Cap.4

1) (Tan-2003) Um sistema de computador tem espaço suficiente para armazenar 4 programas em sua memória principal. Esses programas ficam ociosos durante metade de seu tempo, esperando por E/S. Que fração do tempo da CPU é desperdiçada?

A chance que todos os quatro processos estão ociosos é 1/16, então o tempo ocioso de CPU é 1/16.

2) (Tan-2003) Considere um sistema de troca de processos entre a memória e o disco no qual a memória é constituída dos seguintes tamanhos de lacunas em ordem na memória: 10kB, 4kB, 20kB, 18kB, 7kB, 9kB, 12kB e 15kB. Qual lacuna é tomada pelas solicitações sucessivas do segmento de:
a) 12kB
b) 10kB
c) 9kB
Para o first fit? Repita agora para o best fit, worst fit e next fit.

First fit toma 20 KB, 10 KB, 18 KB. Best fit toma 12 KB, 10 KB, e 9 KB. Worst fit toma 20 KB, 18 KB, e 15 KB. Next fit toma 20 KB, 18 KB, e 9KB.

3) (Tan-2003) Qual é a diferença entre um endereço físico e um virtual?

Memória real usa endereços físicos. Estes são os números que os chips de memória reagem no barramento de endereços Os endereços virtuais são os endereços lógicos que se referem ao espaço de endereço do processo. Assim, uma máquina com uma palavra de 16 bits pode gerar endereços virtuais até 64K, independentemente de a máquina ter mais ou menos memória de 64 KB.

4) (Tan-2003) Para cada um dos seguintes endereços virtuais, calcule o numero da pagina virtual e o deslocamento para uma pagina de 4kB e para uma pagina de 8kB: 2000, 32768, 60000.

Para uma página de tamanho 4 KB (página, offset) os pares são (4, 3616), (8, 0) e (14,2656). Para uma página de tamanho 8KB eles são (2, 3616), (4, 0), (7, 2656).

5) (Tan-2003) Usando a tabela de paginas da Figura 4.10, de o endereço físico correspondente a cada um dos seguintes endereços virtuais:
a) 20
b) 4100
c) 8300

(a) 8212 (b) 4100 (c) 24684

6) (Tan-2003) Uma maquina tem endereçamento virtual de 48 bits e um endereçamento físico de 32 bits. As paginas são de 8 KB. Quantas entradas são necessárias para a tabela de paginas?

Com páginas de 8 KB e um espaço de endereçamento virtual de 48 bits, o número de páginas virtual é 2^48/2^13, que é de 2^35 (cerca de 34 bilhões).

7) (Tan-2003) Se o algoritmo de substituição FIFO é usado com quatro molduras de pagina e oito paginas virtuais, quantas faltas de pagina ocorrerão com a cadeia de referencias 0172327103 se os quatro quadros estão inicialmente vazios? Agora repita este problema para o MRU.

As molduras de página para FIFO são as seguintes: x0172333300 xx017222233 xxx01777722 xxxx0111177 As molduras de página para LRU são as seguintes: x0172327103 xx017232710 xxx01773271 xxxx0111327 FIFO rende seis falhas de página; LRU rende 7.

8) (Tan-2003) Observe a seqüência de paginas da Figura 4.16(b). Suponha que os bits R, para as paginas de B a A, sejam 11011011, respectivamente. Quais paginas serão removidas pelo algoritmo segunda chance?

A primeira página com um bit 0 será escolhido, neste caso, D

9) (Tan-2003) Suponha tau=400 na Figura 4.21. Qual pagina será removida?

A primeira página com R = 0 e idade> τ será escolhido. Uma vez que a verificação começa no fundo, a primeira página (1620) será removida.

10) (Tan-2003) Um computador tem quatro molduras de pagina. O tempo de carregamento de pagina na memória, o instante do ultimo acesso e os bits R e M para cada pagina são mostrados a seguir (os tempos estão em tiques de relógio):

Pagina	Carregado	R	M	Ultima ref.
0	126	1	0	280
1	230	0	01	265
2	140	0	0	270
3	110	1	1	285

a) Qual pagina será trocada pelo NUR?
b) Qual pagina será trocada pelo FIFO?
c) Qual pagina será trocada

pelo MRU?
d) Qual pagina será trocada pelo segunda chance?

