SISTEMAS OPERACIONAIS

INF09344 - Sistemas Operacionais

Prof. José Gonçalves / Profa. Roberta Lima Gomes

1ª. Lista de Exercícios – Processos/Escalonamento

***Data de Entrega: não há**. O objetivo da lista é ajudar no estudo individual dos alunos. Soluções de questões específicas poderão ser discutidas em sala de aula, conforme interesse dos alunos.*

1. Qual a relação entre programa e processo?
2. O que é o BCP? Qual é o seu conteúdo típico?
3. O que é a "imagem" de um processo?
4. Qual é o propósito das *chamadas de sistema* (SVC)?
5. Explique as funções dos escalonadores de curto, médio e longo prazo.
6. Defina *turnaround time*, *waiting time* e *throughput*.
7. O que significa um processo sofrer preempção?
8. A maioria dos escalonadores *round-robin* (escalonamento circular) usa um *quantum* de tamanho fixo. Dê um argumento em favor de um quantum pequeno. Agora pense em um argumento que justifique um *quantum* grande.
9. Discuta o efeito de cada um dos seguintes métodos de atribuição de *quantum* q no escalonamento circular (round-robin).
   a) q fixo e idêntico para todos os usuários
   b) q fixo e único para cada processo
   c) q variável e idêntico para todos os processos
   d) q variável e único para cada processo
10. Considere o seguinte algoritmo de alocação por prioridade, preemptivo, baseado em prioridades que mudam dinamicamente. Números de prioridades maiores indicam prioridades mais altas. Quando um processo está esperando para entrar em execução (na fila de prontos), sua prioridade muda segundo uma taxa a; quando está em execução, sua prioridade muda segundo uma taxa b. Todos os processos têm prioridade a mesma prioridade quando são criados. Valores diferentes para os parâmetros a e b podem determinar muitos algoritmos de alocação diferentes.
    a) Qual algoritmo é obtido com $b > a > 0$?
    b) Qual algoritmo é obtido com $a < b < 0$?
11. Como funciona o exemplo escalonamento por múltiplas filas com realimentação? Qual a relação entre o *quantum* e o nível de prioridade de cada fila nesse esquema?
12. Dois objetivos comuns das políticas de escalonamento são minimizar tempos de resposta e maximizar utilização de recursos.
    a) Indique como esses objetivos se contrapõem um ao outro.
    b) Analise as políticas de escalonamento circular (*round robin*) e múltiplas filas com retorno/realimentação (*multiple queues with feedback*) com base nessas duas perspectivas.
13. Suponha um S.O. com escalonador de filas multiníveis na qual há cinco níveis. O quantum do primeiro nível é 0,5 segundos. Cada nível mais baixo tem um quantum de tamanho duas vezes

maior que o quantum do nível anterior. Um processo não pode sofrer preempção até o seu quantum terminar. O sistema executa processos em lote (*cpu bound*) e interativos (*I/O bound*).

a) Por que esse sistema é deficiente?

b) Quais mudanças mínimas você proporia para tornar o esquema mais aceitável para o mix de processos que pretende?

14. Qual dos algoritmos de escalonamento discutidos em sala de aula poderia ser modificado para acomodar alguns processos de tempo real (processos que *devem* ter uma resposta dentro de um certo período de tempo) misturados com os outros tipos de processo? Para aqueles algoritmos que você poderia modificar, explique como você faria isso. Para aqueles que isso não é possível, explique o porquê.

15. Cinco processos, de A até E, chegam ao computador ao mesmo tempo. Eles têm seus tempos de processamento estimados em 10, 6, 2, 4 e 8 minutos respectivamente. Suas prioridades (atribuídas externamente) são 3, 5, 2, 1 e 4, respectivamente, sendo 5 o representante da prioridade mais alta. Nenhum dos processos faz I/O. Para cada um dos algoritmos de escalonamento abaixo, determine o tempo médio de turnaround dos processos. Ignore o overhead causado pela troca de contexto.

a) Round Robin (fila começa em A, indo em ordem até E ; quantum = 4)

b) Escalonamento com prioridade

c) FIFO (ordem de execução: A, B, C, D, E)

d) SJF

16. Suponha que os processos seguintes fiquem prontos para execução nos tempos indicados:

Processo	Duração da fase de uso da CPU	Tempo de chegada
P1	8	0
P2	4	3
P3	1	4

a) Desenhe diagramas de Gantt ilustrando a execução destes processos usando os algoritmos de alocação FCFS, SJF e STRF (SJF preemptivo).

b) Qual o tempo de processamento (*turnaround*) médio para esses processos em cada um desses algoritmos?

