Guilherme Schievelbein – 180186

2. O que é memória virtual ? Qual a sua finalidade ?

É um método de gerenciamento de memória que permite dar endereços lógicos aos endereços físicos de memória. A memória virtual permite mapear uma área de memória maior do que a que se tem na memória principal, mantendo uma parte na memória principal e o resto em disco, trazendo as partes em disco para a memória principal quando forem necessárias.

Isso permite a execução de programas maiores do que a memória física disponível. Essa técnica também faz com que o programador não precise se preocupar com limitações de memória.

12. Responda :

a. O que é, e qual é a causa do fenômeno de *thrashing* ?

Thrashing é quando um processo gasta mais tempo paginando do que executando. Ocorre quando não são alocados páginas suficientes que um processo precisa para executar. Isso causa um alto número de page faults.

b. Como o sistema operacional pode detectar que está ocorrendo *thrashing* ? Neste caso, qual a providência que o sistema operacional pode tomar para reduzir/eliminar este problema (em tempo de execução) ?

O sistema pode detectar quando está ocorrendo thrashing analisando a quantidade de page faults que estão ocorrendo. Pode ser eliminado diminuindo o nível de multiprogramação. Quando o SO detecta thrashing ele bloqueia o processo que o está provocando, para evitar que esse processo provoque thrashing nos outros processos.

15. Explique os seguintes algoritmos de alocação de memória : First-Fit, Best-Fit eWorst-Fit ? A que tipo de gerência de memória estes algoritmos estão associados ?

First-fit: O algoritmo percorre a memória até encontrar o primeiro segmento grande o suficiente para conter o processo ao qual que ele está tentando alocar memória e carrega o processo neste segmento.

Best-fit: O algoritmo percorre toda a memória, encontra o menor segmento livre que pode conter o processo e o carrega neste segmento.

Worst-fit: O algoritmo percorre toda a memória, encontra o maior segmento disponível e aloca o processo no mesmo.

Estão associados a swap. Têm a função de alocar um processo recém criado ou vindo da área de swap à memória.

75. Responda :

a. Qual a vantagem de multiprogramação ? Para que serve ?

Multiprogramação permite que vários processos estejam na memória ao mesmo tempo, garantido que a CPU sempre tenha algo para executar. Junto com recursos de time-sharing de máguinas modernas, têm-se a ilusão que o computador executa várias tarefas ao mesmo tempo.

b. Qual a desvantagem de se ter um grau de multiprogramação muito grande ?

Um grau de multiprogramação muito grande pode ocasionar thrashing. Também prejudica a segurança do sistema, pois o processo pode acabar usando algum recurso que pertencia a outro processo.

91. Para cada uma das transições de estados de um processo listadas abaixo, indique se ela é possível ou não.

JUSTIFIQUE suas respostas e para os casos em que a transição for possível, dê UM exemplo de situação que a provoque.

a. Apto -> executando

Quando um processo termina de executar, um novo processo apto passa a ser executado

b. Bloqueado suspenso -> bloqueado

Quando o processo volta para a memória principal.

c. Executando*!*Apto

Quando um processo dá yeld, ou o alocador de processos mudou de processo executando.

d. Bloqueado-> Executando

Não. O processo bloqueado precisa primeiro ir para apto.

e. Bloqueado suspenso ->apto

Não, primeiro o processo precisaria ir para bloqueado.

f. Apto ->Término

Não, apenas um proceso executado poder terminar

g. Apto ->Bloqueado

O processo usa algum recurso de E/S.

61. Um sistema com paginação utiliza uma TLB e mantém a tabela de páginas integralmente em memória, ou seja, ela NÃO fica parcialmente no disco. O tempo de acesso a uma posição de memória realizada através de uma entrada da tabela de páginas que está na TLB (*hit*) é de 100 ns, caso contrário, isto é, para acessos feitos através de uma entrada da tabela de páginas que está em memória, o tempo de acesso é 180 ns. Pergunta-se : qual o *hit ratio* (h) necessário para se ter um tempo de acesso efetivo (médio) de 125 ns ?

h*100+(1-h)*180=125

h=0.6875

16. Em um determinado instante de tempo a memória de uma máquina apresenta as seguintes partições livres : 100K, 500K, 200K, 300K e 600K (nesta ordem). Quais partições ocuparão os processos de 212K, 417K, 112K e 426k (considerando que são submetidos a execução nesta ordem) para cada um dos três algoritmos de alocação

de mémoria ? No seu ponto de vista qual, neste caso específico, é o melhor algorítmo ?

first-fit:

212k vai para a partição de 500k, deixando 288k livres.

417k vai para a partição de 600 k, deixando 183k livres.

112k vai para a partição de 288k, deixando 176k livres.

Não sobra espaço para o processo de 426k.

Best-fit:

212k vai para a partição de 300k, deixando 88k livres.

417k vai para a partição de 500k, deixando 83k livres.

112k vai para a partição de 200k, deixando 88k livres.

426k vai para a partição de 600k, deixando 174k livres.

Worst-fit:

212k vai para a partição de 600k, deixando 388k livres.

417k vai para a partição de 500k, deixando 83k livres.

