1ª prova 16/12/2003:

1) Qual problema é resolvido(e como é resolvido) pelo overlay? É necessário alguma facilidade especial de hardware para a implementação de overlay? Qual? Cite, exemplificando uma outra solução para o mesmo problema que NÂO necessite de facilidade especial de Hardware.

O problema resolvido é a falta de memória, ou seja, quando o usuário quer executar processos que não cabem na memória da máquina. O overlay divide o processo em partes iguais e aloca na memória apenas instruções e dados que são necessárias em determinado momento. Quando outras instruções são necessárias elas são carregadas no espaço que foi anteriormente ocupado por instruções que não são mais necessárias.
O overlay não precisa de suporte de Hardware sua implementação exige muito do programador.
O swap de processos não precisa de suporte especial de HW. Se a memória não tem espaço para executar um processo, um outro processo é removido da memória para um armazenamento auxiliar temporariamente liberado espaço para o novo processo executar e, em seguida, retornando a memória para continuar executando.

2) Qual a relação entre a falta de página e página vítima? Essa relação sempre existe? Por quê?

A relação é quando um processo solicita uma página e a página não foi carregada na memória, quando tenta carrega-la não há página livre na memória. Então o SO utilizando um algoritmo de substituição escolhe uma página (página vítima) para salvar em disco, liberando espaço para a página que gerou a falta de página. Essa relação existe, pois se a memória estiver toda ocupada e ocorrer falta de página, o SO tem que escolher uma página vítima para dá lugar a essa página que gerou a falta. Na realidade, o SO sempre deixa algumas páginas livres, quando pode escolher páginas vítimas. Dessa forma otimiza o tempo, ou seja, o tempo que levaria para escolher uma página vítima e fazer todo processo necessário de tratamento dessa página, não seria junto com o carregamento da página que gerou a falta em situações que a memória estaria toda ocupada. Com isso nem sempre a falta vai startar a escolha de página vítima. Se há páginas livres necessariamente não precisa escolher página vítima para gravar no disco.

3) Por que o polling (esperao cupada) tem um loop de espera? Qual é (e como é) o mecanismo de HW que possibilita existir uma melhor forma de resolver o problema.

Porque as vezes está em laço, lendo o registrador status várias e várias vezes até que o bit ocupado esteja desativado. Se a controladora e o dispositivo forem rápidos, esse método será razoável. Se o polling se tornar ineficiente, quando testado repetidamente, raramente encontrando um dispositivo pronto para serviço, enquanto outros processadores úteis da CPU permanecem por fazer. Em tais casos, pode ser mais eficiente fazer com que a controladora notifique a CPU quando o dispositivo ficar pronto para serviço, em vez de exigir que a CPU faça repetidas consultas para verificar a conclusão de uma operação de I/O . O mecanismo de HW que permite que um dispositivo notifique a CPU é a interrupção.

4) A) Diga quais são, exemplificando, os 3 tipos de tempo gastos na leitura de um disco.
B) Diga quais são, exemplificando, as 3 dimensões que precisam ser identificadas para o acesso ao disco. Explique como este acesso tridimensional pode ser simplificado (use o conceito de cilindro.

O tempo gasto na leitura de um disco é a soma de três tempos:

O tempo de seek: é o tempo gasto para que a cabeça de leitura se desloque até a trilha onde está o setor a ser lido.
O tempo de rotação: é o tempo gasto para que o setor a ser lido passe sob a cabeça de leitura.
O tempo de latência: é o tempo gasto na transferência de um conjunto de bytes do disco para a memória.
Nos discos modernos há vários pratos, um em cima do outro, com um espaçamento entre eles. Um prato tem 2 superfícies, logo, precisam ser informados 3 informações para ler/grava/apagar um dado: trilha (faixa do disco concêntricas). As trilhas se dividem em setores (uma fatia do disco) Não sei a definição de setores., e superfície.

5) Processos podem se comunicar através de mensagens. O envio de mensagens de um processo a outro processo costuma ser implementado através da cópia do conteúdo da mensagem que está em um processo para uma variável do outro processo. Como seria possível, com a paginação, fazer o envio da mensagem SEM a cópia do conteúdo da mensagem.

