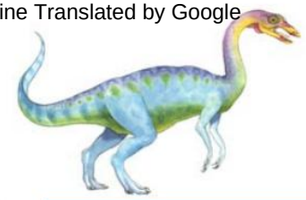
Sistema de arquivos em camadas





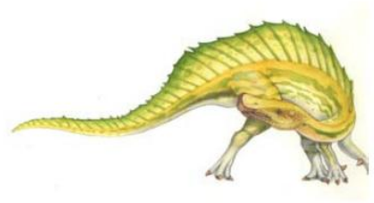
Camadas do sistema de arquivos

- **Os drivers de dispositivo** gerenciam dispositivos de E/S na camada de controle de E/S
  - Comandos como “ler drive1, cilindro 72, trilha 2, setor 10, no local de memória 1060” produzem resultados de baixa comandos específicos de hardware de nível para controlador de hardware
  - O comando **básico do sistema de arquivos ,** como “recuperar bloco 123”, é traduzido para o driver do dispositivo
- Também gerencia buffers de memória e caches (alocação, liberação, substituição)
  - Os buffers armazenam dados em trânsito
  - Os caches armazenam dados usados com frequência
  - **O módulo de organização de arquivos** compreende arquivos, endereços lógicos e blocos físicos
- Traduz o bloco lógico # para o bloco físico #
- Gerencia espaço livre, alocação de disco



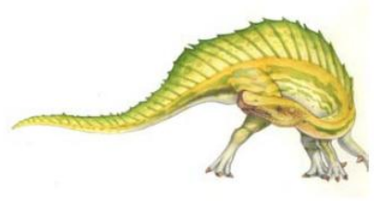
Camadas do sistema de arquivos (cont.)

- **O sistema de arquivos lógico** gerencia informações de metadados
  - Traduz o nome do arquivo em número de arquivo, identificador de arquivo, local, mantendo blocos de controle de arquivo **(inodes** no Unix)
  - Gerenciamento de diretório
  - Proteção
- Camadas úteis para reduzir a complexidade e a redundância, mas adicionam sobrecarga e podem diminuir o desempenho
  - Camadas lógicas podem ser implementadas por qualquer método de codificação de acordo com o designer do sistema operacional
- Muitos sistemas de arquivos, às vezes muitos dentro de um sistema operacional
  - Cada um com seu próprio formato (CD-ROM é ISO 9660; Unix tem **UFS,** FFS; Windows tem FAT, FAT32, NTFS, bem como disquete, CD, DVD Blu-ray, Linux tem mais de 40 tipos, com **sistema de arquivo estendido** ext2 e ext3 liderando; além de sistemas de arquivo distribuídos, etc.)
- Novos ainda chegando – ZFS, GoogleFS, Oracle ASM, FUSE



Implementação do sistema de arquivos

- Temos chamadas de sistema no nível da API, mas como implementamos suas funções?
  - Estruturas no disco e na memória
- **O bloco de controle de inicialização** contém informações necessárias para o sistema inicializar o SO a partir desse volume
  - Necessário se o volume contiver SO, geralmente o primeiro bloco do volume
- **Bloco de controle de volume (superbloco, tabela de arquivo mestre)** contém detalhes do volume
  - Total de blocos, número de blocos livres, tamanho do bloco, ponteiros de bloco livre ou matriz
- A estrutura de diretório organiza os arquivos
  - Nomes e números de inode, tabela de arquivo mestre
- **O Bloco de Controle de Arquivo (FCB)** por arquivo contém muitos detalhes sobre o arquivo
  - Número do inode, permissões, tamanho, datas
  - NFTS armazena em uma tabela de arquivo mestre usando estruturas de banco de dados relacionais



Um bloco de controle de arquivo típico

file permissions
file dates (create, access, write)
file owner, group, ACL
file size
file data blocks or pointers to file data blocks

Estruturas do sistema de arquivos na memória

ÿ Tabela de montagem que armazena montagens de sistemas de arquivos, pontos de montagem, tipos de sistemas de arquivos

ÿ A figura a seguir ilustra as estruturas de sistema de arquivos necessárias fornecidas pelos sistemas operacionais

ÿ A Figura 12-3(a) refere-se à abertura de um arquivo

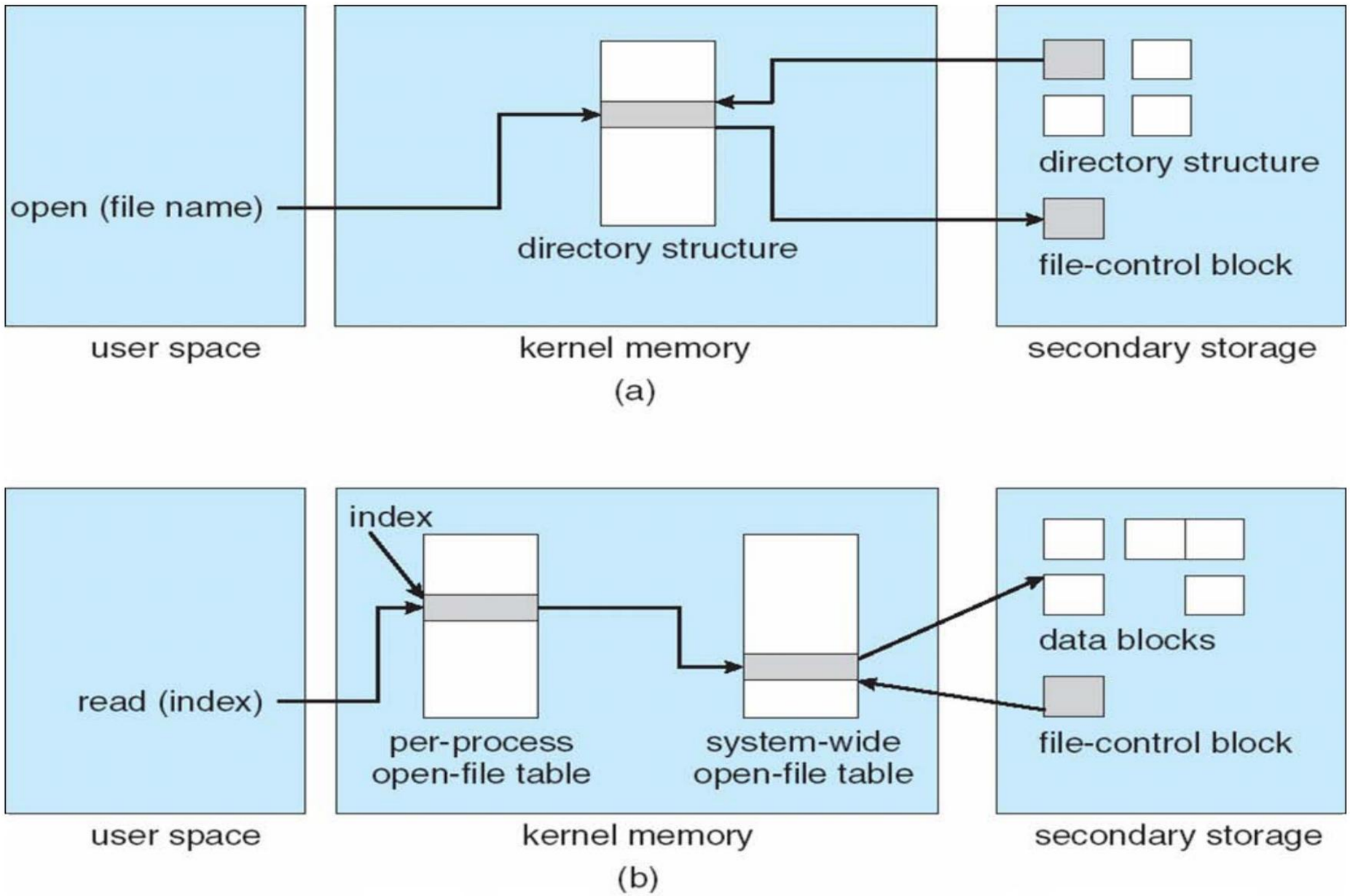
ÿ A Figura 12-3(b) refere-se à leitura de um arquivo

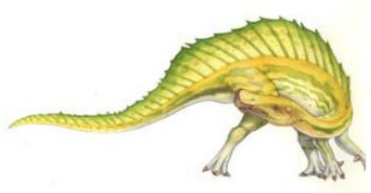
ÿ Além disso, os buffers contêm blocos de dados do armazenamento secundário

ÿ Open retorna um identificador de arquivo para uso subsequente

ÿ Dados de leitura eventualmente copiados para o endereço de memória do processo do usuário especificado

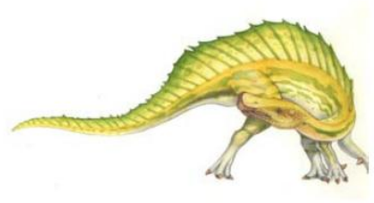
Estruturas do sistema de arquivos na memória

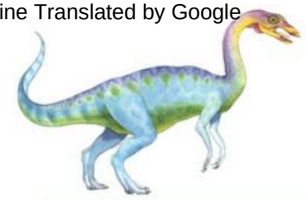




Partições e Montagem

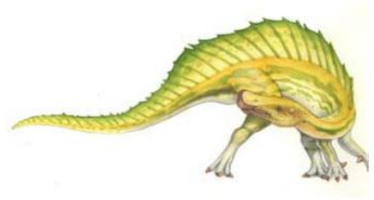
- A partição pode ser um volume que contém um sistema de arquivos (“cozido”) ou **bruto** – apenas uma sequência de blocos sem nenhum arquivo sistema
- O bloco de inicialização pode apontar para o volume de inicialização ou para o carregador de inicialização, um conjunto de blocos que contém código suficiente para saber como carregar o kernel do sistema de arquivos
  - Ou um programa de gerenciamento de inicialização para inicialização de vários sistemas operacionais
- **A partição raiz** contém o sistema operacional, outras partições podem conter outros sistemas operacionais, outros sistemas de arquivos ou ser raw
  - Montado no momento da inicialização
  - Outras partições podem ser montadas automaticamente ou manualmente
- No momento da montagem, a consistência do sistema de arquivos é verificada
  - Todos os metadados estão corretos?
    - Se não, corrija e tente novamente
    - Se sim, adicione à tabela de montagem, permita acesso