NRU remove página 2. FIFO remove página 3. LRU remove página 1. Segunda chance remove página 2.

11) (Tan-2003) tentando encontrar meios de reduzir a quantidade de área de troca necessária em seus sistemas operacionais. Ideia: não perder tempo com o salvamento do texto do programa na área de troca, mas, simplesmente, paginá-lo diretamente do arquivo binário quando necessário. Se é que isso é possível, sob quais condições essa idéia funciona para o código do programa? E sob quais condições ela funciona para os dados?

Ele trabalha para o programa se o programa não pode ser modificado. Ele trabalha para o dados se os dados não podem ser modificados. No entanto, é comum que o programa não pode ser modificado e extremamente raro que os dados não podem ser modificados. Se a área de dados sobre o arquivo binário foram substituídos por páginas atualizadas, a próxima vez que o programa foi iniciado, ele não teria os dados originais.

12) (Tan-2003) Explique as diferenças entre fragmentação interna e externa. Qual delas ocorre em sistemas de paginação? E qual ocorre em sistemas de segmentação pura?

Fragmentação interna ocorre quando a última unidade de alocação não é completa.
Fragmentação externa ocorre quando o espaço é desperdiçado entre duas unidades de alocação.
Em um sistema de paginação, o espaço perdido na última página está perdido para a fragmentação interna.
Em um sistema de segmentação puro, algum espaço é invariavelmente perdido entre os segmentos. Isto é devido à fragmentação externa.

Cap.5

1) (Tan-2003) Suponha que um computador possa ler ou escrever uma palavra de memória em 10ns. Suponha também que, quando uma interrupção ocorre, todos os 32 registradores da CPU mais o contador de programa e a PSW são colocados na pilha. Qual é o número máximo de interrupções por segundo que essa máquina pode processar?
Cada interrupção deve escrever e ler 34 palavras na memória. Isso utiliza 10ns*68 = 680ns; portanto o número máximo de interrupções será 1/340ns = 1.470.588 interrupções.

2) (Tan-2003) O que é independência de dispositivo? Por que esta característica é desejável?
Esse conceito propõe que é possível desenvolver um programa aptos a acessar todo tipo de dispositivo de E/S sem se preocupar com o seu tipo. É uma característica desejável, pois um programa poderia acessar tanto um disco rígido, cd-rom ou disquete sem necessitar alterações em seu código para cada dispositivo diferente.

3) O que são dispositivos de bloco e de caracter? Dê exemplos de cada um.
Dispositivos de bloco são aqueles que armazenam dados em forma de blocos, cada qual com seu endereço. O acesso a esses blocos são independentes entre si. Exemplos: disco rigido, pen-drive, cd-rom
Dispositivos de caracter são aqueles que enviam e recebem fluxos de caracter, sem endereçamento e estrutura de blocos. Exemplos: impressoras, mouse, interfaces de rede.

4) Diferencie E/S separada da memória de E/S mapeada em memória. Cite 2 vantagens e 2 desvantagens do segundo modelo (mapeada). Qual dos modelos é utilizado no processador MIPS (na máquina virtual uMPS)?
E/S separada da memória: os espaços de endereçamento de memória e E/S são diferentes.
E/S mapeada na memória: todos os registradores de controle são mapeados mapeados no espaço de endereçamento da memória, sendo que cada registrador possui um endereço na memória ao qual nenhuma memória é acessada.
Vantagens:

- com E/S mapeado em memória, um driver de dispositivo pode ser implementado todo em linguagem C; caso fosse em memória separada, esse driver teria de possuir diretivas em assembly.

- Não é preciso um controlo rigoroso sobre os processos de usuário quanto ao acesso a E/S. O sistema operacional apenas deve deixar de mapear os endereços dos registradores de controle na memória virtual.