Um processo A quer mandar uma mensagem para o processo B. O processo A manda a mensagem para a memória física (send (msg, B)) uma página, na qual contém a mensagem e sua tabela de páginas está marcada com o bit de presença ativo para esta página. O processo B compartilha essa página na memória física, com isso o processo A desabilita na sua tabela de página o bit de presença desta página, e o processo B habilita na sua tabela de página (msg = receive();)

P1 15/12/2003

1) Como o problema da fragmentação externa pode ser resolvido quando existe paginação?

Ocorre fragmentação externa quando os processos estão alocados na memória de forma não contínua existindo espaço total para alocação de um processo, porém ele não pode ser alocado porque o espaço não é contínuo. Com a paginação a memória é divida em páginas de tamanho fixo (geralmente 4 KB), onde os processos tem sua tabela de página onde pe mapeado o endereçamento lógico e físico de cada página.

2) Qual o problema que existe em relação aos endereços utilizados quando um programa é carregado na memória e se torna um processo? Como este problema pode ser resolvido quando existe paginação?

A amarração de endereços pode ser feita em diversas fases nas quais preparam a criação de um programa executável: os processo possui um nome e um endereço,
Printf(“o valor ..%d”,i);
Na linguagem de máquina CALL 73F
quando são carregados a HW fazer a tradução do nome para linguagem de máquina, dependendo do modo que foi carregado (endereço em tempo de carga, endereçamento relativos, registrador-base) vão existir várias correções de endereços durante a execução, carregamento, compilação..
Na paginação o processo é quebrado em páginas de tamanho fixo, todo processo tem seu espaço de endereçamento lógico no qual o processo começa por zero e acha que está sozinho na memória. As páginas são alocadas na memória física quando vão ser executadas, e existe uma tabela para cada processo, que faz o mapeamento do endereçamento lógico para o físico. Não sendo necessário alocação contínua de páginas na memória física.

3) Diga quais são, exemplificando, os 6 bits que podem existir na tabela de páginas. Quais são os 3 bits utilizados (e como estes são utilizados) durante o mecanismo de resolução dos problemas de falta de memória.

Bit presente – informa se a página está na memória bit =1 (válido) ou se estiver inválido bit = 0 a página não pertence ao processo, a página pode ter sido escolhida como vítima; a página pode nunca ter sido carregada, está em disco.

Bit acessado (ligado/desligado): bit utilizado na implementação do algoritmo de substituição de páginas de 2ª chance ou do algoritmo relógio. Indicando se a página foi ou não acessada recentemente.

Bit de escrita ou alterável: é aquele que se estiver ligado (bit=1) indica que o conteúdo da página pode ser alterado, se o bit=0 o conteúdo da página não poderá ser modificado.

Bit executável: indica se a página considerada pode ou não ser executada. Nas páginas de código este bit se encontra ligado e nas páginas de dados o mesmo se encontra desligado, para que não se execute uma página que não seja de código.

Bit núcleo: ligado a página pertence ao SO, e por isso os usuários não tem permissão de escrita, modificação, somente o administrador do SO tem este tipo de permissão, desligado não pertence ao SO.

Bit alterado: o bit é ativado pelo HW sempre que qualquer palavra ou byte na página for alterado, indicando que a página foi modificada. Quando uma página é selecionada para substituição, examinamos seu bit de modificação. Se o bit estiver ativado, sabemos que a página foi modificada desde que foi lida do disco. Nesse caso, devemos gravar a página no disco. Se o bit não estiver ativo a página não foi modificada desde que foi carregada na memória.

4) O que é e como funciona a TLB? Explique porque seu bom funcionamento é de extrema importância em uma das soluções do problema de tabela de páginas grandes.

É um mecanismo interno a CPU chamado de memória associativa (memória rápida) ou TLB que aumenta a velocidade da tradução dos endereços lógicos para físicos.

Na primeira vez que o HW faz o mapeamento de uma página o HW vai na tabela de páginas gastando 1 acesso a memória. O HW coloca os valores das páginas reais e virtuais numa memória especial, interna ao chip da CPU. Conforme o HW vai fazendo acesso a memória ele vai completando as linhas da TLB. Se mais tarde vier uma instrução que precise de uma determinada página, o HW antes de ir na tabela de página ele verifica se já existe na TLB a tradução da página, se existir o HW não precisa consultar a tabela de páginas. Com essa memória associativa se consegue traduzir o endereço de virtual para físico de forma rápida.Quando a TLB enche, o HW tem que escolher um mapeamento que já foi feito e jogar fora para gravar o mapeamento de novos endereços a serem usados, o que leva um tempo maior pois ocorrerá dois acesso , 1 na TLB e outra na tabela de páginas. A idéia é o HW manter automaticamente na TLB o mapeamento das últimas páginas utilizadas.