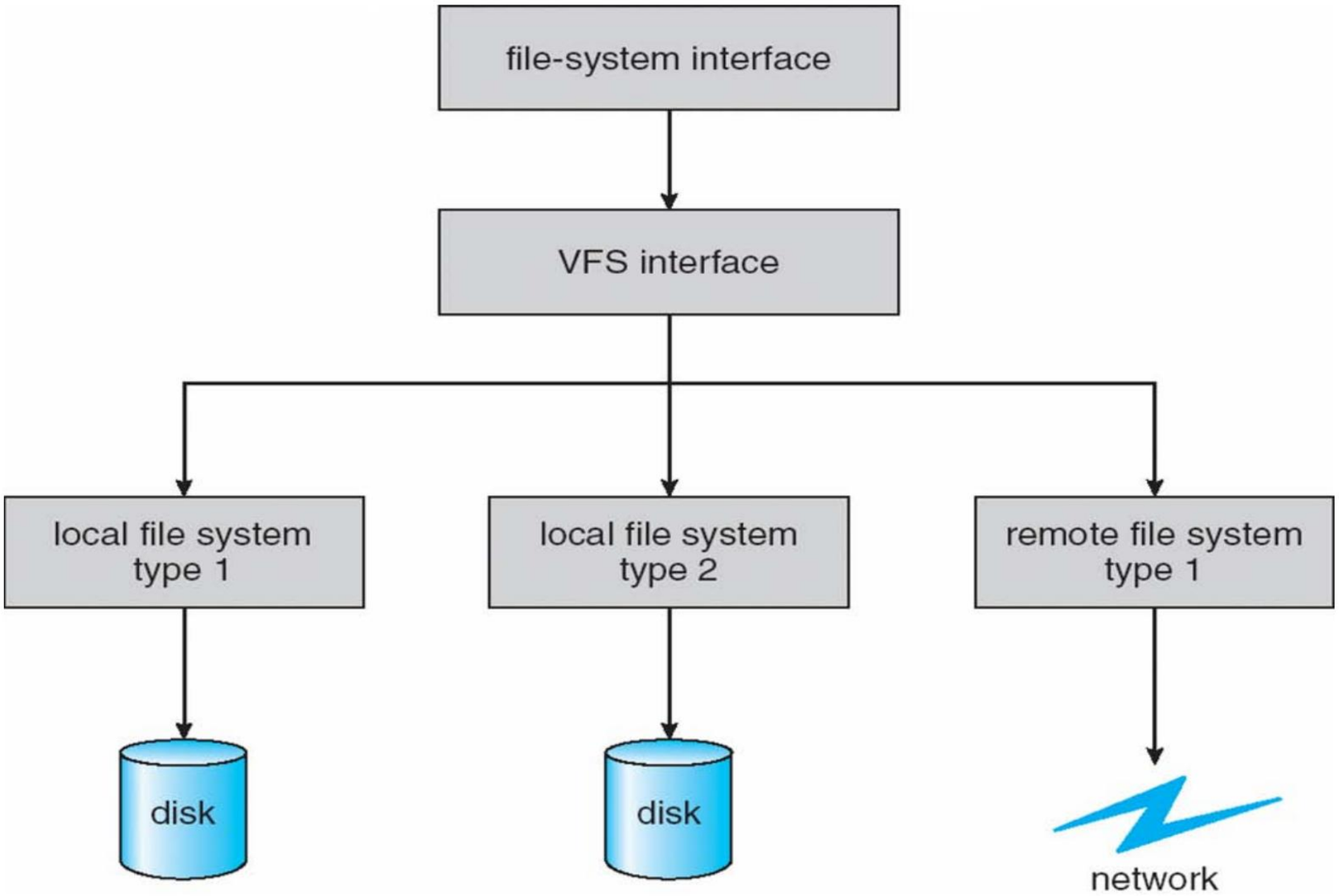


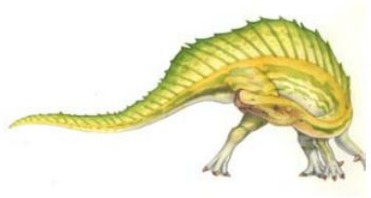
Sistemas de arquivos virtuais

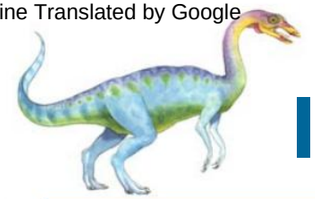
- Os sistemas de arquivos virtuais (VFS) no Unix fornecem uma maneira orientada a objetos de implementar sistemas de arquivos
- O VFS permite que a mesma interface de chamada de sistema (a API) seja usada para diferentes tipos de sistemas de arquivos .
  - Separa as operações genéricas do sistema de arquivos dos detalhes de implementação .
  - A implementação pode ser um dos muitos tipos de sistemas de arquivos ou sistemas de arquivos de rede.
    - Implementa vnodes que contêm inodes ou detalhes de arquivos de rede
  - Em seguida, despacha a operação para rotinas de implementação do sistema de arquivos apropriadas
- A API é para a interface VFS, em vez de qualquer tipo específico de sistema de arquivos



Visão esquemática do sistema de arquivos virtual







Implementação do Sistema de Arquivos Virtual

- Por exemplo, o Linux tem quatro tipos de objetos:
  - inode, arquivo, superbloco, dentry
- O VFS define um conjunto de operações nos objetos que devem ser implementadas
  - Cada objeto tem um ponteiro para uma tabela de funções
    - A tabela de funções possui endereços de rotinas para implementar essa função naquele objeto

Implementação de Diretório

ÿ **Lista linear** de nomes de arquivos com ponteiro para os blocos de dados

ÿ Simples de programar ÿ

Demorado para executar

ÿ Tempo de busca linear

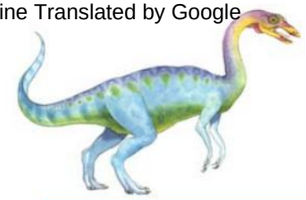
ÿ Poderia manter ordenado alfabeticamente por meio de lista vinculada ou usar árvore B+

ÿ **Tabela Hash** – lista linear com estrutura de dados hash

ÿ Diminui o tempo de busca no diretório

ÿ **Colisões** – situações em que dois nomes de arquivo são hash para o mesmo local

ÿ Só é bom se as entradas tiverem tamanho fixo ou usar o método de estouro encadeado



Métodos de alocação - Contíguos

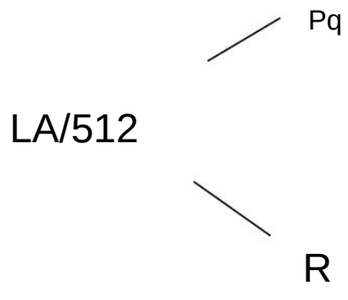
ÿ Um método de alocação refere-se a como os blocos de disco são alocados para arquivos:

ÿ **Alocação contígua** – cada arquivo ocupa um conjunto de blocos contíguos

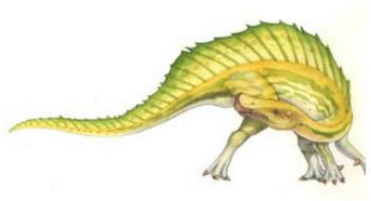
- ÿ Melhor desempenho na maioria dos casos
- ÿ Simples – apenas a localização inicial (bloco nº) e o comprimento (número de blocos) são necessários
- ÿ Os problemas incluem encontrar espaço para o arquivo, saber o tamanho do arquivo, fragmentação externa, necessidade de **compactação off-line (tempo de inatividade)** ou **on-line**

Alocação Contígua

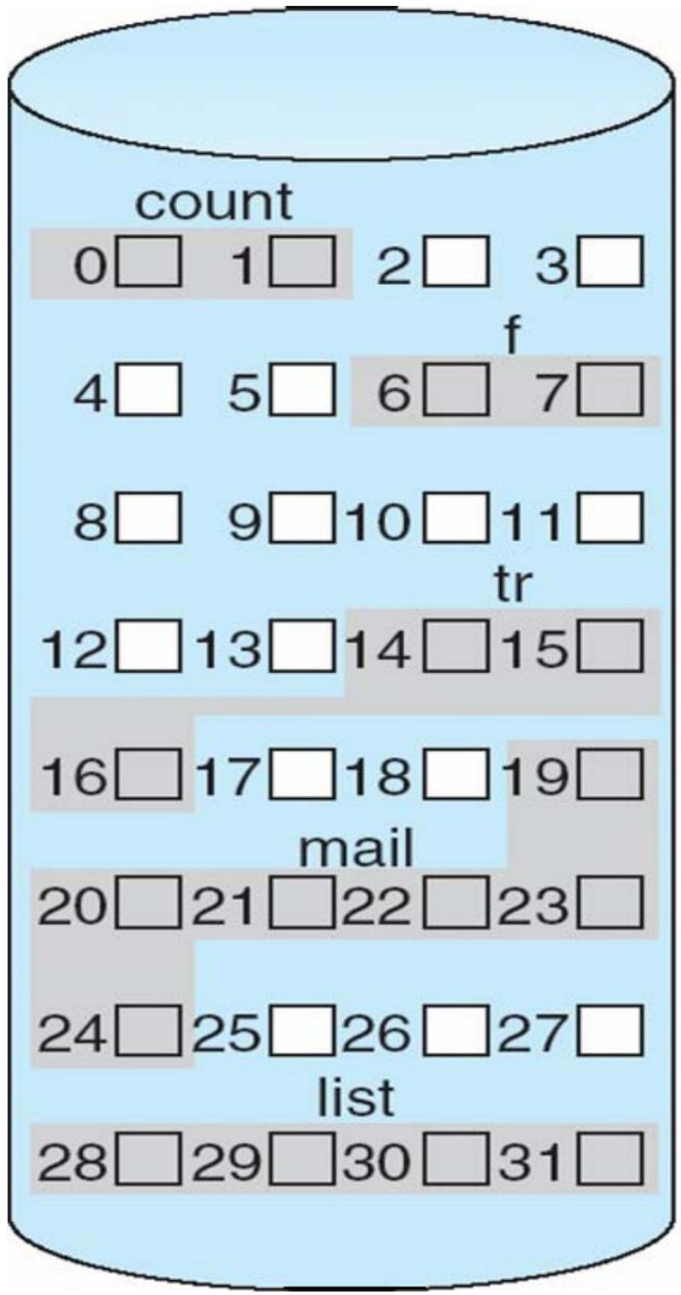
- Mapeamento do lógico para o físico



Bloco a ser acessado = Q + endereço inicial
Deslocamento no bloco = R



Alocação Contígua de Espaço em Disco



directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

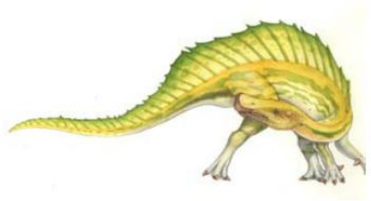
Sistemas baseados em extensão

ÿ Muitos sistemas de arquivos mais recentes (por exemplo, Veritas File System) usam um esquema de alocação contíguo modificado

ÿ Os sistemas de arquivos baseados em extensão alocam blocos de disco em extensões

ÿ Uma **extensão** é um bloco contíguo de discos

- ÿ As extensões são alocadas para alocação de arquivos
- ÿ Um arquivo consiste em uma ou mais extensões



Métodos de alocação - vinculados

ÿ **Alocação vinculada** – cada arquivo é uma lista vinculada de blocos

ÿ O arquivo termina em um ponteiro nulo

ÿ Sem fragmentação externa ÿ Cada

bloco contém um ponteiro para o próximo bloco ÿ Sem

compactação, fragmentação externa

ÿ Sistema de gerenciamento de espaço livre chamado quando novo bloco é necessário

ÿ Melhora a eficiência agrupando blocos em grupos, mas aumenta a fragmentação interna

ÿ A confiabilidade pode ser um problema ÿ

Localizar um bloco pode levar muitas E/Ss e buscas em disco ÿ Variação

FAT (File Allocation Table)

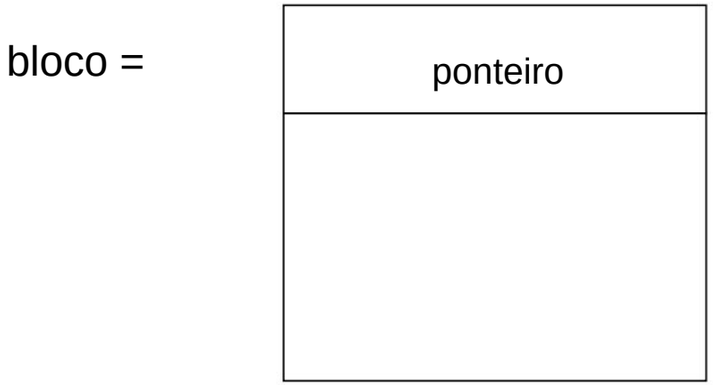
ÿ O início do volume tem uma tabela, indexada pelo número do bloco ÿ Muito

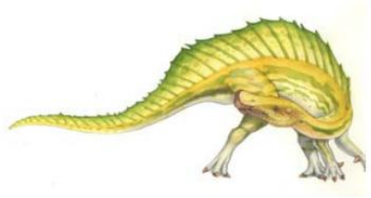
parecido com uma lista vinculada, mas mais rápido no disco e armazenável em cache

ÿ Nova alocação de bloco simples

Alocação vinculada

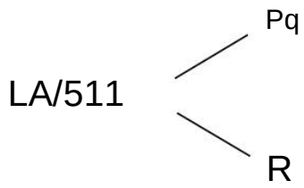
ÿ Cada arquivo é uma lista vinculada de blocos de disco: os blocos podem estar espalhados em qualquer lugar do disco