Desvantagens

- hardware deve ser equipado para desabilitar seletivamente a cache, para evitar problemas com os registros de controle mapeados em memória. Isso requer uma maior complexidade tanto por parte de hardware, quanto por software.
- Dispositivo de E/S mapeados

em memória não conseguem reconhecer os endereços de memória quando estes são lançados no barramento de memória, de forma que eles não consigam responder.

5) Explique E/S programada, orientada à interrupção e DMA.

E/S programada: forma mais simples de E/S em que a CPU realiza todo o processo de E/S. Porém, esse método prende toda a utilização da CPU.

E/S orientada à interrupção: utiliza interrupções para que a CPU fique livre enquanto aguarda o fim da E/S. No entanto, para cada acesso terá de ser realizado uma interrupção, fato que desperdiça tempo e processamento da CPU.

E/S usando DMA: o DMA realiza E/S programada no lugar da CPU, o que a libera para executar demais funções. Assim, o número de interrupções fica reduzido. Geralmente um DMA é muito mais lento que a CPU, porém sua viabilidade ainda é alta.

Quando multiplas interrupcoes de diferentes dispositivos ocorrem ao mesmo tempo, um esquema de prioridades deve ser utilizado para determinar a ordem na qual as interrupções devem ser atendidas. Discuta quais aspectos devem ser considerados na atribuição destas prioridades.

Há técnicas de identificação de dispositivos que disparam interrupções. Essas técnicas permitem criar um esquema de prioridade. No caso de múltiplas linhas, a CPU seleciona, apenas, a linha de prioridade com prioridade mais elevada. Com sondagem por software, a ordem pela qual os dispositivos são sondados determina sua prioridade. Similarmente, a ordem dos módulos numa cadeia prioriatizada determina a sua prioridade. Finalmente, a arbitragem do barramento pode empregar um esquema de prioridade.

Tipicamente, ao termino de uma operacao de I/O, uma única interrupcao e levantada e é tratada apropriadamente pelo processador. Em certos casos, porem, o codigo de tratamento da interrupcao pode ser dividido em 2 partes. A primeira parte executa imediatamente apos o termino da operacao de I/O, e esta parte escalona uma segunda interrupcao para a segunda parte ser executada mais tarde. Qual o proposito de se utilizar esta estratégia no projeto de tratadores de interrupcao? Cite um SO que se utiliza desta estrategia.

1 - Precisa-se da habilidade de adiar o tratamento de interrupções durante o processamento crítico.

2 – Precisa-se de uma maneira eficiente de “despachar” para o tratador de interrupção de dispositivo apropriado sem ter que, primeiro, verificar todos os dispositivos para ver qual disparou a interrupção.

3 – Precisa-se de interrupções multinível, para que o SO possa distinguir entre interrupções de alta e baixa prioridade para poder responder com o devido grau de urgência.

Como a DMA aumenta a concorrencia do sistema? Como ela complica o projeto do hardware?

O DMA aumenta a concorrência do sistema permitindo que a CPU realize tarefas enquanto o sistema de DMA transfere dados por meio dos barramentos do sistema de memória. O projeto do hardware é complicado porque o DMA precisa ser integrado ao sistema, e o sistema precisa permitir que o controlador de DMA se já um controlador do barramento. O roubo de ciclos também pode ser necessário para permitir que a CPU e o controlador de dados compartilhem o uso do barramento de memória.

6) O que é uma interrupção precisa? Quais são suas características?

Interrupção precisa é aquela em que deixa a máquina em um estado bem definido. Características:

- O contador de programa é salvo em lugar conhecido;
- todas as instruções anteriores ao do PC foram devidamente executadas;
- nenhuma instrução posterior ao PC foi executada;
- estado de execução da instrução apontada pelo PC é conhecido.

7) Explique as 4 camadas do software de E/S.

Tratadores de interrupção:quando ocorre uma interrupção, a rotina de tratamento de interrupção faz o necessário para tratar a interrupção e depois pode desbloquear o driver que a chamou.

Drivers dos dispositivos: cada dispositivo de E/S ligado à CPU necessita de uma rotina para sua utilização. Essa rotina, ou código, é o driver do dispositivo, cuja função é implementar a comunicação do subsistema de E/S com os dispositivos, por meio de controladores.