A importância da TLB é que existe HW que não possui tabela de páginas e esse mapeamento é feito pela TLB somente.

5) O swap de processos resolve 2 problemas em sistemas operacionais diferentes. Como e quais são estes problemas e soluções?

1º problema: falta de memória, neste caso o processo é maior que a área livre disponível na memória, o SO remove um processo da memória temporariamente para um armazenamento auxiliar e, em seguida, carrega o processo que vai ser executado.

2º problema: Thrashing

Este caso é específico para o SO UNIX. Quando ocorre excessiva paginação, o sistema não faz processamento. Todos os processos gastam seu tempo de execução simplesmente fazendo paginação. Nesse caso o SO escolhe um ou mais processos que estão causando o Thrashing e executa um swap de tais processos para o disco, aguarda até que os outros processos terminem e em seguida aloca os processos bloqueados para a execução.

P2 08/09/2003

1) a) Sem considerar as operações de criar, excluir, renomear e alterar informações de controle de arquivos, qual é o conjunto mínimo de chamadas ao SO que podem ser usadas para manipular arquivos? Descreva os parâmetros deste conjunto mínimo de chamadas.

b)Quais são as chamadas (com seus parâmetros e seus retornos) comumente existentes em SOs reais em substituição ao conjunto mínimo do item anterior.

c) Explique pelo menos 2 vantagens das chamadas do item "b" sobre as chamadas do item "a".

2) Exemplifique um bitmap de controle para blocos do disco. Com este bitmap, imagine uma execução de um programa que escreve 11KB no arquivo "nada.txt" que está originalmente vazio. Mostre como ficam as estruturas de controle (bitmap inclusive, se necessário) depois da escrita nos casos de sistema de arquivo do tipo:

a) FAT
b) Indexada (UNIX)
c) NTFS

Considere que cada bloco ocupa 1 KB

3) a) Como um programa altera o conteúdo de um arquivo mapeado em memória?
b) Qual a vantagem de utilizar arquivos mapeados na memória ao invés de utilizar arquivos com chamadas convencionais (item 1b).

4) Discuta a oposição entre cache de disco e paginação sob demanda. Mostre duas possíveis soluções para o problema.

5) Suponha que a cabeça do disco está no cilindro 71, antes estava no 61. Existe uma série de pedidos para os seguintes cilindros: 43, 735, 457, 887, 474, 754, 511, 875 e 65. Mostre, explicando, a execução de 4 diferentes algoritmos de escalonamento de disco.

P2 16/02/2004

1) Diferencie dispositivos de bloco e dispositivo de caractere. Classifique, explicando, os dispositivos: impressora laser, DVD e fita magnética. Qual é o relacionamento entre arquivo físico e dispositivos (mencione o tipo de dispositivo) ?

2) Explique quais são as duas abordagens existentes em relação ao tipo de dado armazenado em uma entrada de diretório. Existe uma facilidade importante que só é possível em um dos tipos. Explique que facilidade é esta e porque só é possível neste tipo.

3) Normalmente em um SO não existe chamada para cópia de arquivos. Escreva um trecho de código, utilizando as chamadas estudadas, que faça uma cópia de um arquivo grande (>4GB).

4) Exemplifique, mostrando as estruturas, uma situação para um arquivo de 9 blocos, na qual a gerência com NTFS é pior que a gerência do UNIX. Explique porque você considerou pior.

5) Explique como o conceito tradicional de setor circular foi modificado nos discos atuais, descrevendo como é feito atualmente. Explique a razão que motiva a alteração do conceito original.

P2 - 02/08/2004

1)A fragmentação externa é um problema importante na alocação de um arquivo? Por quê?
Sim. Quando todos os blocos de memória livre são pequenos demais para acomodar um segmento. Nesse caso um processo pode simplesmente ter de esperar até que mais memória se torne disponível, a compactação pode ser usada para criar um bloco de memória livre maior.