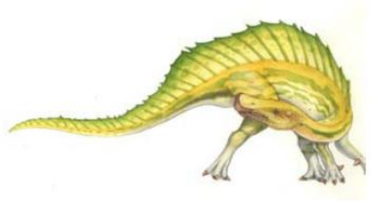


Alocação vinculada

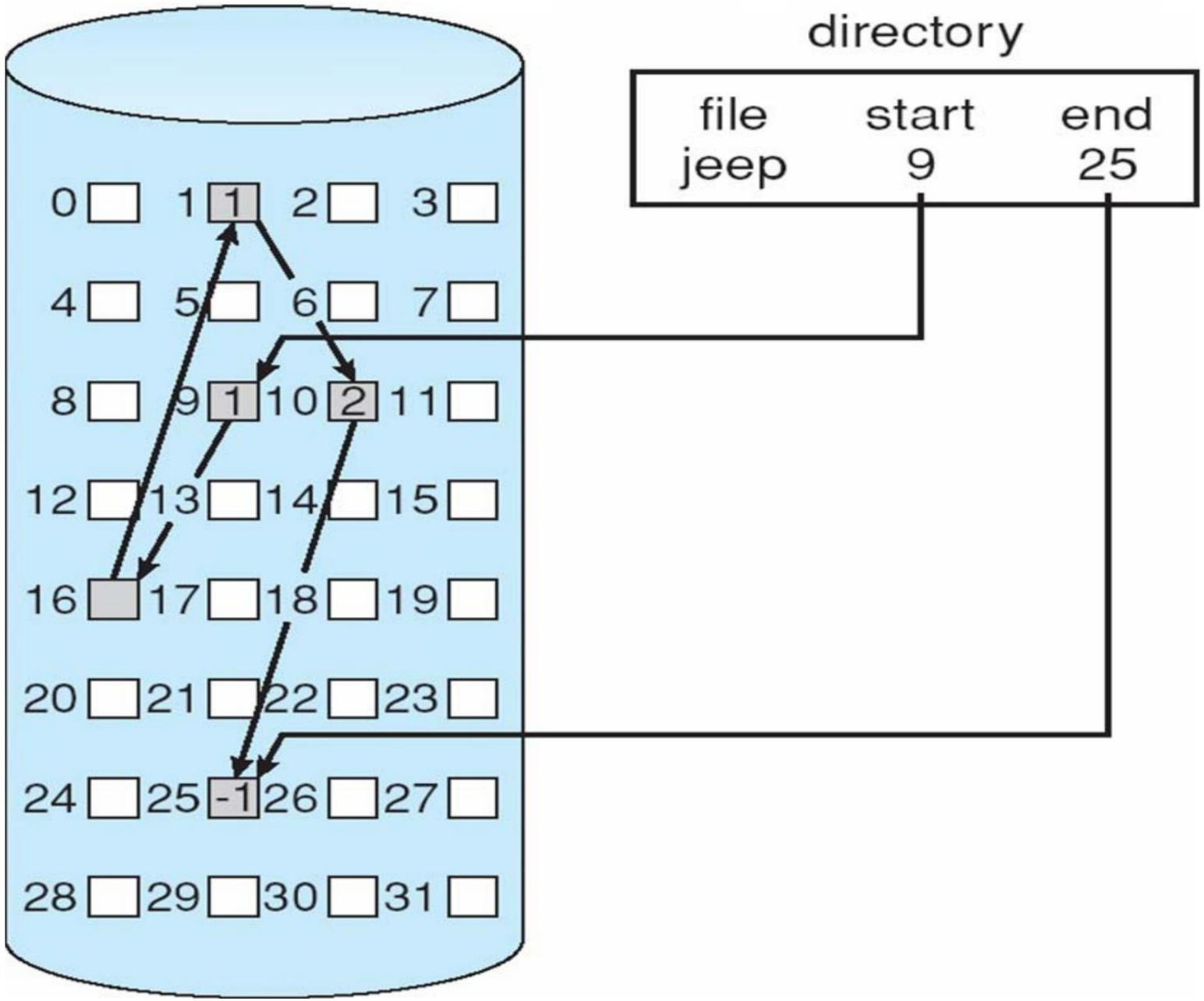
- Mapeamento



O bloco a ser acessado é o Q-ésimo bloco na cadeia vinculada de blocos que representam o arquivo.
Deslocamento no bloco = R + 1



Alocação vinculada



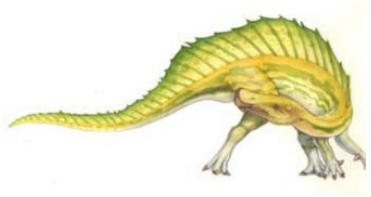
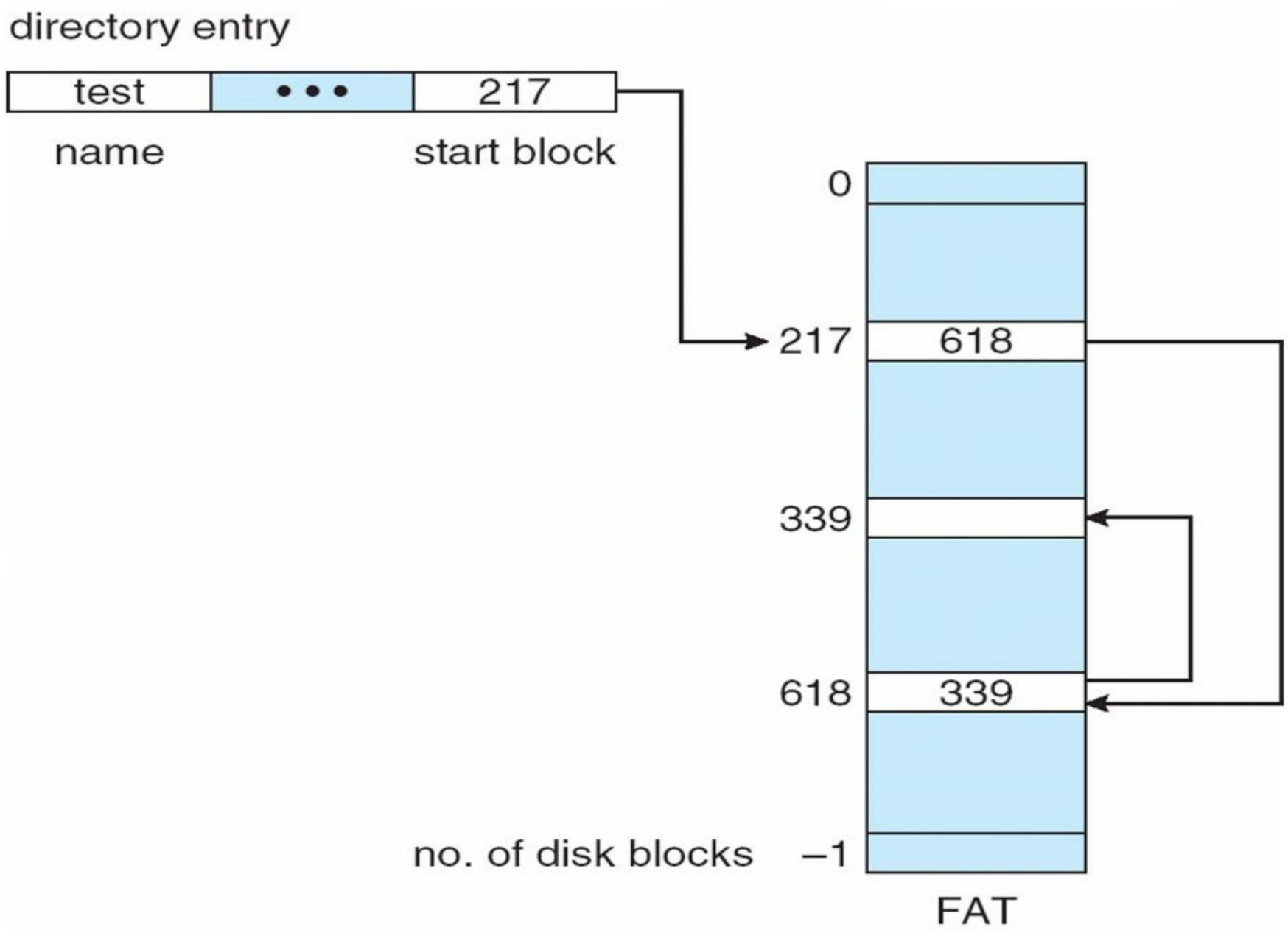
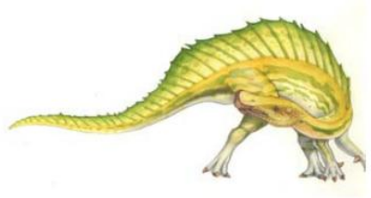


Tabela de alocação de arquivos





Métodos de Alocação - Indexados

- **Alocação indexada**
  - Cada arquivo tem seu(s) próprio **(s) bloco(s)** de índice de ponteiros para seus blocos de dados

- Visão lógica

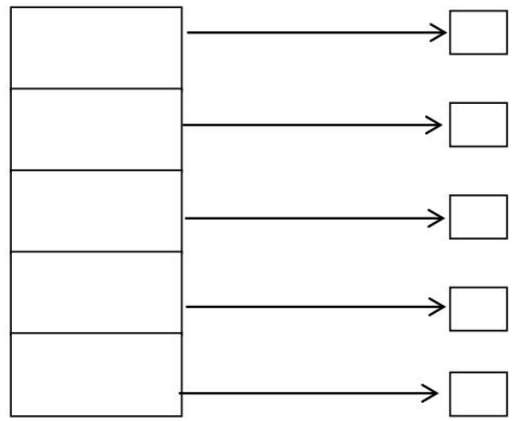
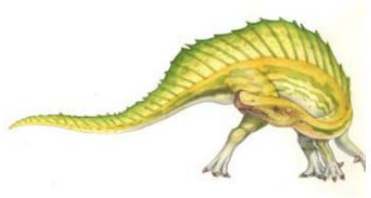
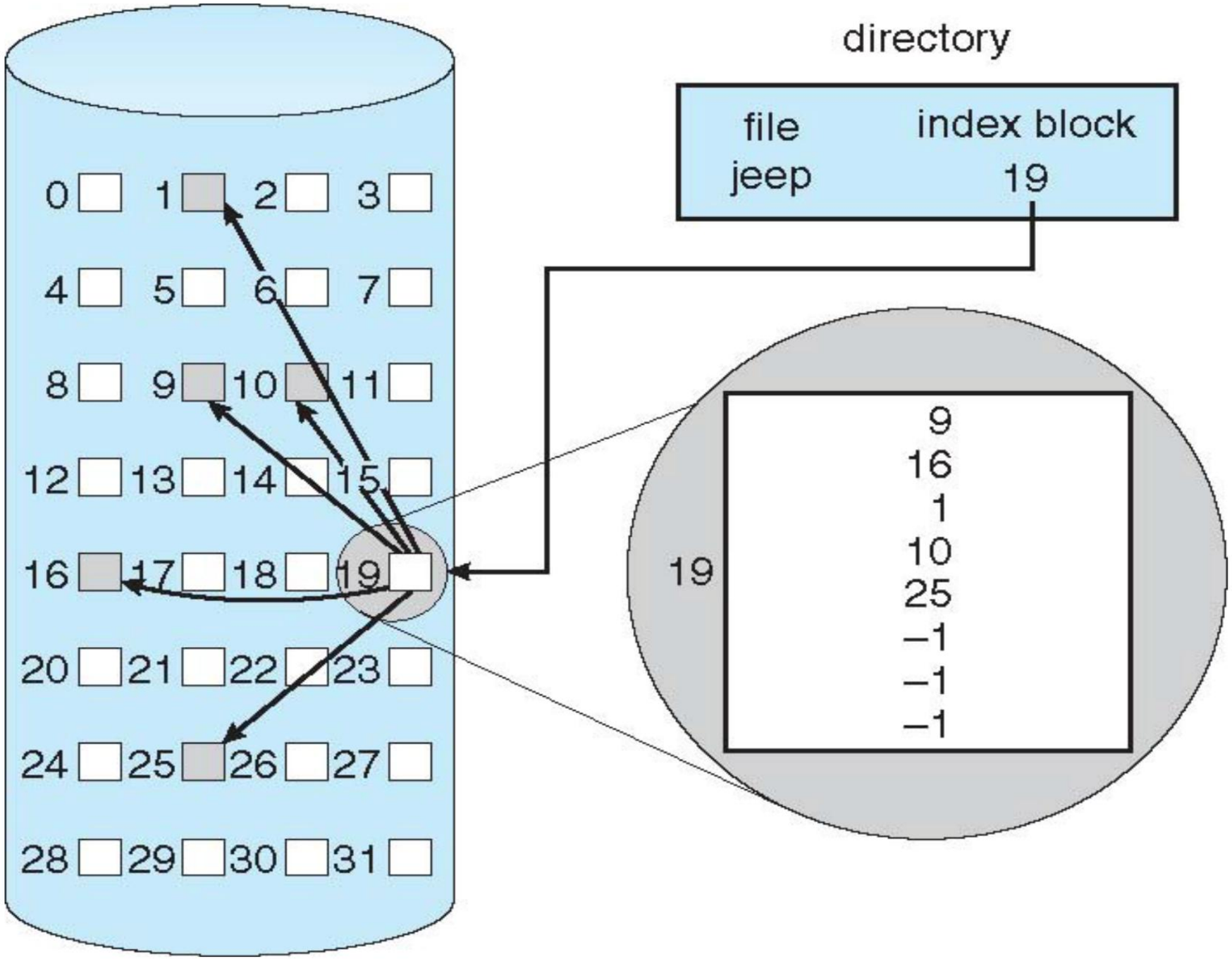


tabela de índice

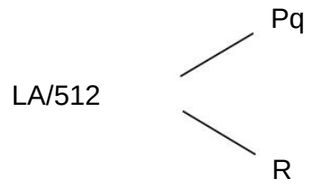


Exemplo de alocação indexada



Alocação Indexada (Cont.)

