Henrique Yukio Sekido

NUSP: 14614564

1. Espaço de endereçamento lógico é o endereço com relação ao início do programa. Isso quer dizer que endereços lógicos iguais podem ter endereços físicos diferentes, pois os programas podem estar em locais diferentes na memória.  
   Já o espaço de endereçamento físico, é o endereço real e válido na memória, é algo mais concreto.
2. Supomos que um processo A - tamanho 230k - um processo B - tamanho 390k - e um processo C - tamanho 120k- estejam alocados na memória. De repente, apagamos o processo B, ficando um espaço de tamanho 390k entre os processos A e C. Após isso, chegou um processo D - tamanho 350k - e alocou-se ele nesse espaço entre os dois processos mais antigos.  
   Com isso, sobrou um espaço muito pequeno, de tamanho 50k, que provavelmente não será ocupado, devido ao seu pequeno espaço de endereçamento. Assim, formou-se uma fragmentação interna na paginação, um espaço gasto sem nenhuma utilidade.
3. Supomos que temos duas partições, uma com 100k e outra com 200k. Estamos querendo alocar um processo de tamanho 225k na memória, e o espaço total disponível é de 300k. No entanto, o processo não pode ser alocado, pois a memória disponível está particionada, de modo que não cabe o processo de 225k em uma única partição. Com isso, temos um espaço ocioso e com capacidade suficiente, mas o particionamento impede a alocação do processo.
4. Supomos um espaço de memória com tamanho 1000k. A diferença entre as duas estratégias de partição é que a fixa já tem um tamanho e número de partições pré estabelecidos. Já a partição variável, se chegar um processo de 176k, será alocada dinamicamente uma partição de 176k para aquele processo.  
   A vantagem da fixa é que a sua implementação é mais simples, e a desvantagem é que ela tem uma tendência a desperdiçar bastante memória, pois os processo podem não ocupar todo o tamanho da partição.  
   Já a variável, a sua vantagem é que otimiza a utilização da memória. Por outro lado, ela tem como desvantagem a dificuldade em alocar e liberar a memória. Ainda, tem uma menor fragmentação interna, mas uma maior fragmentação externa.
5. MMU, ou unidade de gerenciamento de memória, é um hardware que tem como função realizar o mapeamento de endereços lógicos para os endereços físicos na memória principal. Ele é efetivo em gerenciamento de memória virtual, manipulação e proteção de memória e controle de cache. Assim, o SO pode usá-lo para proteger o acesso à memória de programas que não deveriam poder acessar.
6. Paginação é a divisão do espaço de endereçamento virtual em blocos de tamanhos fixos, chamados páginas. Já a segmentação, é a divisão do espaço de endereçamento virtual em segmentos virtuais, que são blocos de tamanho arbitrário.
7. a) First fit



b) Best fit



c) Worst-fit



d) Next fit



O algoritmo mais eficiente foi o best fit.

1. O bit R significa referenciado e o bit M significa modificado. O bit M é essencial na troca de página, pois o algoritmo NRU o utiliza para classificar as páginas em 4 classes diferentes.

Classe 0: não referenciada, não modificada. Classe 1: não referenciada, modificada. Classe 2: referenciada, não modificada. Classe 3: referenciada, modificada.

De acordo com esse algoritmo, quando necessário, é removido uma página aleatória da classe mais baixa e não vazia.

Ainda, o bit M avisa se a página tem de ser gravada no disco antes de ser substituída por uma nova. Assim, evita-se a perda de informações.

1. O tamanho das páginas tem grande influência no gerenciamento de memória. Se o tamanho das páginas forem maiores, ocorre uma leitura mais eficiente, tem uma tabela menor, mas ocorre maior fragmentação interna. Já se o tamanho das páginas forem menores, ocorre uma leitura menos eficiente, tem uma tabela maior, porém uma menor fragmentação interna.
2. A falta de página ocorre quando uma página ainda não foi carregada na memória principal. Ela teria que ser carregada para a memória principal, pois seria utilizada, mas ainda não tinha sido carregada.