Software de E/S independente de dispositivo: executar as funções comuns de E/S para todos os dispositivos e fornecer uma interface uniforme em nível de usuário: (1) uniformizar interfaces para drivers de dispositivos; (2) armazenar

em buffer; (3) reportar erros; (4) alocar e liberar dispositivos dedicados; (5)providenciar um tamanho de bloco independente do dispositivo.

Software de E/S do espaço do usuário:bibliotecas ligadas aos programas de usuário e até mesmo programas que executam fora do núcleo.

8) (Tan-2003) Em qual das quatro camadas do software de E/S se realiza cada uma das seguintes atividades:

a) Calcular a trilha, setor e cabeçote para uma leitura do disco.

Drivers dos dispositivos

b) Escrever comandos no registradores do dispositivo.

Drivers dos dispositivos

c) Verificar se o usuário tem permissão para usar o dispositivo.

Software de E/S independente do dispositivo

d) Converter inteiros binários em ASCII para impressão.

Software de E/S do espaço de usuário

9) (Tan-2003) Por que os arquivos de saída para a impressora são normalmente colocados em um spool no disco antes de serem impressos?

Para se evitar que um processo abra um arquivo especial de caracteres para a impressora e impeça que os demais processos possam imprimir seus arquivos. Assim, é criado um processo especial(daemon) e um diretório especial (diretório de spool) que, para se imprimir um arquivo, esse processo gera primeio todo o arquivo a ser impresso e o coloca no diretório de spool. O Daemon fica encarregado de imprimir os arquivos

10) (Tan-2003) Em um certo computador, o tratador de interrupção de relógio requer 2ms (incluindo a troca de processos) para cada tique do relógio. O relógio trabalha a 60Hz, Qual fração da CPU é dedicada ao relógio?

2ms*60 = 120ms

fracao = 120ms/1s = 0.12 = 12%

11) (Tan-2003) Muitas versões do Unix usam um inteiro de 32 bits sem sinal para manter o controle da hora como o número de segundos desde a origem do tempo. Quando esses sistemas vão zerar novamente o horário (ano e mês)? Podemos esperar que isso realmente ocorra?

2^32 bits equivalem a 4294967292s = 138 anos e 1 mês, aproximadamente.

É provável que isso não ocorra, uma vez que já existem (ou já está bem próximo de lançar) versões do unix de 64 bits.

12) Explique o algoritmo do elevador. Para que serve?

Funciona da mesma forma que um elevador: mantém o mesmo sentido enquanto houver alguma requisição em algum andar. Esse algoritmo é muito utilizado nos processos de escalonamento de solicitações em discos rígidos.

13) Explique como se dá o processo de boot (carga do sistema operacional) em um PC.

A solução para o paradoxo está na utilização de um pequeno e especial programa, chamado *sistema de iniciação, boot loader* ou *bootstrap*. Este programa não tem a completa funcionalidade de um sistema operacional, mas é especialmente construído para que seja capaz de carregar um outro programa para permitir a iniciação do sistema operacional. Freqüentemente, boot loaders de múltiplos estágios são usados, neste caso vários pequenos programas se complementam em seqüência, até que o último deles carregue o sistema operacional.

Os primeiros computadores programáveis tinham chaves no painel frontal para permitir ao operador colocar o sistema de iniciação na memória antes de iniciar a CPU. Este poderia então ler o sistema operacional de um meio de armazenamento externo como uma fita de papel.

Nos computadores modernos o processo de iniciação começa com a execução pela CPU de um programa contido na memória ROM (o BIOS do IBM PC) em um endereço predefinido (a CPU é programada para executar este programa depois de um reset automaticamente). Este programa contém funcionalidades rudimentares para procurar por dispositivos que podem conter um sistema operacional e que são, portanto, passíveis de participar de um boot. Definido o dispositivo é carregado um pequeno programa de uma seção especial deste.