- Precisa de tabela de índice
- Acesso aleatório
- Acesso dinâmico sem fragmentação externa, mas tem sobrecarga de bloco de índice
- Mapeamento de lógico para físico em um arquivo de tamanho máximo de 256K bytes e tamanho de bloco de 512 bytes.
  - Precisamos apenas de 1 bloco para a tabela de índice



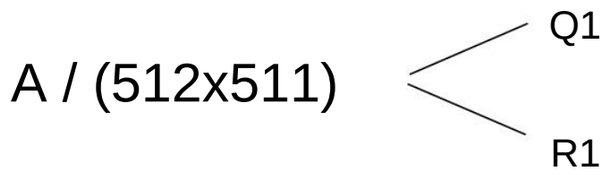
Q = deslocamento para a tabela de índice

R = deslocamento para dentro do bloco

Alocação Indexada – Mapeamento (Cont.)

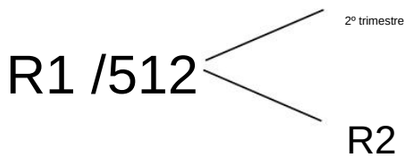
- Mapeamento de lógico para físico em um arquivo de comprimento ilimitado (tamanho de bloco de 512 palavras)

- Esquema vinculado – Blocos de link da tabela de índice (sem limite de tamanho)



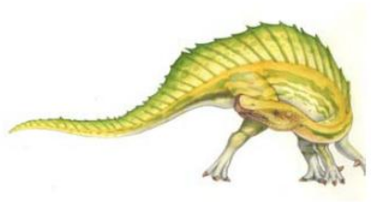
Q1 = bloco de tabela de índice

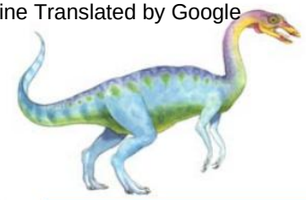
R1 é usado da seguinte forma:



Q2 = deslocamento para o bloco da tabela de índice

Deslocamento R2 no bloco de arquivo:





Alocação Indexada – Mapeamento (Cont.)

ÿ Índice de dois níveis (blocos de 4K podem armazenar 1.024 ponteiros de quatro bytes no índice externo -> 1.048.567 blocos de dados e arquivo tamanho de até 4 GB)

A / (512x512) → Q1, R1

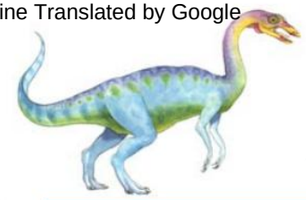
Q1 = deslocamento para o índice externo

R1 é usado da seguinte forma:

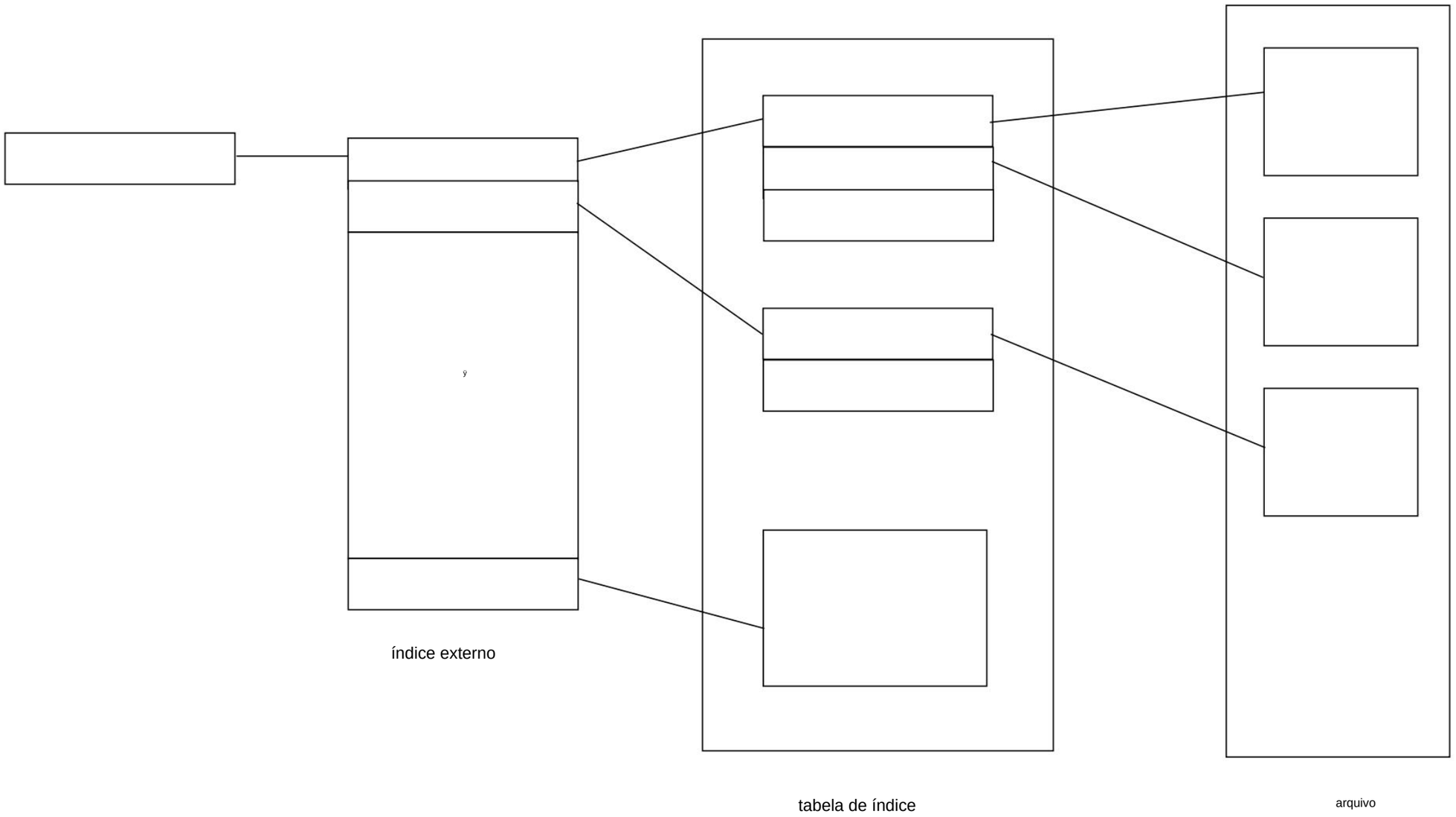
R1 /512 → 2º trimestre, R2

Q2 = deslocamento para o bloco da tabela de índice

Deslocamento R2 no bloco de arquivo:

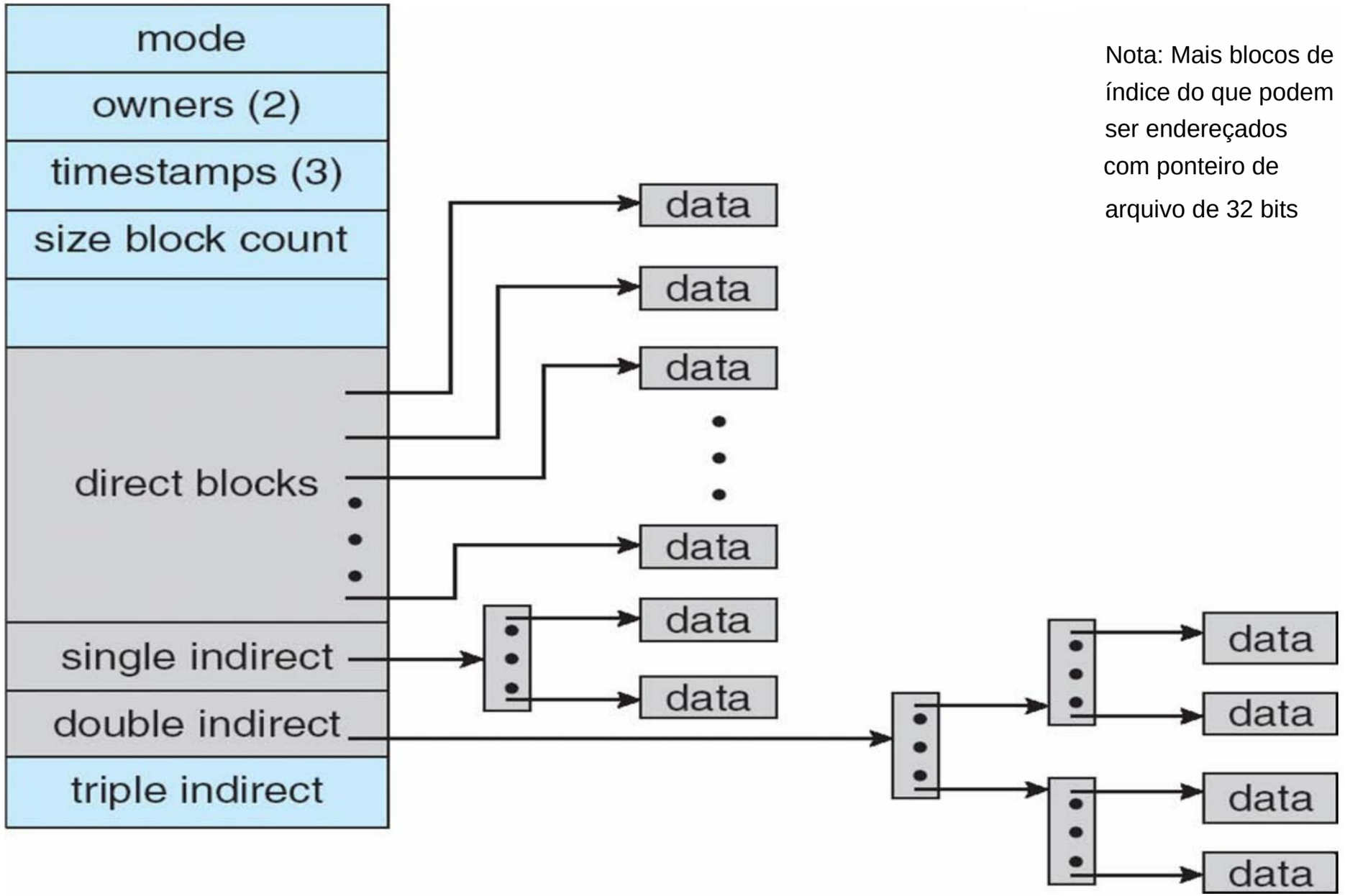


Alocação Indexada – Mapeamento (Cont.)



Esquema combinado: UNIX UFS

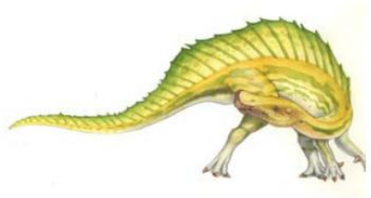
(4K bytes por bloco, endereços de 32 bits)



Nota: Mais blocos de índice do que podem ser endereçados com ponteiro de arquivo de 32 bits

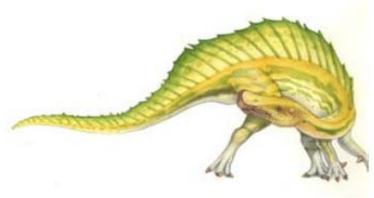
Desempenho

- O melhor método depende do tipo de acesso ao arquivo
  - Contíguo ótimo para sequencial e aleatório
- Vinculado bom para sequencial, não aleatório
- Declare o tipo de acesso na criação -> selecione contíguo ou vinculado
- Indexado mais complexo
- O acesso de bloco único pode exigir 2 leituras de bloco de índice e, em seguida, leitura de bloco de dados
  - O clustering pode ajudar a melhorar o rendimento e reduzir a sobrecarga da CPU



Desempenho (Cont.)