Quando ocorre uma falta de página, o processo que demandou a página é suspenso e seu descritor é inserido em uma fila especial(fila dos processos esperando uma página virtual). Após isso, uma página real livre deve ser alocada, a página virtual acessada deve ser localizada no disco, e a operação de leitura de disco deve ser feita, indicando o endereço da página virtual e da página real. Depois, a tabela de páginas do processo é atualizada para indicar que a página virtual agora está válida. O descritor do processo sai da fila especial e volta a fila normal.

1. O custo da implementação da memória virtual é a redução de desempenho, pois parte da memória é armazenada em disco. Assim, leva-se um tempo maior para acessar essas informações que são alocadas para o disco. Por outro lado, a memória virtual possibilita um aumento do espaço de armazenamento da memória, fazendo com que pareça que a memória parece que tenha um armazenamento maior do que realmente tem.
2. As causas do thrashing são diversas, como memória física insuficiente; gerenciamento de memória precário feita pela aplicação ou pelo SO, sendo ineficiente; e aplicações que demandam muita memória, causando uma paginação excessiva. Ele é detectado pelo sistema quando há um frequente número de page faults, quando as cores da CPU estão sendo utilizadas ao máximo, assim como a alta atividade no disco. Além disso, verifica-se uma queda de performance do SO. Outra causa pode estar associada ao tamanho da página a ser alocada. Se for muito grande, pode cobrir muitas localidades desnecessárias, se for muito pequena, a localidade desejada pode não ser alocada. Assim, pode haver uma paginação excessiva desnecessária. Para eliminar o problema, o sistema pode alocar dinamicamente as páginas proporcionalmente ao tamanho das páginas. Assim, ocorre apenas a quantidade suficiente de paginações.
3. 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
4. 9 faltas de página
5. 15 faltas de página

1. 12 faltas de página

14. a) 8 faltas de página

b) 10 faltas de página

c) 8 faltas de página

15. a) No instante (T + 3), (T + 4) e (T + 5), pois nesses instantes houveram referenciação de um endereço na página com page fault. Como a memória principal estava cheia, então como ocorreu page fault, obrigatoriamente ocorreu uma page out.

b) No instante (T + 1)

c) A página 3, pois ela foi a primeira a ser descartada na paginação.

d) O sistema identifica verificando o bit de validade da página.

16. O tamanho da pagina é 4KB e existem 2^20 páginas.

17. O número de páginas só depende do tamanho do campo d. O espaço de endereçamento total tem 32 bits. O que vai determinar se há mais ou menos páginas nesses 32 bits, é o tamanho de cada página. Então, se d for maior, terá menos páginas(a + b + c vai ser menor), e se d for menor, terá mais páginas(a + b + c vai ser maior).

18. tamanho da página 2^13.

número de entradas = 48 - 13 = 35

Esse esquema não funcionaria para um computador de 32 bits, pois ele poderia ter no máximo 2^32 endereços virtuais. Logo, 2^48 endereços não fariam sentido.

19. O algoritmo de Aging consiste em colocar um contador para cada processo. Nesse contador, geralmente de 8 bits, a cada interrupçao de relógio, o contador é deslocado para a direita, e, se a página tiver sido referenciada, insere-se 1 na esquerda, caso contrário, 0. O problema do algoritmo NFU, é que se houver uma página que participou de tempo de execução bastante longo, o seu contador vai ficar exorbitante, e páginas de códigos que seriam utilizadas posteriormente no processo sempre seriam removidas, pois seus contadores são menores que a página de grande tempo de execução. O aging resolve isso, fazendo com que a cada tempo de relógio, o contador seja deslocado para a direita, “esquecendo” de um bit R antigo. Assim, se por 8 tempos de relógio aquela página não for referenciada, ela estará resetada com valores nulos e poderá ser removida.