112k vai para a partição de 388k, deixando 276k livres.

Não sobra espaço para o processo de 426k.

Nesse caso, o best-fit é melhor, pois consegue alocar todos os processos sem precisar efetuar desfragmentação.

29. Suponha um algoritmo de escalonamento (curto termo) que favorece os processos que consumiram menos tempo de CPU em uma janela de tempo recente. Explique porque esse algoritmo favorece os processos I/O bound sem provocar postergação indefinida (*starvation*) nos processos CPU-bound ?

Processos I/O bound usam menos CPU, isso faz com que passem ao estado executando mais rapidamente quando saem do estado bloqueado(terminaram a operação de entrada ou saída). Não prejudicam os processos CPU-bound pois os processos I/O bound ficam muito tempo

bloqueados, dando opurtunidade para a execução dos processos CPU bound. Caso os processos I/O bound começarem a usar muita CPU, eles passarão a ser CPU bound, e terão menos prioridade.

67. Responda : Em um sistema com *m* processos e *n* CPUs, determine :

a. O número máximo de processos que podem estar nos estados apto, executando e bloqueado ? Determine as circunstâncias em que essas situações ocorrem.

Aptos: m-n processos. n processos sendo executados e os processos restantes aptos.

Executando: n processos executando. Cada CPU executa um processo.

Bloqueado: m – n processos. Um processo deixa de executar para dar lugar a outro processo. Caso todos outros processos estejam bloqueados, ele continua executando, mesmo se estiver efetuando E/S.

b. O número mínimo de processos que podem estar nos estados apto, executando e bloqueado ? Determine as circunstâncias em que essas situações ocorrem.

Aptos: 0 processos. Todos os outros processos estão executando ou bloqueados.

Executando: n. Conforme explicado da questão a, sempre haverá n processos executando.

Bloqueado: 0. Todos os outros processos estão executando ou aptos.

8. Os três principais estados de um processo são : *ready*, *running* e *blocked*. Descreva quais eventos fazem um processo mudar de estado. Descreva TODAS situações possíveis envolvendo estes três estados.

De Ready para Running: Processo que estava running foi para ready ou blocked, liberando processador, um processo que estava ready passa para running.

De Running para Ready: Processo rodou por tempo suficiente, escalonador libera a CPU para outro processo. Processo libera a CPU voluntáriamente via chamada de sistema.

De Running para Blocked: Processo fez requisição de I/O, vai para blocked para esperar a entrada.

De Blocked para Ready: I/O pedida pelo processo foi finalizada, processo vai para Ready e está pronto para usar o processador novamente.

11. Pode-se ter multiprogramação em ambientes monousuário ?

Sim, podemos ter um único programa (user) executando várias threads (jobs).

1. Considerando a afirmação : «um escalonador que apresenta prioridades necessariamente é preemptivo.». Responda: A afirmação é correta ? Justifique a sua resposta.

Não, ser não-preemptivo só significa que o processo não vai ter um tempo máximo para ficar sendo executado no processador, isso não interfere no sistema de prioridades. Um

escalonador não-preemptivo pode ser visto como um escalonador preemptivo com quantum infinito, logo, é possível ser não-preemptivo e ter prioridades.

5. Um sistema que emprega memória virtual utilizando paginação está executando um programa de tamanho de 8 Kbytes. A máquina possui uma memória física de 3 Kbytes e define quadros de 1 Kbytes. Na execução deste programa foi gerada a seguinte sequência de referências de páginas (*string de referência*) :

5 2 3 5 2 5 1 5 4 2 3 2 1 3 2 3

Supondo que a memória principal esteja inicialmente vazia indique quais páginas estarão residentes na memória principal após cada referência de página se a política de substituição for : (a) LRU e (b) FIFO. Qual a melhor política entre FIFO e LRU para estas referências ? Justifique sua resposta.

obs : indique com um «*» a página a ser substituída na próxima requisição em caso de ocorrer um *page fault*.

```
        5  2  3  5  2  5  1  5  4  2  3  2  1  3  2  3

FIFO    5  5  5* 5* 5* 5* 1  1  1* 2  2  2  2* 2* 2* 2*
           2  2  2  2  2  2* 5  5  5* 3  3  3  3  3  3
              3  3  3  3  3  3* 4  4  4* 4* 1  1  1  1
        F  F  F           F  F  F  F  F     F
```

```
LRU   5  5  5* 5  5  5  5  5  5  5* 3  3  3* 3  3  3
        2  2  2* 2  2  2* 2* 4  4  4* 4* 1  1  1* 1*
           3  3  3* 3* 1  1  1* 2  2  2  2  2* 2  2
      F  F  F           F     F  F  F     F
```

LRU é melhor para essa sequência de referências, pois resulta em um page fault a menos.

18. Responda :

a. O que é *working set* ?