- Adicionar instruções ao caminho de execução para economizar uma E/S de disco é razoável
  - Intel Core i7 Extreme Edition 990x (2011) a 3,46 GHz = 159.000 MIPS
    - http://en.wikipedia.org/wiki/Instructions_per_second
  - Unidade de disco típica a 250 E/S por segundo
    - 159.000 MIPS / 250 = 630 milhões de instruções durante uma E/S de disco
  - Unidades SSD rápidas fornecem 60.000 IOPS
    - 159.000 MIPS / 60.000 = 2,65 milhões de instruções durante uma E/S de disco

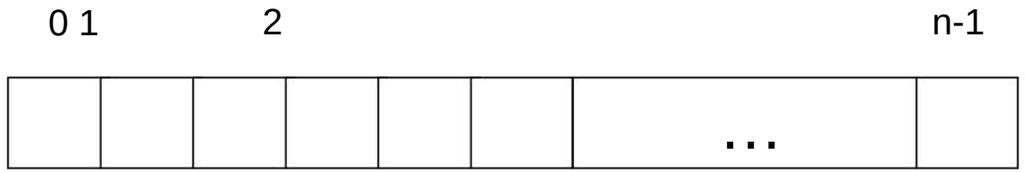


Gerenciamento de Espaço Livre

ÿ O sistema de arquivos mantém **uma lista de espaço livre** para rastrear blocos/clusters disponíveis

ÿ (Usando o termo “bloco” para simplificar)

ÿ **Vetor de bits** ou **mapa de bits** *(n* blocos)



bit[i] = 1 ÿ bloco[i] livre

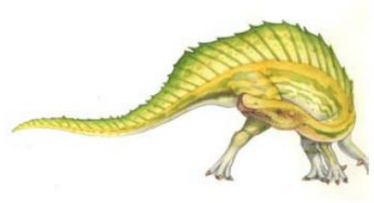
0 ÿ bloco[i] ocupado

ÿÿÿ

Cálculo do número de blocos

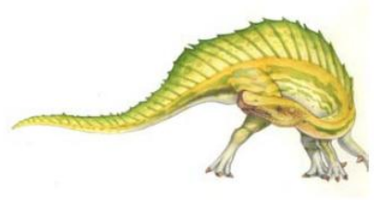
(número de bits por palavra) *
(número de palavras de valor 0) +
deslocamento do primeiro bit 1

As CPUs têm instruções para retornar o deslocamento dentro da palavra do primeiro bit “1”

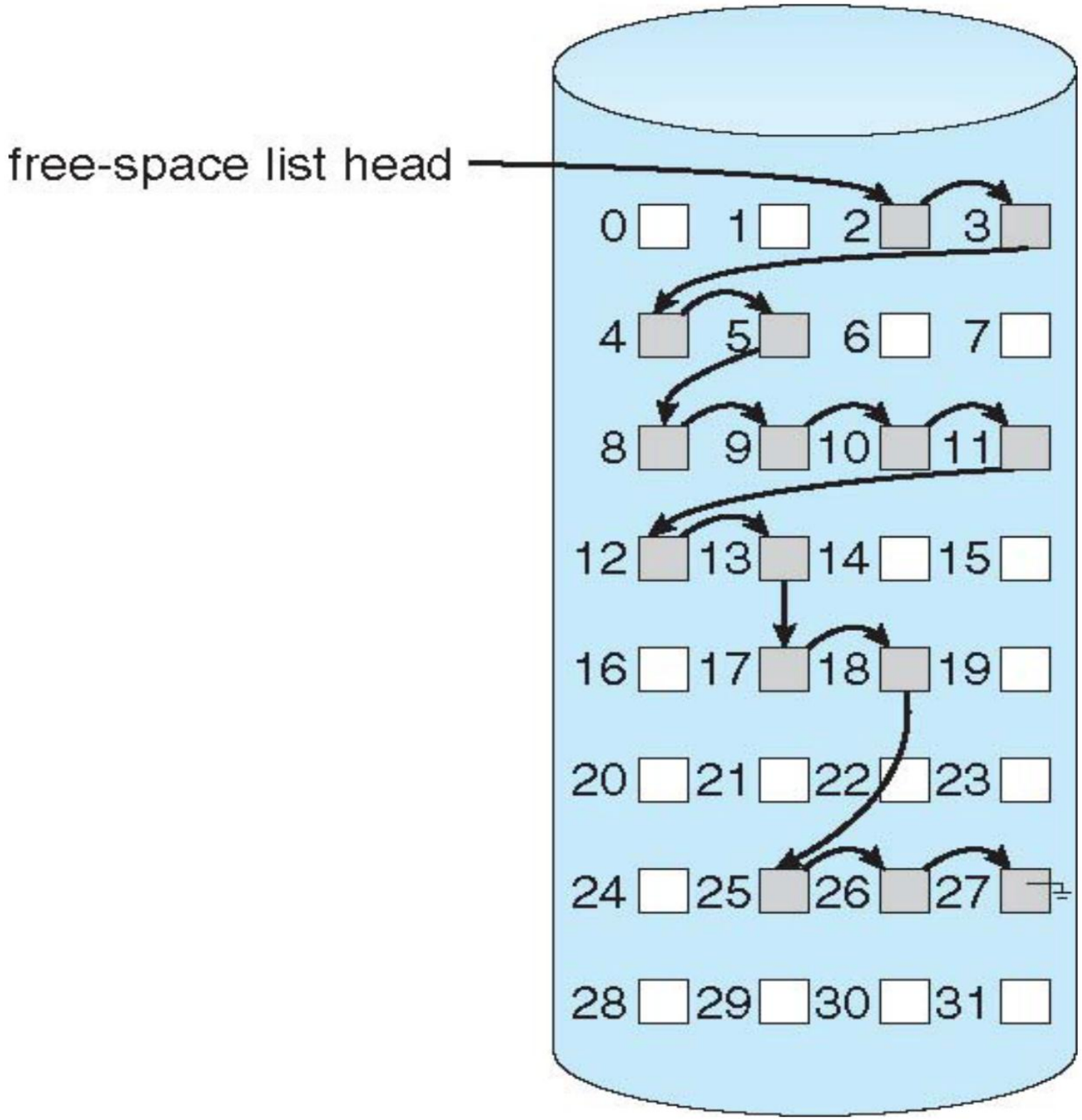


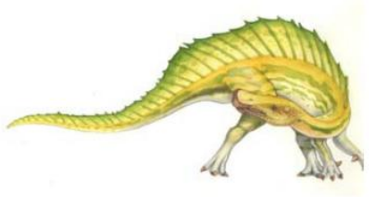
Gerenciamento de Espaço Livre (Cont.)

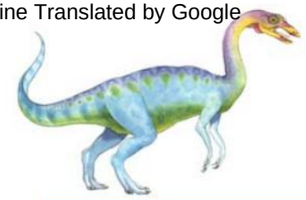
- O mapa de bits requer espaço extra
  - Exemplo:
    - tamanho do bloco = 4 KB = 212 bytes
    - tamanho do disco = 240 bytes (1 terabyte)
    - *n* = 240/212 = 228 bits (ou 256 MB) se
    - clusters de 4 blocos -> 64 MB de memória
- Fácil de obter arquivos contíguos
- Lista vinculada (lista livre)
  - Não é possível obter espaço contíguo facilmente
  - Sem desperdício de espaço
  - Não há necessidade de percorrer a lista inteira (se # blocos livres forem registrados)



Lista de Espaço Livre Vinculado em Disco







Gerenciamento de Espaço Livre (Cont.)

ÿ Agrupamento ÿ

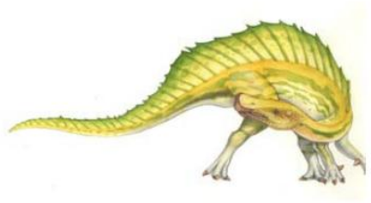
Modifique a lista vinculada para armazenar o endereço dos próximos *n-1* blocos livres no primeiro bloco livre, mais um ponteiro para o próximo bloco que contém ponteiros de bloco livre (como este)

ÿ Contagem

ÿ Como o espaço é frequentemente usado e liberado de forma contígua, com alocação de alocação contígua, extensões ou agrupamento

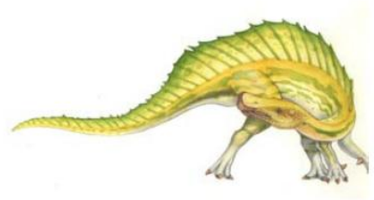
ÿ Mantenha o endereço do primeiro bloco livre e a contagem dos blocos livres seguintes

ÿ A lista de espaços livres tem entradas contendo endereços e contagens



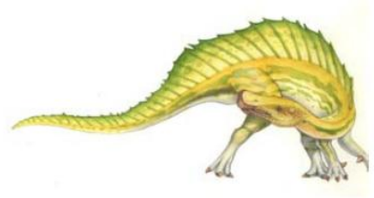
Gerenciamento de Espaço Livre (Cont.)

- Mapas espaciais
  - Usado no ZFS
  - Considere E/S de metadados em sistemas de arquivos muito grandes
    - Estruturas de dados completas, como mapas de bits, não caberiam na memória -> milhares de E/Ss
  - Divide o espaço do dispositivo em unidades **de metaslab** e gerencia metaslabs
    - Um determinado volume pode conter centenas de metaslabs
  - Cada metaslab tem um mapa espacial associado
    - Usa algoritmo de contagem
  - Mas registra em arquivo de log em vez de sistema de arquivo
    - Log de todas as atividades de bloco, em ordem de tempo, em formato de contagem
  - Atividade do Metaslab -> carrega mapa de espaço na memória em estrutura de árvore balanceada, indexado por deslocamento
    - Reproduzir log nessa estrutura
    - Combinar blocos livres contíguos em uma única entrada



Eficiência e Desempenho

- Eficiência dependente de:
  - Algoritmos de alocação de disco e diretório
  - Tipos de dados mantidos na entrada de diretório do arquivo
  - Pré-alocação ou alocação conforme a necessidade de estruturas de metadados
  - Estruturas de dados de tamanho fixo ou variável



Eficiência e Desempenho (Cont.)

- Desempenho
  - Manter dados e metadados próximos **Cache de buffer**
  – seção separada da memória principal para blocos usados com frequência Gravações **síncronas**
  às vezes solicitadas por aplicativos ou necessárias pelo sistema operacional
    - Sem buffer/cache – as gravações devem atingir o disco antes do reconhecimento
    - Escritas **assíncronas** mais comuns, com buffer, mais rápidas
  - **Free-behind** e **read-ahead** – técnicas para otimizar o acesso sequencial
  - Lê frequentemente mais devagar do que escreve

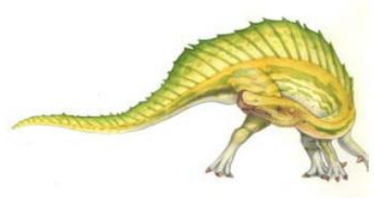
Cache de página

ÿ Um **cache de página** armazena em cache páginas em vez de blocos de disco usando técnicas e endereços de memória virtual