O pequeno programa normalmente não é o sistema operacional, mas apenas um segundo estágio do sistema de inicialização, assim como o Lilo ou o Grub. Ele será então capaz de carregar o sistema operacional apropriado, e finalmente transferir a execução para ele. O sistema irá inicializar, e deve carregar *drivers* de dispositivos (device drivers) e outros programas que são necessários para a operação normal de um sistema operacional.

O processo de inicialização é considerado completo quando o computador está pronto para ser operado pelo usuário. Computadores pessoais modernos tipicamente levam

cerca de um minuto para executar o processo de inicialização (deste tempo, cerca de 15 segundos são devidos a cada chamada do processo de inicialização, e o restante para carregar o sistema operacional). No entanto, sistemas mais complexos como servidores podem levar vários minutos para terminar o processo de inicialização e carregar todos os serviços. Para garantir maior disponibilidade, estes iniciam certos serviços preferencialmente antes de outros.
Muitos sistemas embutidos, ou embedded systems, podem iniciar instantaneamente - por exemplo, esperar um minuto para uma televisão ligar é inaceitável. Assim, estes sistemas têm seu sistema operacional inteiro na ROM ou na memória flash, podendo executá-lo diretamente.
Em computação, uma **seqüência de inicialização** compreende toda e qualquer operação que um computador executa, após ter sido ligado, visando carregar o sistema operacional.

14) Explique o que é, e para que serve, o entrelaçamento utilizado na formatação de um disco.
O entrelaçamento serve para otimizar a leitura de um disco, no sentido de minimizar seu tempo de processamento. Após a leitura do primeiro setor de disco, os dados são passados para um buffer e paralelamente a isso, o cabeçote já procura se alinhar com o próximo setor. Se não tivéssemos o entrelaçamento (os setores fossem alinhados continuamente), o disco teria de rotacionar quase uma volta completa até encontrar o próximo setor, o que gera um gasto temporal elevado. Já com o entrelaçamento há tempo do buffer copiar seus dados para a memória principal e já em seguida capturar os dados, já em seqüência, dos demais setores. Caso o buffer necessite mais tempo para transferência, pode-se utilizar o entrelaçamento duplo.

Sem entrelaçamento 1|2|3|4|5|6|7|8

Entrelaçamento simples 1|5|2|6|3|7|4|8

Entrelaçamento duplo 1|4|7|2|5|8|3|6

Descreva 3 circunstancias nas quais se deve utilizar I/O bloqueante. Descreva 3 circunstancias nas quais se deve utilizar I/O nao-bloqueante. Por que simplesmente nao se implementa I/O nao-boqueante e coloca-se os processos em espera ociosa (busy-wait) ate que seus dispositivos estejam prontos?
Geralmente, o bloqueio de I/O é apropriado quando o processo só estiver esperando por um evento especifico. Alguns exemplos incluem um disco, fita ou teclado lido por uma aplicação. O I/O não bloqueante é útil quando o I/O pode vir de mais de uma origem e a ordem da chegada do I/O não é predeterminada. Alguns exemplos incluem daemons de rede escutando mais de um socket de rede, gerenciadores de janela que aceitam movimento do mouse e entrada do teclado, e programas de gerenciamento de I/O, como um comando copy que copia dados entre dispositivos de I/O. No ultimo caso, o programa poderia otimizar seu desempenho colocando a entrada e a saída do buffer e usando o I/O não-bloqueante para manter os dois dispositivos totalmente ocupados. O I/O não-bloqueante é mais complicado para programadores, devido ao "encontro" assíncrono que é necessário quando ocorre um I/O. Além disso, a espera ocupada é menos eficiente do que o I/O controlado por interrupção, de modo que o desempenho geral do sistema diminuiria.

15) Quais são as três maneiras de se manter a hora do dia. Explique cada uma delas.
1. Fazer um contador de 64 bits. A desvantagem é que sua manutenção é dispendiosa, visto que ele deverá contar várias vezes por segundo.

Fazer um contador que armazene horas em segundos, com um contador auxiliar que conte os tiques. Se for um contador de 32 bits, esse método funcionará por mais de 100 anos.
contar os tiques, porém com referência no tempo de boot do sistema. Assim, quando for solicitada a hora do dia, o sistema soma o tempo de boot com o contador de tiques,fornecendo o horário correto.