2) Supondo tamanho do bloco igual a 2 KB e a alocação do Unix. Dados os seguintes trechos de código, diga, justificando, quantos blocos do disco são utilizados pelos novos arquivos
a) int fd = creat (“a.dat”, 0644); // 0644 é a permissão do arq write (fd, buffer, 30 * 1024)
30*1024
Serão necessários: 10 blocos para os primeiros blocos de dados deste arquivo; mais 1 bloco que será usado como bloco de indireção e que conterá o endereço dos próximos 5 blocos de dados usados por este arquivo; mais 5 blocos de dados restantes.
Total de blocos usados pelo arquivo “a.dat”: 10+1+5=16 blocos (não considerando o bloco onde fica o INODE deste arquivo.

b) int fd = creat (“b.dat”, 0644); // 0644 é a permissão do arq
write (fd, buffer, 1100 * 1024);
1100 * 1024 = 550 * 2048
10 blocos para guardar os 10 primeiros blocos para guardar os 10 primeiros blocos de dados do arquivo; restam 540 blocos de dados a serem referenciados no INODE. Usando 1 bloco de indireção, bloco este que tem tamanho 2KB e considerando que cada entrada tem 4 bytes, este bloco de indireção pode guardar 512 endereços de blocos.
Logo, ainda restam 28 blocos a serem referenciados no INODE. Então precisaremos usar mais 2 blocos para utilizar a técnica de dupla indireção. Com isso, conseguiremos referenciar os últimos 28 blocos da dados do arquivo.
Portanto, o número total de blocos utilizados pelo arquivo “b.dat” será de
10 + 540 + 1 + 2 = 553 blocos (não considerando o bloco onde fica o INODE deste arquivo)

3)Diferencie lista de capacidades de lista de acesso. Descreva as duas implementações típicas de lista de acesso.
A lista de acesso especifica para cada nome de usuário listando os tipos de acesso permitidos, quando um usuário solicita acesso a determinado arquivo, os SO verifica a lista de acesso associadas àquele arquivo.E a lista de capacidade é uma lista com as operações permitidas por usuários, permissões de alterar ler e escrever.
Permissões associadas a um grupo de usuários; para cada arquivos existem somente 3 categorias de usuários (dono do grupo, único grupo e todos os demais usuários)

4) Supondo que a FAT seja um array chamado FAT e que o diretório corrente seja um array de registros chamado DIR. Foi executado o seguinte código:
int fd = open ("dados.dat", O_RDONLY);
read (fd, x, 1);
Escreva o código em C ou Pascal que descubra qual o bloco do disco que precisa ser lido na chamada read acima.
Suponha tamanho do bloco igual a 2 KB e os campos nome e bl_inicial no diretório.

```
I, J: INTEGER;
DIR: ARRAY [1..5000] OF RECORD OF
            NOME: CHAR[25];
            BL_INICIAL: INTEGER;
    FAT: ARRAY[1..5000] OF INTEGER;
BEGIN
WHILE DIR[I].NOME<> "DADOS.DAT" DO
    BEGIN
        I:=I+1;
    END;
    J:= DIR[I].BL_INICIAL;
    WRITELN("O BLOCO A SER LIDO É O ", J);
END.
```

5) Porque os algoritmos de escalonamento lidam com números de cilindro e não com número de bloco ou de números de superfície.
O disco é formado por um conjunto de pratos. Cada prato tem duas superfícies: a superfície de trilhas correspondentes em cada superfície chama-se cilindro.
Não se usa número de bloco nos algoritmos de escalonamento pois pode-se mudar o bloco a ser acessado mas não necessariamente haverá deslocamento da cabeça para ler/escrever um outro bloco pois podem ambos os blocos estarem na mesma trilha e conseqüentemente, no mesmo cilindro. Não se usa também número de superfície pois pode-se ler uma trilha correspondente em qualquer superfície sem necessariamente deslocar a cabeça.

6) Existem casos em que a chamada read:

a) bloqueia o processo que faz a chamada? Se existe, quando ocorre bloqueio e o desbloqueio?
Sim, se o arquivo for mapeado em memória e a página solicitada pelo arquivo não estiver na memória. Isso causará uma falta de página e o processo será bloqueado. O mesmo será desbloqueado quando a página tiver sido carregada do disco para a memória.

b) Não bloqueia o processo? Se existe este caso, quando ocorre?
Sim, se a página solicita já estiver na memória.