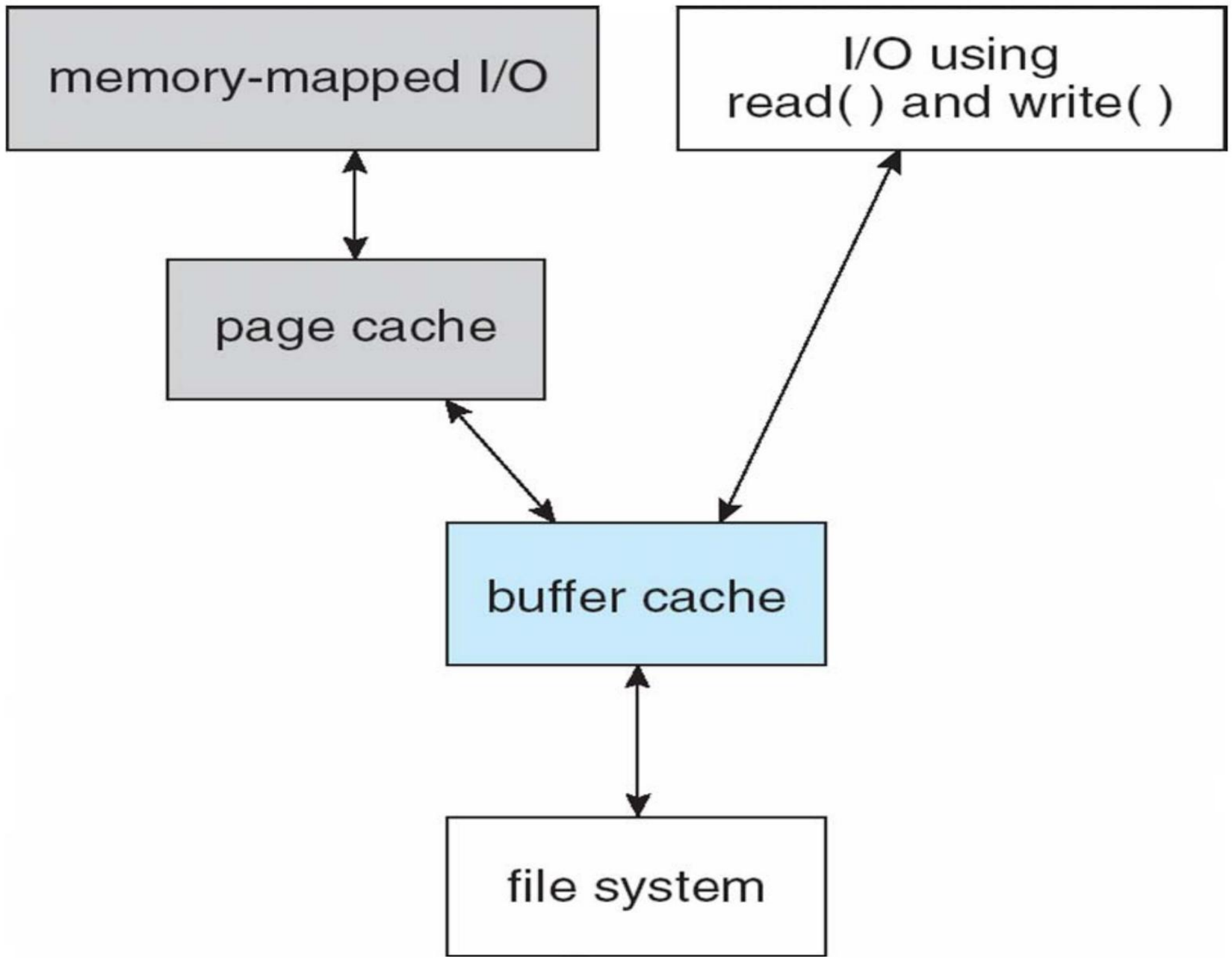
ÿ E/S mapeada na memória usa um cache de página

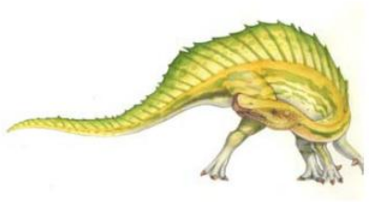
ÿ A rotina de E/S através do sistema de arquivos usa o cache de buffer (disco)

ÿ Isso nos leva à seguinte figura



E/S sem um cache de buffer unificado

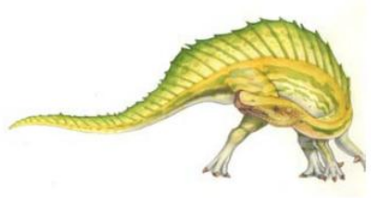




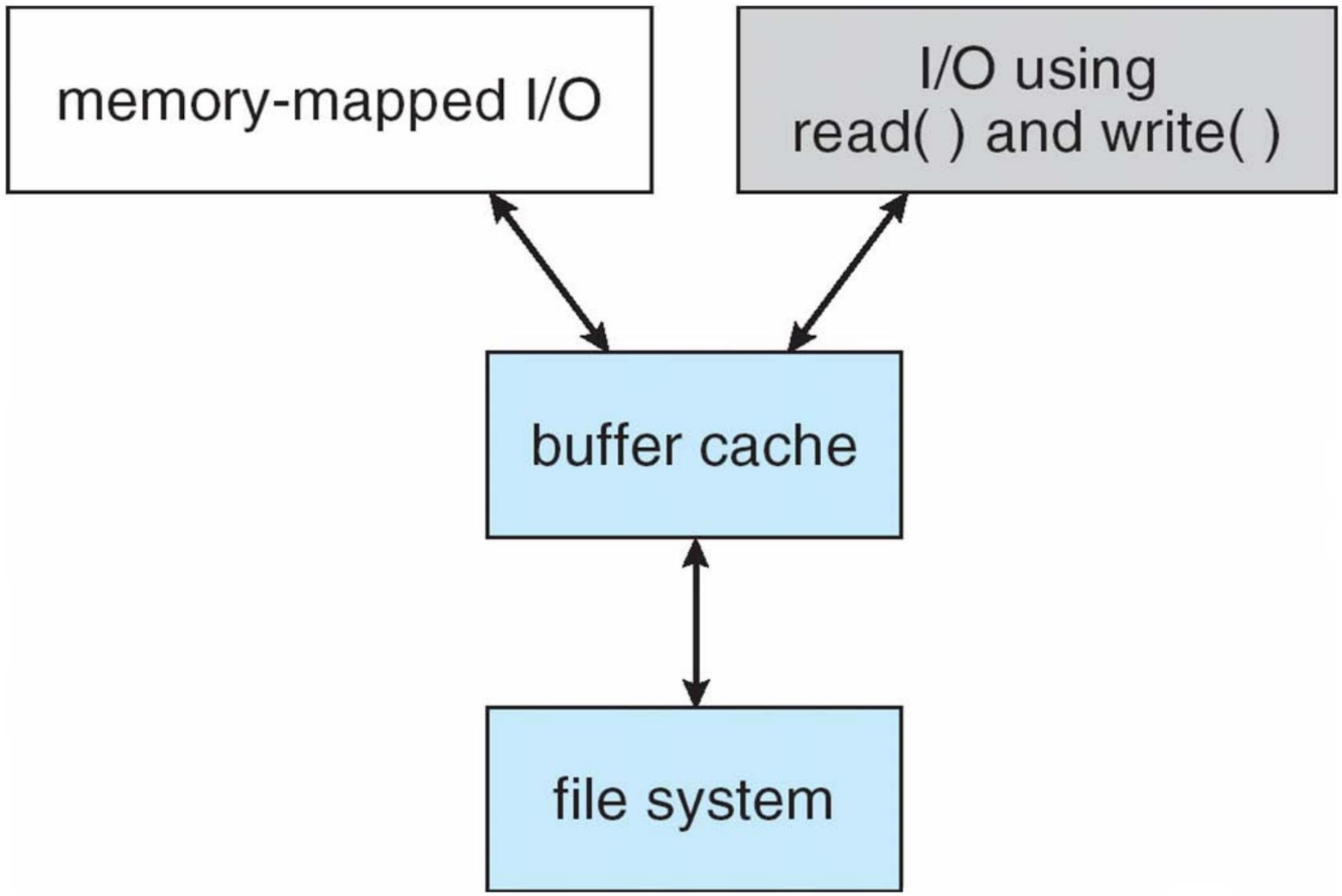
Cache de Buffer Unificado

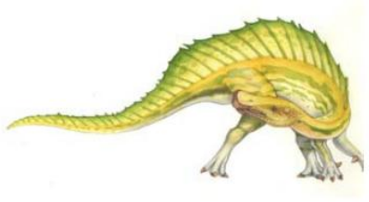
ÿ Um **cache de buffer unificado** usa o mesmo cache de página para armazenar em cache páginas mapeadas na memória e páginas comuns E/S do sistema de arquivos para evitar **cache duplo**

ÿ Mas quais caches têm prioridade e quais algoritmos de substituição usar?



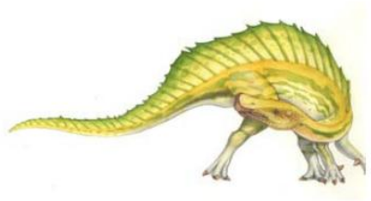
E/S usando um cache de buffer unificado





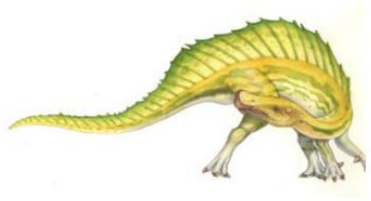
Recuperação

- **Verificação de consistência** – compara dados na estrutura de diretório com blocos de dados no disco e tenta corrigir inconsistências
  - Pode ser lento e às vezes falha
- Use programas de sistema para **fazer backup** de dados do disco para outro dispositivo de armazenamento (fita magnética, outro dispositivo magnético disco, óptico)
- Recupere arquivos ou discos perdidos **restaurando** dados do backup



Sistemas de arquivos estruturados em log

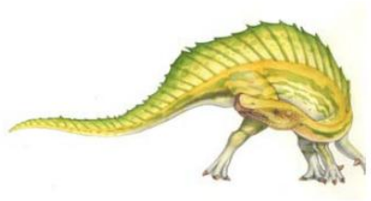
- Os sistemas de arquivos **estruturados em log** (ou **journaling)** registram cada atualização de metadados no sistema de arquivos como um **transação**
- Todas as transações são gravadas em um log
  - Uma transação é considerada confirmada quando é gravada no log (sequencialmente)
  - Às vezes, para um dispositivo ou seção separada do disco
  - No entanto, o sistema de arquivos pode ainda não estar atualizado
- As transações no log são gravadas de forma assíncrona nas estruturas do sistema de arquivos
  - Quando as estruturas do sistema de arquivos são modificadas, a transação é removida do log
- Se o sistema de arquivos travar, todas as transações restantes no log ainda deverão ser executadas
- Recuperação mais rápida de falhas, elimina a possibilidade de inconsistência de metadados

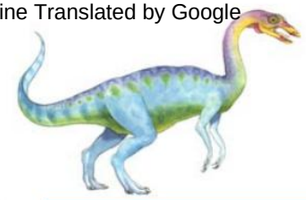


O Sistema de Arquivos de Rede Sun (NFS)

- Uma implementação e uma especificação de um sistema de software para acessar arquivos remotos através de LANs (ou WAN)

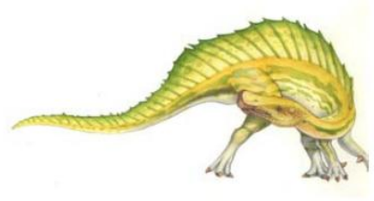
- A implementação faz parte dos sistemas operacionais Solaris e SunOS executados em estações de trabalho Sun usando um protocolo de datagrama não confiável (protocolo UDP/IP e Ethernet





NFS (Continuação)

- Estações de trabalho interconectadas vistas como um conjunto de máquinas independentes com sistemas de arquivos independentes, que permite o compartilhamento entre esses sistemas de arquivos de forma transparente
  - Um diretório remoto é montado sobre um diretório do sistema de arquivos local
    - O diretório montado parece uma subárvore integral do sistema de arquivos local, substituindo a subárvore descendente do diretório local
  - A especificação do diretório remoto para a operação de montagem não é transparente; o nome do host do diretório remoto deve ser fornecido
    - Os arquivos no diretório remoto podem ser acessados de maneira transparente
  - Sujeito à acreditação de direitos de acesso, potencialmente qualquer sistema de arquivos (ou diretório dentro de um sistema de arquivos), pode ser montado remotamente em cima de qualquer diretório local

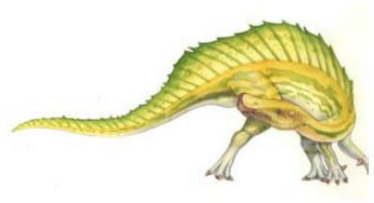


NFS (Continuação)