Cap.6

Por que alguns sistemas rastreiam o tipo de um arquivo, enquanto outros deixam esta tarefa para o usuario e outros simplesmente nao implementam multiplos tipos de arquivos? Que sistema e "melhor"?
Alguns sistemas permitem diferentes operações sobre o arquivo com base no tipo do arquivo (por exemplo, um arquivo ASCII pode ser lido como um stream, enquanto um arquivo de banco de dados pode ser lido por um índice para um bloco). Outros sistemas deixam tal interpretação dos dados de um arquivo para o processo e não oferecem ajuda no acesso aos dados. O "melhor" método

depende das necessidades dos processos no sistema e das demandas que os usuários fazem sobre o sistema operacional. Se um sistema executar principalmente aplicações de um banco de dados, pode ser mais eficiente para o sistema operacional implementar um arquivo do tipo banco de dados e oferecer operações apropriadas, em vez de fazer com que cada programa os implemente (possivelmente, de diferentes maneiras). Para sistemas de uso geral pode ser melhor implementar apenas os tipos de arquivos básicos, para manter o tamanho do sistema operacional menor e permitir o maximo de liberdade aos processos do sistema.

Por que alguns sistemas rastreiam o tipo de um arquivo, enquanto outros deixam esta tarefa para o usuario e outros simplesmente nao implementam multiplos tipos de arquivos? Que sistema e "melhor"?

Uma vantagem de fazer com que o sistema admita diferentes estruturas de arquivos reside no fato de que o suporte é fornecido pelo SO, as aplicações não precisam prover o suporte. Além disso, se o sistema oferece o suporte para diferentes estruturas de arquivos, ele pode implementar o suporte de forma eficiente, presumivelmente mais eficiente que uma aplicação. A desvantagem de fazer com que o sistema forneça o suporte para tipos de arquivos definidos é que isso aumenta o tamanho do sistema. Além disso, as aplicações que podem exigir diferentes tipos de arquivos além do que é fornecido pelo sistema podem não ser capazes de executar em tais sistemas. Uma estratégia alternativa é que o SO não defina suporte para as estruturas de arquivo e, em vez disso, trate todos os arquivos como uma série de byte. Essa é a técnica utilizada pelos sistemas UNIX. A vantagem dessa técnica é que ela simplifica o suporte do SO para os sistemas de arquivos uma vez que o SO não precisa mais fornecer a estrutura para diferentes tipos de arquivos. Além do mais, isso permite que as aplicações definam estruturas de arquivo, aliviando assim as situação em que um SO pode não oferecer uma definição de arquivo exigida para uma aplicação especifica.

1) (Tan-2003) Dê 5 nomes diferentes de caminhos para o arquivo /etc/passwd. Dica: pense sobre as entradas de diretório "." e "..".

/etc/../etc/passwd
/./etc/password
/./../etc/../etc/password
/etc/./../password
/etc/../etc/password/.

2) (Tan-2003) No Windows, quando um usuário dá dois cliques sobre um arquivo relacionado pelo Windows Explorer, é executado um programa e aquele arquivo é oferecido como parâmetro. Liste duas formas diferentes de como o sistema operacional poderia saber qual programa executar.

Pela extensão do arquivo. Por exemplo, num arquivo file.doc o sistema operacional pode detectar que ele pode ser aberto com o Word.

Pelo programa que gerou o arquivo, assim o sistema operacional pode abrir com o mesmo programa que o gerou.

3) (Tan-2003) Nos primeiros sistemas Unix, os arquivos executáveis (arquivos a.out) começavam com um número mágico muito específico que não era escolhido aleatoriamente. Esses arquivos eram iniciados por um cabeçalho, seguido pelos segmentos de código e de dados. Por que um número específico foi escolhido para os arquivos executáveis, se outros tipos de arquivos tinham um número mágico mais ou menos aleatório como primeira palavra?