10/7/2002

1)a) O que existe em comum entre escalonamento de disco e escalonamento de procesos?
b)Suponha que cheguem pedidos de leitura no disco de 5ms em 5ms. Suponha que cada pedido do disco sejam atendidos de 50ms em 50ms. Dados os pedidos dos seguintes blocos (nesta ordem) 4, 7, 5, 20, 6, 18. Diga, justificando, como ficaria a ordem de escalonamento para estes blocos utilizando 4 algoritmos conhecidos.

2) Seja um arquivo que ocupa os blocos: 4, 7, 5, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29.

a) Mostre como ficam as estruturas de controle para MS-DOS, Unix e Windows NT (NFTS)
b) Qual das 3 estruturas poderia ser considerada a melhor? Justifique porque a escolhida é melhor que cada uma as outras duas.

3)a) É possível existir um sistema operacional que leia e escreva em arquivos mas não possua uma chamada que "abra" os arquivos? Justifique.
b)Suponha que exista o sistema do item a), compare-o com um sistema operacional que tenha a chamada de abertura . Qual seria o mais vantajoso? Justifique.
c)Em um sistema com abertura, quais seriam os parâmetros típicos de uma chamada de leitura? E uma chamada de escrita? Como poderia ser feito um acesso de leitura aleatório.
d) Qual o relacionamento das chamadas de leitura e escrita com o mapeamento de arquivo na memória ?

4) a) Por que os objetivos da cache de disco e da memória virtual são contraditórios?
b) Descreva 2 formas de diminuir ou resolver esta contradição.

Prova de Reposição 17/12/2004

1) a) Qual a vantagem da cachê de disco write through para a write back ?
b) Qual a desvantagem da write throught para write back?
c) A mesma vantagem existe na cache de memória ? Por quê ?

2) Em qual das soluções para tabela de páginas grandes a existência da TLB tem a maior importância ? Por que?

3) a) Quais são os 6 bits que podem existir na tabela de página ?
b) Destes 6, 2 são fundamentais e necessitam existir em hardware. Quais são ? Por quê ?
c) Os 4 demais não existir obrigatoriamente em hardware. O que pode ser feito caso eles não existam ?

4) A) Qual a vantagem da alocação encadeada sobre a alocação contínua ?
b) Qual a vantagem da alocação FAT simples sobre a alocação encadeada simples ?

c) Qual a vantagem da alocação indexada sobre a alocação com FAT?
d) Qual a vantagem da alocação por extensão sobre a alocação indexada ?

Prova final 09/08/2004

1) Supondo que a FAT seja um array chamado FAT e que o diretório corrente seja um array de registros chamado DIR. Foi executado o seguinte código:

```
Int fd = open ("dados.dat", O_RDONLY);
Lseek (fd, 20480, SEEK_BEGIN);  // A partir do início
Read (fd, x, 1);
```

Escreva o código em C ou Pascal que descubra qual o bloco do disco que precisa ser lido da chamada read acima.
Supondo tamanho do bloco igual a 2KB e os campos nome e bl_inicial no diretório.

2) Seja uma máquina com tabela de páginas invertida. Dado um endereço lógico, 3020H (em hexadecimal), mostre como este endereço é convertido em endereço físico. Utilize páginas de 4 KB.
3) Mostre, com valores, a(s) estrutura(s) de controle de arquivo(s) para os 3 casos básicos de inconsistência utilizando FAT.
4) Descreva os passos (mencionando o responsável) que ocorrem na forma de E/S que você considera mais eficiente. Justifique a escolha.
5) Explique a diferença entre cachê de memória e cachê de disco. Explique a contradição entre cachê de disco e paginação sob demanda. Para resolver esta contradição, qual novo conceito que surge e qual conceito deixa de ser utilizado?