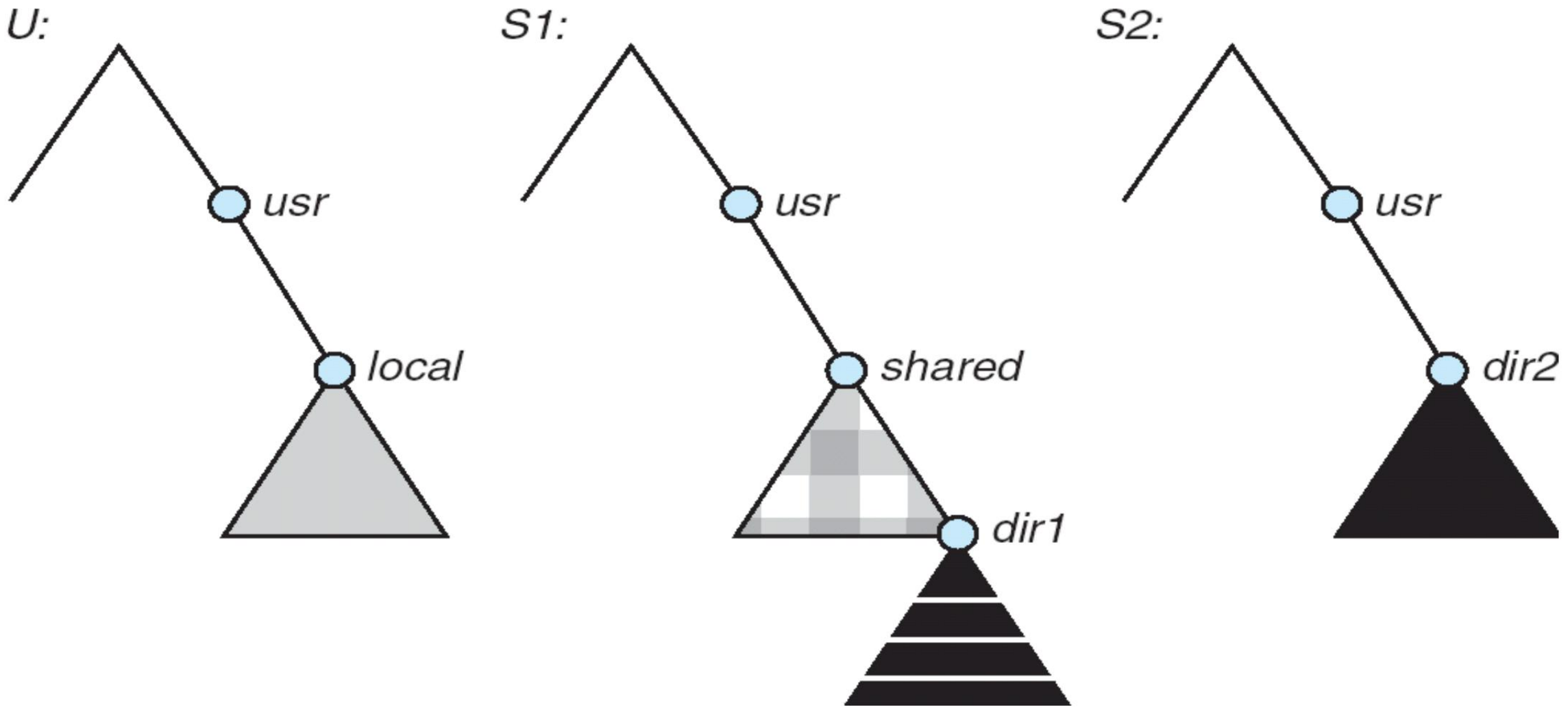
- O NFS foi projetado para operar em um ambiente heterogêneo de diferentes máquinas, sistemas operacionais e arquiteturas de rede; as especificações NFS são independentes dessas mídias

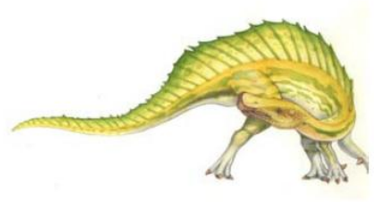
- Essa independência é alcançada por meio do uso de primitivas RPC construídas sobre um banco de dados externo Protocolo de representação (XDR) usado entre duas interfaces independentes de implementação

- A especificação NFS distingue entre os serviços fornecidos por um mecanismo de montagem e os serviços reais serviços de acesso remoto a arquivos

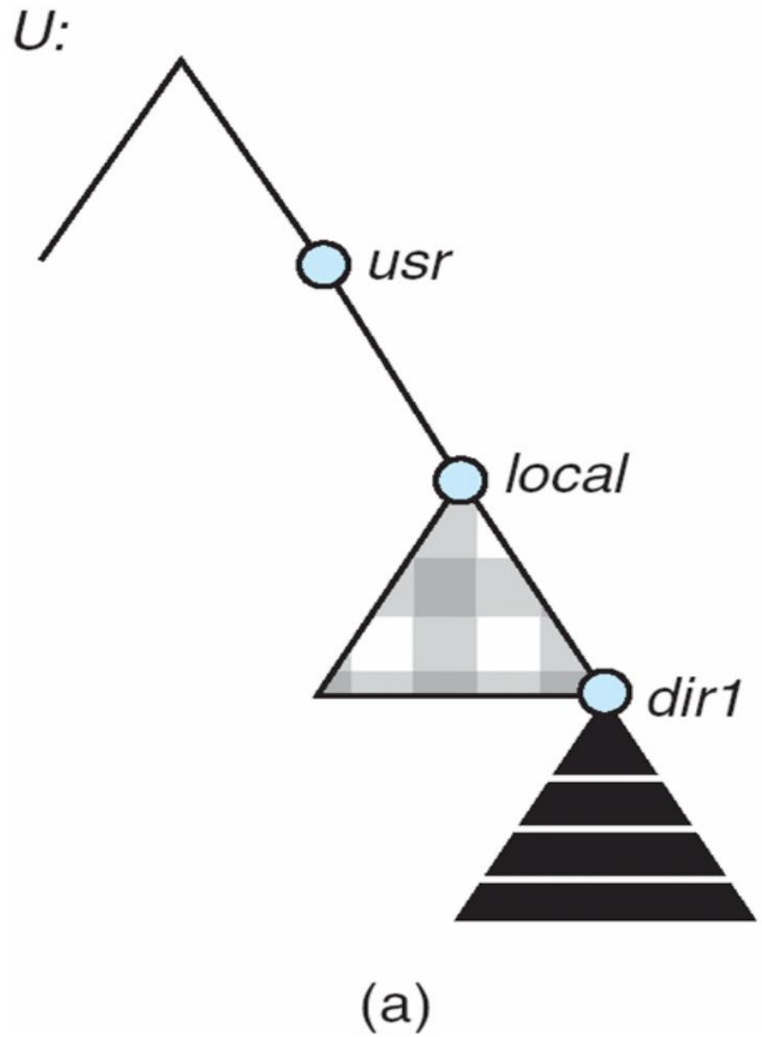


Três sistemas de arquivos independentes

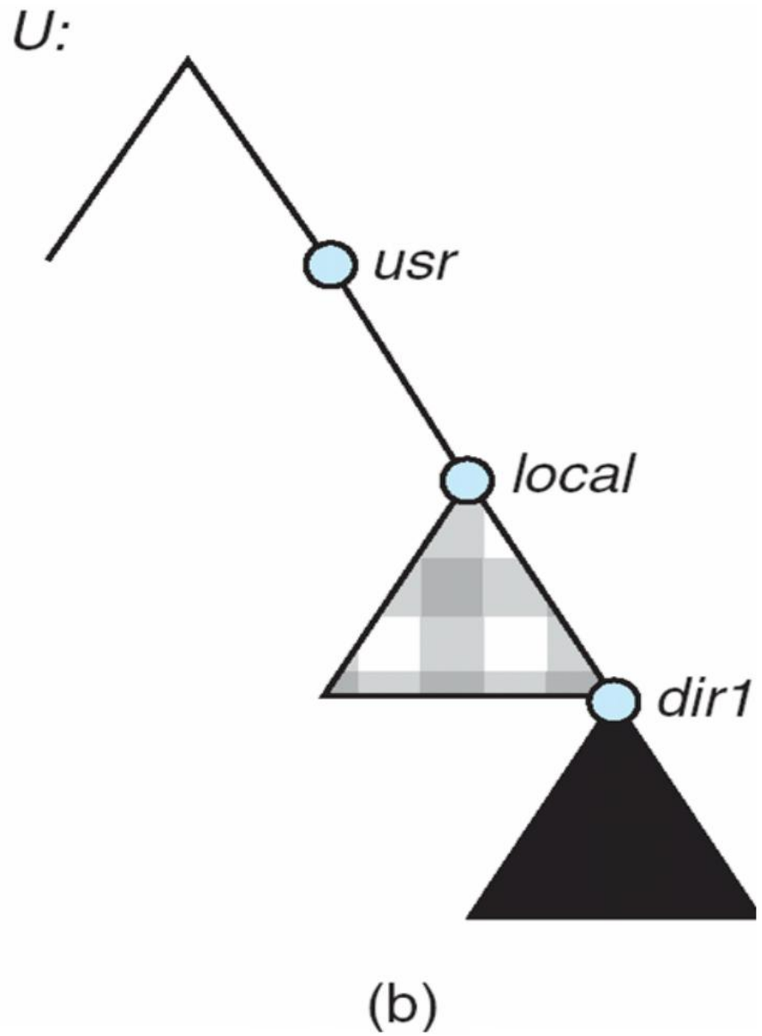




Montagem em NFS



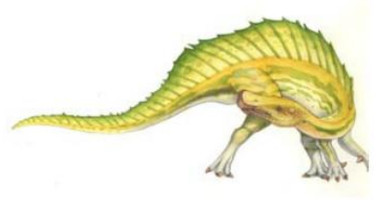
Montagens



Montagens em cascata

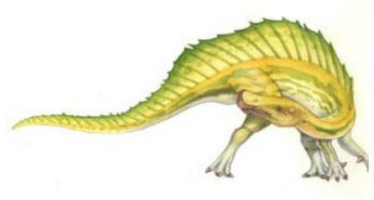
Protocolo de montagem NFS

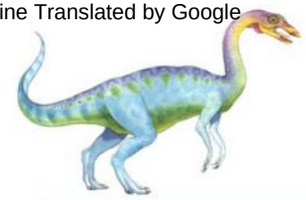
- Estabelece conexão lógica inicial entre servidor e cliente
- A operação de montagem inclui o nome do diretório remoto a ser montado e o nome da máquina servidora que o armazena
  - A solicitação de montagem é mapeada para o RPC correspondente e encaminhada para o servidor de montagem em execução no servidor máquina
  - Lista de exportação – especifica os sistemas de arquivos locais que o servidor exporta para montagem, juntamente com os nomes das máquinas que têm permissão para montá-los
- Após uma solicitação de montagem que esteja em conformidade com sua lista de exportação, o servidor retorna um identificador de arquivo — uma chave para acessos
- Identificador de arquivo – um identificador do sistema de arquivos e um número de inode para identificar o diretório montado dentro do sistema de arquivo exportado
- A operação de montagem altera apenas a visão do usuário e não afeta o lado do servidor



Protocolo NFS

- Fornece um conjunto de chamadas de procedimentos remotos para operações de arquivo remoto. Os procedimentos oferecem suporte ao seguinte operações:
  - procurar um arquivo dentro de um diretório
  - ler um conjunto de entradas de diretório
  - manipular links e diretórios acessar atributos de arquivo
  - leitura e escrita de arquivos
- Os servidores NFS não têm **estado;** cada solicitação deve fornecer um conjunto completo de argumentos (o NFS V4 está chegando disponível – muito diferente, com estado)
- Os dados modificados devem ser confirmados no disco do servidor antes que os resultados sejam retornados ao cliente (perda vantagens do cache)
- O protocolo NFS não fornece mecanismos de controle de simultaneidade