Como esses sistemas carregavam o programa diretamente na memória e iniciavam sua execução logo na palavra 0, os arquivos executáveis possuíam em seu cabeçalho, ou seja, em seu número mágico, uma instrução de desvio (BRANCH) seguido do endereço de inicio do código a ser executado. Dessa forma, era possível executar um arquivo binário diretamente de um espaço de endereço de um novo processo, mesmo sem saber o real tamanho do cabeçalho.

4) (Tan-2003) Alguns sistemas operacionais fornecem uma chamada ao sistema "rename" para atribuir um novo nome a um arquivo. Há alguma diferença entre usar esta chamada para alterar o nome de um arquivo e apenas copiá-lo para um novo arquivo com o novo nome e depois remover o antigo?

Sim. Com a criação de um novo arquivo de mesmo nome suas informações de data de criação e data de modificação seriam diferentes, caso fosse utilizada a chamada rename, que não altera essas datas.

5) (Tan-2003) Um sistema operacional simples suporta somente um diretório, mas permite que o diretório tenha muitos arquivos com tamanhos arbitrários de nomes. Pode ser aproximadamente simulado um sistema hierárquivo de

arquivos? Como?
Sim é possível. Poderia ser adicionado ao nome do arquivo os caminhos relativos, como é utilizado em diretórios. Por exemplo:
Usr/utfpr/SO/cap3.pdf
Usr/utfpr/Mecânica/exercicios.pdf
Usr/Estagio/Relatório.doc

6) (Tan-2003) Considere a árvore de diretórios da Figura 6.10. Se /usr/jim for o diretório de trabalho, qual é o nome do caminho absoluto para o arquivo cujo caminho relativo é ../ast/x?
/usr/ast/x

7) Quando se fala em arquivos armazenados em disco, diferencie fragmentação interna de fragmentação externa.
Fragmentação interna: perda de espaço dentro de uma área ou tamanho fixo. No particionamento de memória fixo, um bloco possui um tamanho determinado e , quando um arquivo possui um tamanho inferior ao bloco, ocorre uma perda de espaço que não será utilizado.
Fragmentação externa: perda de espaço ocasionado por lacunas vazias disponíveis na memória. Se, um programa, por exemplo, termina sua execução e libera o seu espaço que estava ocupando na memória, e uma nova execução, de menor tamanho é alocado nesse espaço recém liberado, a memória ficará com lacunas em seu espaço. Esse problema pode ser resolvido com uso de algoritmos de compactação.

8) (Tan-2003) A alocação contígua de arquivos leva a uma fragmentação do disco, conforme mencionado no texto, pois algum espaço no último bloco do disco será desperdiçado nos arquivos cujo tamanho não corresponda a um número integral de blocos. Essa fragmentação é interna ou externa? Explique.
É uma fragmentação externa, uma vez que a perda de memória é entre os arquivos alocados na memória, e não nos próprios arquivos entre si.

9) (Tan-2003, adaptado) Um modo de usar a alocação contígua de disco e não sofrer com as lacunas é compactar o disco toda vez que um arquivo for removido. Faz algum sentido essa compactação? Explique.
Não. O ideal é compactar o disco após um novo arquivo ser adicionado no local do antigo arquivo removido. Algoritmos de compactação consomem elevado tempo e processamento da CPU, por isso deve ter seu uso reduzido ao mínimo.

10) (Tan-2003) Alguns compradores de dispositivos digitais precisam armazenar dados - por exemplo, arquivos. Dê o nome de um dispositivo moderno que requer armazenamento de arquivos para o qual a alocação contígua seria uma boa idéia.
Uma máquina fotográfica digital. Cada foto tirada pelo usuário pode ser armazenada contiguamente na memória da máquina e, quando for necessário transferi-las, todas as fotos podem ser removidas de uma vez, liberando espaço na memória.