Prova final 01/03/2004

1) Seja um disco com 200 cilindros, para este disco chegam pedidos de leitura para os cilindros 55, 58, 39, 18, 90, 160, 150, 38 e 184 (nesta ordem). Dado que a cabeça está atualmente no cilindro 100 indo para o 200, mostre como os pedidos serão atendidos utilizando os seguintes algoritmos: a) FCFS b) SSTF c) Elevador

2) Suponha que a tabela de páginas de uma máquina esteja contida toda em registradores. Suponha os seguintes tempos.

- 8 milissegundos para tratar uma falta de página quando existem páginas físicas livres;
- 20 milissegundos para tratar uma falta de página quando é necessário fazer a substituição de página;
- 100 nanossegundos para acessar a memória física.

Suponha que me 70% das faltas de página, a página tenha que ser substituída. Mostre, explicando, qual a maior taxa para falta de página de forma que o tempo efetivo de acesso à memória seja de 200 nanossegundos.

3)Responda se as afirmativas seguintes são verdadeiras ou falsas, justificando sua resposta:

- Para todos os processos, existe um número tal que um processo tenha menos páginas físicas do que este número, ele executará mais lentamente, caso ele tenha mais páginas físicas que este número sua velocidade de execução praticamente não sofrerá alteração

- O tamanho do Working Set de um certo processo é alterado dependendo do número de processos que estão em execução.

4)Considere a alocação de arquivos do Unix. Assuma que os blocos têm 4KB e cada ponteiro para bloco tenha 8 bytes. Diga, justificando, qual é o maior tamanho possível de arquivo que pode ser gerenciado (deixe os cálculos inidicados).

5) Para cada uma das operações abaixo, diga, justificando, se o inode necessita ser alterado (considere o arquivo não esparso e desconsidere a data e hora de última alteração):

- Leitura de um bloco no meio do arquivo;
- Escrita em um bloco no meio do arquivo;
- Escrita após o final de um arquivo de uma informação cujo tamanho é maior que um bloco;
- Escrita após o final de um arquivo de uma informação com 1 byte de tamanho.

P2 15/12/2004

1) Quais são os três possíveis tipos de conteúdo de uma entrada no diretório? Qual(is) deste(s) tipos(s) permite(m) o conceito de link? Porque?
Arquivo, os atributos de um arquivo e endereço de disco, outra possibilidade é onde uma entrada de diretório armazena o nome do arquivo e um ponteiro para outra estrutura de dados onde os atributos e os endereços de disco estão localizados.
Arquivo. Porque é uma técnica que permite que mais de um arquivo apareça em diretórios diferentes.

2)A) Exemplifique, justificando, um dispositivo cuja E/S seja feita de melhor forma por DMA
Disco. Ima leitura no disco é melhor com o uso do DMA pois o DMA é um dispositivo que está ligado tanto ao barramento que liga a CPU aos dispositivos como ao barramento que liga a CPU a memória, então ele faz a transferência do disco para a memória sem ocupar a CPU.

B) Exemplifique, justificando, um dispositivo cuja E/S seja feita de melhor forma por interrupção.
Mouse. Como o volume de dados é pequeno pode-se usar a interrupção que não ocupa o barramento, podendo-se executar outros processo.

C) Existe algum dispositivo cuja forma de E/S mais vantajosa seja a espera ocupada? Por que?
Não. Porque a espera ocupada ocupa a CPU sem fazer nada e para volume de dados maiores ou menores pode-se usar a interrupção ou a DMA.

3)Suponha que os blocos de dados de um certo arquivo ocupem os blocos 1024 a 2047 do disco. Suponha que cada bloco tenha 2 KB. Mostre com valores, como ficam as estruturas de controle de alocação deste arquivo no UNIX.

4) Seja o seguinte mapa de bits de blocos é 1KB e que os blocos são alocados em ordem crescente. Mostre como fica a estrutura de controle da alocação por extensão quando as seguintes chamadas do UNIX são executadas:
fd = creat (“dados1.sat”, 0644);

write (fd, &var1, 1024);
lseek (fd, 4096, SEEK_BEGIN);
write(fd, &var2, 3*1024);
close(fd);
Creat é a chamada ao SO para a criação de um novo arquivo.