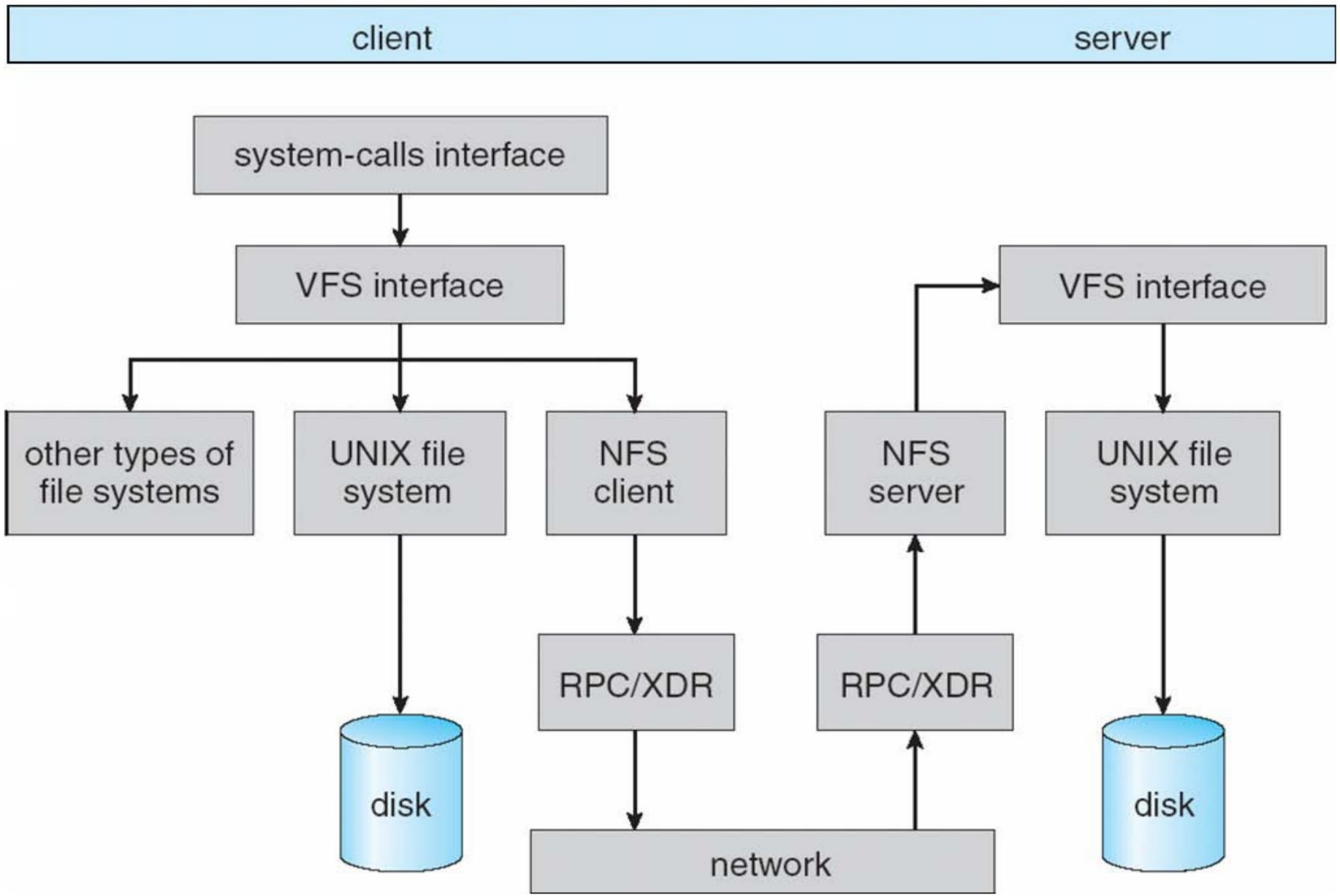
Três camadas principais da arquitetura NFS

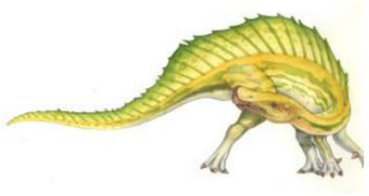
- Interface do sistema de arquivos UNIX (com base nas chamadas **de abertura, leitura, gravação** e **fechamento** e **descritores de arquivo)**

- Camada *do Sistema de Arquivos Virtual* (VFS) – distingue arquivos locais de remotos, e os arquivos locais são ainda mais diferenciados de acordo com seus tipos de sistema de arquivos
  - O VFS ativa operações específicas do sistema de arquivos para lidar com solicitações locais de acordo com seus tipos de sistema
  - Chama os procedimentos do protocolo NFS para solicitações remotas

- Camada de serviço NFS – camada inferior da arquitetura
  - Implementa o protocolo NFS

Visão esquemática da arquitetura NFS

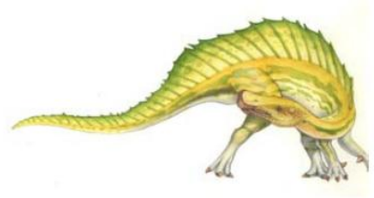




Tradução de nome de caminho NFS

ÿ Executado dividindo o caminho em nomes de componentes e executando uma chamada de pesquisa NFS separada para cada par de nome de componente e diretório vnode

ÿ Para tornar a pesquisa mais rápida, um cache de pesquisa de nome de diretório no lado do cliente mantém os vnodes para acesso remoto. nomes de diretório



Operações remotas NFS

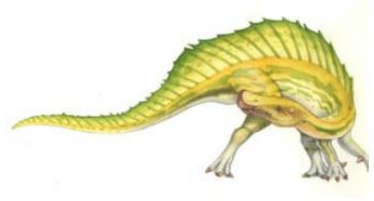
- Correspondência quase um-para-um entre chamadas regulares do sistema UNIX e os RPCs do protocolo NFS
  (exceto abrir e fechar arquivos)

- O NFS adere ao paradigma de serviço remoto, mas emprega técnicas de buffer e cache para fins de
  de desempenho

- Cache de blocos de arquivo – quando um arquivo é aberto, o kernel verifica com o servidor remoto se deve buscá-lo ou
  revalidar os atributos armazenados em cache
  - Os blocos de arquivos em cache são usados somente se os atributos em cache correspondentes estiverem atualizados

- Cache de atributos de arquivo – o cache de atributos é atualizado sempre que novos atributos chegam do servidor

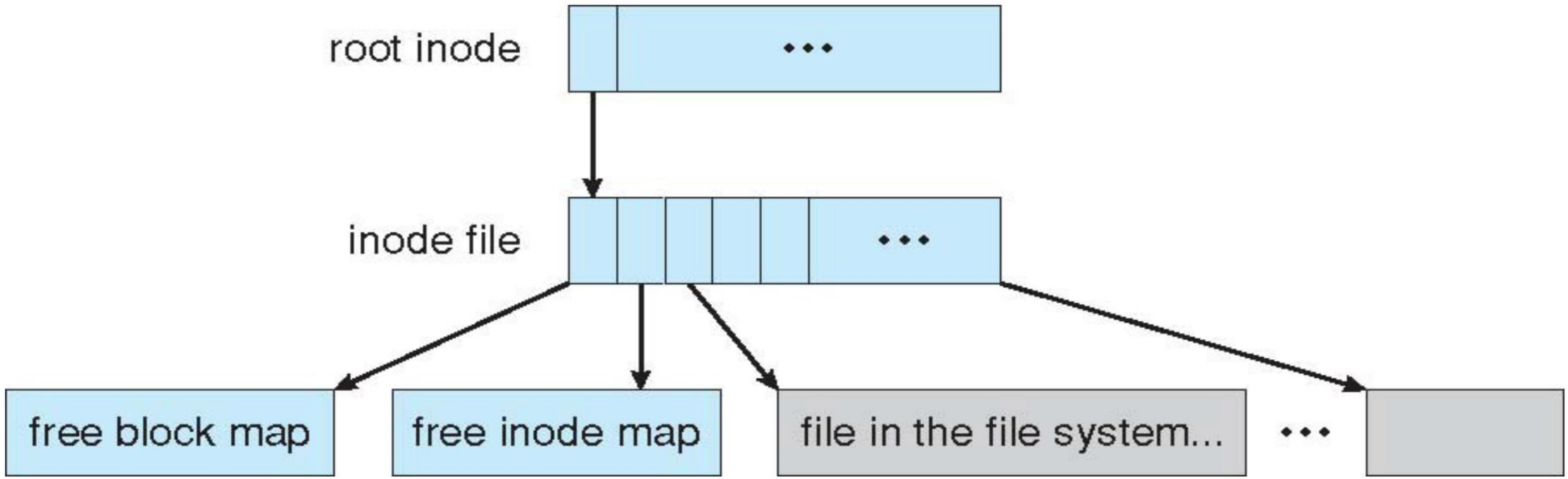
- Os clientes não liberam blocos de gravação atrasada até que o servidor confirme que os dados foram gravados no disco

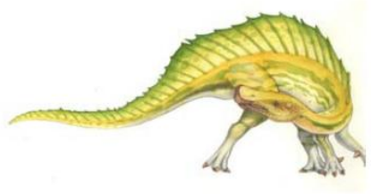


Exemplo: Sistema de arquivos WAFL

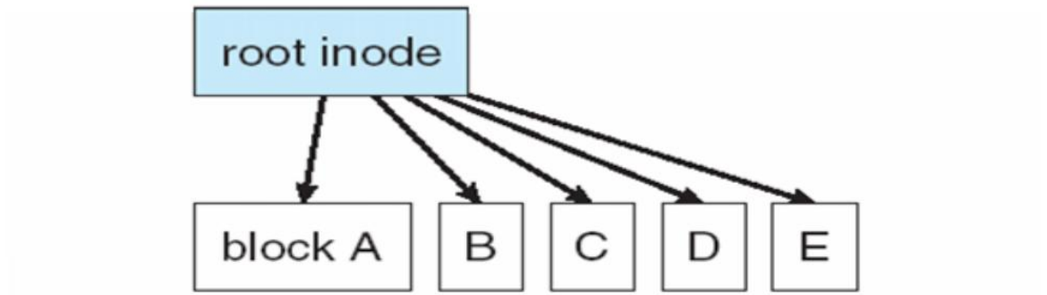
- Usado em Network Appliance “Filers” – appliances de sistema de arquivos distribuídos
  - “Layout de arquivo para gravação em qualquer lugar”
- Atende NFS, CIFS, http, ftp
- E/S aleatória otimizada, gravação otimizada
  - NVRAM para cache de gravação
- Semelhante ao Berkeley Fast File System, com extensas modificações

O layout do arquivo WAFL

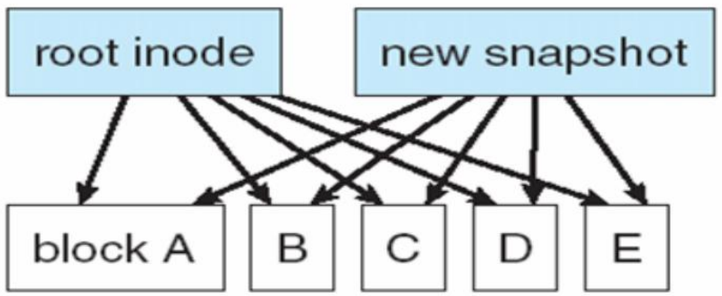




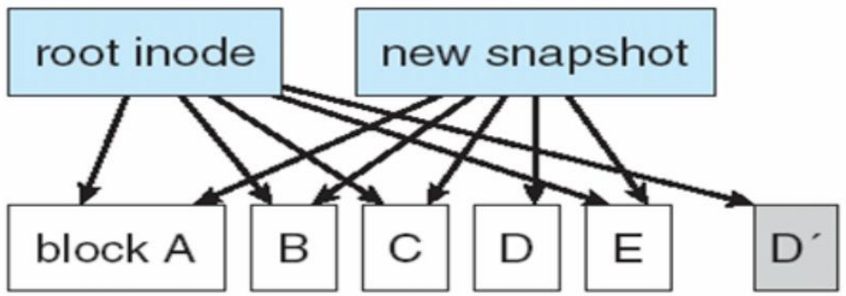
Instantâneos em WAFL



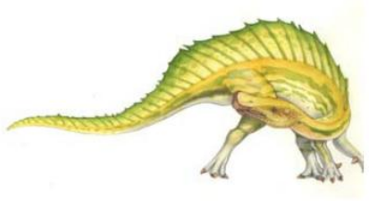
(a) Before a snapshot.



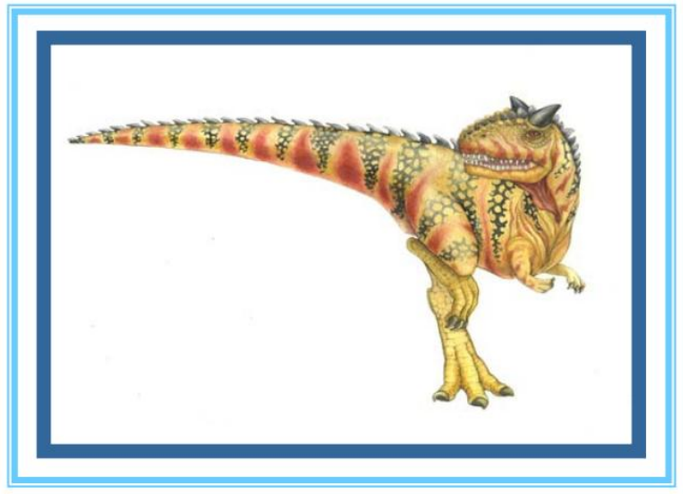
(b) After a snapshot, before any blocks change.



(c) After block D has changed to D´.



Fim do Capítulo 10



Gerenciamento de Espaço Livre (Cont.)

- Necessidade de proteger:
  - Ponteiro para lista livre
  - Mapa de bits
    - Deve ser mantido no disco
    - A cópia na memória e no disco pode ser diferente Não é possível permitir que block[i] tenha uma situação em que bit[i] = 1 na memória e bit[i] = 0 no disco
  - Solução:
    - Definir bit[i] = 1 no disco
    - Alocar block[i]
      Definir bit[i] = 1 na memória