É o conjunto de páginas sendo usadas por um processo num determinado instante.

b. Determine o *working set* nos instantes *t*1 e *t*2 considerando uma janela de _ = 5 para a seguinte sequência de acessos a páginas : 2 6 1 5 7 7 7 7 5 1 *t*1 6 2 3 4 1 2 3 4 4 4 4 3 4 3 4 4 4 *t*2 1 3 2 3 4

t1 = {1, 5, 7}

t2 = {3, 4}

3. Um tipo de escalonador empregado em sistemas operacionais é o de múltiplas filas com realimentação (filas multinível). A idéia neste caso é classificar os processos de acordo com suas diferentes necessidades de CPU e de I/O (CPU bound e I/O bound) criando diversas filas com prioridades diferentes. Se um processo utiliza muito

tempo de CPU ele é transferido de uma fila de mais alta prioridade para uma fila de mais baixa prioridade.

Responda :

a. Este esquema privilegia os processos I/O bound ? Em caso afirmativo, qual a argumentação (justificativa) para este tipo de privilégio ?

Sim. O motivo para isso é que processos I/O bound usam pouca CPU e então voltam para o estado bloqueado. Desse modo, os processos I/O bound não são prejudicados tendo que esperar muito tempo depois de ficarem aptos, uma vez que já ficaram por um tempo bloqueados e não irão usar a CPU por muito tempo.

b. Neste esquema existe a probabilidade de um processo CPU bound sofra «starvation» (postergação infinita)? Caso esta possibilidade seja real como pode ser resolvido este problema ?

Caso haja um grande número de processos I/O bound, pode ocorrer starvation. Pode ser resolvido fazendo com que o escalonador reclassifique os processos após muito tempo se executar

c. Qual critério é empregado para classificar um processo nas diversas filas quando ele é criado ?

Quando criados são colocados na mesma fila, a mudança de filas ocorre durante a execução, conforme o tempo que o processo usa o processador.

d. Como o sistema operacional pode determinar se um processo é CPU bound ou I/O bound, movendo-o de uma fila para outra, se esta característica só pode ser conhecida em tempo de execução ?

O SO também pode levar em conta de que tipo o processo pai era. Colocando o processo na mesma fila que o processo que o criou.

13. Justifique o porquê da existência de uma área em disco específica (partição) para o *swap*? (Pense nas diferenças entre reler páginas de código do sistemas de arquivos e de reler estas páginas à partir do espaço de *swap*).

Existe por questão de eficiência. A área de swap é usada pela memória virtual, como se fosse uma extensão da memória física. Por isso é muito mais rápido trazer para a memória algo que está no swap do que do hd, pois o sistema traduz o endereçamento, e caso esteja na memória virtual, traz os dados para a memória física, sem perder tempo lendo as páginas do sistema de arquivos, que é um processo custoso.

25. Explique, CLARA e DETALHADAMENTE, quando são necessários e quais as funções dos escalonadores de curto termo, médio termo e longo termo.

Escalonador de curto termo: Seleciona da memória os processos que estão prontos para executar a aloca a CPU para eles.

Escalonador de médio-termo: Remove um processo executado parcialmente da memória, para ser continuar executando de onde parou mais tarde. Usado com técnicas de compartilhamento de tempo.

Escalonador de longo-prazo: Determina que processos são trazidos para a memória para serem processados.

106. Um escalonador hipotético de curto prazo favorece os processos que tenham usado o menor tempo de processador em uma janela de tempo _*t* em um passado recente. Explique por quê esse algoritmo favorece os processos do tipo I/O bound sem causar postergação indefinida de processos CPU bound.

Desse modo, os processos I/O bound serão executados primeiro. Mas justamente por serem I/O bound, não usarão a CPU por muito tempo, e passarão para o estado bloqueado, dando opurtunidade para que os processos CPU bound sejam executados.

99. Responda :

a. O que é seção crítica ? Em que situações elas aparecem ? Porque é importante que um processo (thread) execute uma seção crítica o mais rapidamente possível ? Que mecanismo(s) pode(m) ser usado(s) para proteger uma seção crítica ?

A sessão crítica é uma região de código que pode ser executado por apenas uma thread por vez. Ocorre em partes em que váriaveis comuns são alteradas. É importante que seja executada o mais rápido possível pois quando uma thread está executando uma seção crítica, todas as outras thread não podem executar suas seções críticas. Podem ser usados semáfaros e MUTEX.

b. O que acontece quando um processo executa uma operação V sobre um semáforo *S* sem ter previamente executado uma operação P sobre esse semáforo ?

O semáforo deixará de funcionar corretamente. A operação P decrementa o valor do semáforo, e a operação V incrementa o valor do semáforo. P é usado quando vai entrar na seção crítica e V e usado quando terminou de executar a secção crítica. Como o valor do semáforo vai ficar maior do que realmente deveria ser, será possível threads possam acessar essa seção, mesmo quando essa operação deveria ser bloqueada.

65. Responda :

a. Qual a função dos estados de suspenso no diagrama de estados de um processo ? Qual nível de escalonamento (curto, médio ou longo prazo) eles estão associados ?

Um processo suspenso é um processo que foi removido da memória física e armazenado na memória virtual. Está associado ao escalonamento de médio prazo.

b. Uma política de escalonamento do tipo SJF é uma forma de escalonamento por prioridades ? JUSTIFIQUE sua resposta.

Sim. Desse modo processos com menor tempo tem prioridade maior.