5) Diga quais são, explicando, os 3 tipos de tempos gastos em um acesso ao disco. Qual(is) deste(s) tempo(s) o escalonamento de disco objetiva minimizar? Qual dos akgoritmos de escalonamento mais minimiza este(s) tempo(s)? Por que?
Tempo de seek – o tempo que a cabeça demora para se mexer até a trilha
Tempo de latência – tempo gasto para o setor passa embaixo da cabeça
Tempo de transferência – é o tempo gasto na leitura de todo um setor pela cabeça para que o dado possa ser transferido para a memória controladora.
O escalonamento de disco objetiva minimizar o tempo de seek primeiro pois ele atende os pedidos que estão mais próximos de onde a cabeça está, evitando que ela se movimente muito. Só que esse algoritmo é injusto, os pedidos distantes podem ficar "esquecidos".

P1 30/05/2005

1)Dada uma máquina com paginação, mostre(com números de página) como é realizada a tradução de endereço lógico para endereço físico na instrução MOV[8200], AX. Nesta tradução são utilizados pelo menos 4 bits de controle da tabela de página. Mostre como estes bits são utilizados. Suponha páginas de 4 KB.

2)Explique como:
a) correção de endereços em tempo de carga
b) segmentação
c) paginação
são utilizados para resolver o problema do carregamento de vários programas na memória. Qual(is) deste(s) não pode(m) ser usados com swap de processos ? Por que?

3)Qual a relação entre thrashing e uso da CPU? Como o thrashing pode ser evitado antes que ele ocorra ? Como o thrashing pode ser resolvido depois que ele ocorre?

4)Calcule qual a probabilidade de tradução sem uso da tabela de páginas para que a perda de desempenho, em relação a uma máquina que não use paginação, seja de apenas 5%. Mostre os valores utilizados para os diferentes tempos que fazem parte da tradução.

5) Qual a razão mais importante para a Intel utilizar diretórios de páginas em sua CPU. Justifique com valores.

P2 08/07/2005

1)Dado o trecho de programas
int g [5000]; //variável global
...
int i;
for (=0;i<1024;i++) g[i] = 1;
for (i=1024; i<4000;i++) g[i] = 0;
for (i=4000; i<5000;i++) g[i] = 1;

a) Mosrte, a sequencia de chamadas ao SO para salvar o vetor g acima em um novo arquivo gastando a menor quantidade de blocos de dados possíveis.
b) Mostre como ficaria a parte de controle de alocação no inode.
c) Mostre como ficaria a estrutura de controle de alocação do NTFS>
OBS.: A chamada para criação de arquivo no Unix é fd= creat(nome_arq, permissões)
OBS.: O tamanho do bloco é 1KB

2) Das três formas que fazem sincronização para a E/S, diga e justifique, para cada uma, se ela é utilizada na prática em sistemas operacionais multiprogramados.

3) Mostre, exemplificando, como ocorrem as 3 inconsistência de bloco típicas em Sos que usam FAT.

4) Explique como o conceito tradicional de setor circular foi modificado nos discos atuais, descrevendo como os discos se organizam atualmente. Explique a razão que motiva a alteração do conceito original.

5) Dado um disco com um bloco de 8Kb, o tamanho da entrada no bloco de indireção igual a 4B e o uso de indireção tripla. Diga, justificando, qual o número máximo de blocos que um arquivo pode ter no UNIX.
Nº de entradas = tamanho do bloco / tamanho da entrada = (8 * 1024)/4 = 2048 entradas em cada bloco de indireção.

10+2048+(2048*2048)+(2048*2048*2048)= 8.594.130.954 *4 =34376 GB

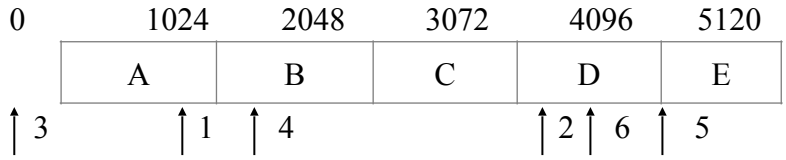
10 blocos do INODE + 2048 de indireção + (2048*2048) de dupla indireção + (2048*2048*2048) de tripla indireção

Reposição 11/07/2005

1) suponha que cheguem pedidos de leitura no disco de 10ms em 10ms. Suponha o tempo de demora para o atendimento de qualquer pedido seja o 25ms. Dados os pedidos para os seguintes cilindros (nesta ordem): 5 (t_chegada = 0ms), 7(t=10ms), 4 (t=20ms), 20 (t=30ms) e 6 (t=40ms). Diga, justificando, como ficaria a ordem de escalonamento para estes cilindros utilizando 3 (três) algoritmos conhecidos (excluindo-se o FCFS)