11) (Tan-2003) O início de um mapa de bits do espaço livre parece-se com isto depois que a partição de disco é formatada pela primeira vez: 1000 0000 0000 0000 (o primeiro bloco é utilizado pelo diretório-raiz). O sistema sempre busca por blocos livres a partir do bloco com menor número; assim, depois de escrever um arquivo A, que usa seis blocos, o mapa de bits se parece com isto: 1111 1110 0000 0000. Mostre o mapa de bits depois de cada uma das seguintes ações adicionais:
a) O arquivo B é escrito, usando cinco blocos.
1111 1111 1111 0000
b) O arquivo A é removido.
1000 0001 1111 0000
c) O arquivo C é escrito, usando oito blocos.
1111 1111 1111 1100
d) O arquivo B é removido.
1111 1110 0000 1100

12) (Tan-2003) Um certo sistema de arquivos usa blocos de disco de 2kB. O tamanho mediano do arquivo é 1kB. Se todos os arquivos forem exatamente de 1kB, qual fração do disco será desperdiçada? Você acha que o desperdício para um sistema de arquivos real será mais alto ou mais baixo do que esse? Explique.
50%. Desperdício = 1kB/2kB = 0.5
Num sistema real, o desperdício será menor, uma vez que o sistema terá arquivos maiores que os blocos de disco. Por exemplo, um arquivo de 111kB desperdiça aproximadamente 1% da memória (1kB/120kB).

13) Considerando a implementação de um sistema de arquivos utilizando a alocação por lista encadeada, por que o acesso aleatório é extremamente lento? Explique. Explique, também, de que forma a FAT (tabela na memória) acelera esta leitura.
Pelo fato que para se localizar o

arquivo numa lista encadeada numa posição n, terá de ser percorrido n-1 elementos até ser encontrado. Se o elemento estiver no fim da lista, todos os elementos terão de ser percorridos. Isso torna a busca de acesso aleatório lenta.
A tabela de alocação de arquivos (FAT) armazena cada ponteiro de bloco de memória. Dessa forma, a busca por um arquivo ainda terá de ser seguida sequencialmente, porém sem a necessidade de se realizar qualquer referência ao disco.

14) Qual é a principal finalidade de uma entrada de diretório?
Armazenar os atributos de um diretório (nome, data de criação, blocos de disco). No caso de sistemas UNIX, a entrada armazena um nome e i-node, que armazena os demais atributos.

Quais sao as vantagens e desvantagens de se gravar o nome do programa criador (aquele que criou o arquivo) junto com os atributos do arquivo (como e feito no Mac OS)?
Registrando o nome do programa criador, o SO é capaz de implementar recursos (como chamada automática do programa quando o arquivo for acessado) com base nessa informação. Porém, isso gera um custo adicional para o SO e exige espaço no descritor do arquivo.

Alguns sistemas automaticamente abrem um arquivo quando ele e referenciado pela primeira vez e fecham o arquivo quando o job termina. Discuta as vantagens e desvantagens deste esquema quando comparado com a abordagem mais tradicional, onde o usuario tem que abrir e fechar o arquivo explicitamente.
A abertura e fechamento automático de arquivos alivia o usuário da chamada dessas funções e, por isso, é mais conveniente para o usuário, porem, isso exige mais custo adicional do que o caso em que são exigidos abertura e fechamento explícitos.

Se um sistema operacional soubesse que uma determinada aplicacao acessaria os dados do arquivo de forma sequencial, como o SO poderia explorar esta informacao para melhorar o desempenho?
O SO deverá utilizar a técnica de alocação adequada. No caso do acesso seqüencial, a alocação contigua é eficiente.

Considere um sistema de arquivos (FS) que utiliza inodes para representar arquivos. Blocos de disco tem tamanho de 8kB, e um ponteiro para um bloco de disco requer 4 bytes. Este FS tem 12 bocos diretos de disco e, tambem, um bloco indireto simples, um duplo e um triplo. Qual e o tamanho maximo de um arquivo que pode ser armazenado neste FS?
Aproximadamente 16 TB.

Se os HDs magneticos tiverem o mesmo custo por gigabyte que as fitas, estas ficarao obsoletas, ou ainda serao necessarias? Explique sua resposta.
Ainda serão necessárias, pois se armazenadas devidamente, resistem por muito mais tempo e os esforços muito maiores que os HDs, possibilitando armazenar backups durante muito mais tempo.