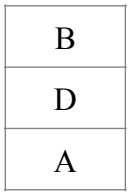
2) Sendo: a) tamanho do bloco em um disco igual a 1KB, b) SO com cache de disco do tipo *write-back*; c) organização da cache com lista duplamente encadeada; d) cache com tamanho máximo de 3 (três) blocos. Supondo a cachê inicialmente vazia, mostre graficamente o que ocorre com a cache em cada uma das chamadas de escrita, abaixo:

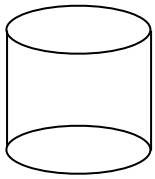
lseek (fd, 1100, SEEK_BEGIN); write(fd, &var1,10);
lseek (fd, 3500, SEEK_BEGIN); write(fd, &var2,20);
lseek (fd, 0, SEEK_BEGIN); write(fd, &var3,30);
lseek (fd, 1400, SEEK_BEGIN); write(fd, &var4,40);
lseek (fd, 4600, SEEK_BEGIN); write(fd, &var5,50);
lseek (fd, 3700, SEEK_BEGIN); write(fd, &var6,60);



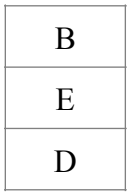
MEMÓRIA CACHE DISCO

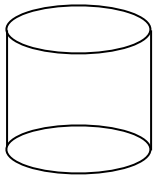
1





MEMÓRIA CACHE DISCO





Obs.: Não coloquei todas as setas, porém o algoritmo usado é o LRU.

3) a) Por que o mesmo algoritmo de escolha de vítima da questão 2 não pode ser usado com paginação? Outro algoritmo que tente aproximar o da questão 2 pode ser usado. Porque o problema do item A não existe neste algoritmo?
O algoritmo que não pode ser usado para paginação é o LRU.
OBS>: O prof. Ao corrigi uma prova colocou que o LRU não pode ser usado para paginação, pelo livro o LRU precisa de suporte de hardware ou se implementado usando o bit de referência .
4) Suponha tamanho do bloco igual a 1 KB, exemplifique (com valores) para o arquivos de 10 KB: a) um arquivo em que a alocação do NTFS seja muito melhor que a do Unix. B) um arquivo em que a alocação do Unix seja melhor que a do NTFS. Justifique suas respostas.
A) NTFS

Arquivo	Bloco inicial	Nº de bloco
A	6	10

1	7
2	8
3	9
4	10
5	11
6	12
7	13
8	14
9	15
10	16

UNIX

Na alocação por extensão cada pedaço contínuo é um elemento do arquivo. Portanto se o arquivo for todo contínuo só existirá um elemento na estrutura, e ela será bem resumida, bem melhor que a estrutura do Unix se o arquivo for pouco descontínuo, porque assim é necessário muito menos informações para

No Unix são necessários 10 inteiros

B) se o arquivo for muito descontinuo, a alocação por extensão é ruim porque o nº de registros na estrutura será grande e com isso, se gastará mais espaço e será mais difícil encontrar um bloco específico, porque terá que varrer toda a estrutura. Nessa situação a estrutura do Unix é melhor.

Arquivo	Bloco inicial	Nº de bloco
A	6	1
A	8	1
A	13	1
A	15	1
A	19	1
A	20	1
A	25	1

A	28	1
A	30	1
A	31	1

No Unix são necessários 10 inteiros

1	7
2	8
3	9
4	10
5	11
6	12
7	13
8	14
9	15
10	16

5)Como pode ser feita, usando paginação, a simulação do compartilhamento de memória entre memória distintas, mantendo sincronizado o conteúdo das páginas “compartilhadas” entre as duas máquinas.

Quando o compartilhamento é suportado com um único espaço de endereçamento, precisa haver mecanismo separado para efetuar a sincronização. Quando a página compartilhada esta sendo usada por uma máquina a outra máquina só poder usar esta página como readOnly (apenas leitura), até que a primeira máquina libere a página, após esta página ser liberada pela primeira máquina, outra máquina poderá usa-la e ela ficara readOnly para a primeira.