17. Em um sistema operacional, o escalonador de curto prazo está organizado como duas filas, a fila A contém os processos do pessoal do CPD e a fila B contém os processos dos alunos. O algorítmo entre filas é round-robin. De cada 11 unidades de tempo de cpu, 7 são fornecidas para os processos da fila A e 4 para os processos da fila B. O tempo de cada fila é dividido entre os processos também por round-robin, com fatias de tempo de 2 unidades para todos. A tabela abaixo mostra o conteúdo das duas filas no instante zero. Considere que está iniciando um ciclo de 11 unidades, e agora a fila A vai receber as suas 7 unidades de tempo. Mostre a sequência de execução dos processos, com os momentos em que é feita a troca (diagrama de Gantt).

OBS: Se terminar a fatia de tempo da fila X no meio da fatia de tempo de um dos processos, a cpu passa para a outra fila. Entretanto, este processo permanece como primeiro da fila X, até que toda sua fatia de tempo seja consumida.

Fila	Processo	Duração do próximo ciclo de CPU
A	P1	6
A	P2	5
A	P3	7
B	P4	3
B	P5	8
B	P6	4

18. Quatro programas devem ser executados em um computador. Todos os programas são compostos por 2 ciclos de cpu e 2 ciclos de entrada e saída. A entrada e saída de todos os programas é feita sobre a mesma unidade de disco. Os tempos para cada ciclo de cada programa são mostrados abaixo:

Programa	CPU	Disco	CPU	Disco
P1	3	10	3	12
P2	4	12	6	8
P3	7	8	8	10
P4	6	14	2	10

Construa um diagrama de tempo mostrando qual programa está ocupando a CPU e o disco a cada momento, até que os 4 programas terminem. Suponha que o algoritmo de escalonamento utilizado seja round-robin com fatias de tempo de 4 unidades. Qual a taxa de ocupação da CPU e do disco?

19. O que acontece com as duas taxas de ocupação calculadas no problema anterior se for utilizado um disco com o dobro da velocidade de acesso (duração dos ciclos de e/s é dividida por 2)?
20. Considere um sistema operacional cuja máquina de estados inclui os estados *Ready* e *Ready-Suspended*. Suponha que seja hora do S.O. despachar um processo e que existam nesse momento processos tanto no estado *Ready* como no estado *Ready-Suspended*, e que pelo menos um processo no estado *Ready-Suspended* possui prioridade maior do que qualquer processo no estado *Ready*. Duas políticas extremas seriam: *(a)* sempre despachar um processo no estado *Ready*, de forma a minimizar *swapping*; e *(b)* sempre dar preferência ao processo de mais alta prioridade, mesmo que isso possa significar a ocorrência de *swapping* quando este não é necessário. Sugira uma política intermediária (explique e crie um algoritmo) que tente balancear prioridade e desempenho.
21. Considere um sistema que possui duas filas de escalonamento, com prioridades 0 e 1, sendo que somente pode ser escalonado um processo da fila de prioridade 1 não existindo processos na fila de prioridade 0. Sabendo que o algoritmo utilizado nas duas filas é o Round-Robin, escreva o pseudo-código dos procedimentos *insere(p)*, onde p é o índice da tabela de descritores de processos e que possui um campo que contém a prioridade dos processos e *r= seleciona()*, que retorna o índice da tabela de descritores que descreve o processo selecionado. Cite duas situações em que cada procedimento é chamado.
22. O sistema operacional Linux utiliza dois algoritmos de escalonamentos, um de tempo compartilhado e outro baseado apenas em prioridades, para tratar tarefas de tempo real. Cada

processo é associado a uma classe de escalonamento. No algoritmo de escalonamento para processos de tempo compartilhado, o Linux adiciona um algoritmo por prioridades, baseado em créditos:

- Cada processo possui um determinado número de créditos (inicialmente, o número de créditos é igual à prioridade do processo);
- O processo com o maior número de créditos na fila de prontos é selecionado pelo escalonador.
- A cada interrupção do temporizador (1ms), o processo em execução perde um crédito; quando seu crédito chega a zero, o escalonador é ativado para selecionar outro processo para ganhar o processador.
- Se nenhum processo na fila de prontos tiver créditos, o algoritmo faz nova atribuição de créditos a todos os processos (inclusive aos processos bloqueados), de acordo com a seguinte regra: créditos := créditos /2 + prioridade ("arredonda p/ cima")

Analise o algoritmo de escalonamento por prioridade usado pelo Linux considerando o seguinte volume de trabalho (processos de A a D chegam no sistema ao mesmo tempo nessa ordem):

Ordem	Processo	Surto de CPU*	Duração de I/O	Tempo total de CPU	Prioridade
1	A	1	4	3	2
2	B	2	6	4	2
3	C	---	---	6	2
4	D	---	---	6	2

(*) tempo de CPU necessário antes de cada solicitação de I/O (processos A e B ficam alternando entre surtos de CPU e em operações de I/O).

Ilustre a evolução no tempo do *(i) estado* e do *(ii) número de créditos* de cada processo. Utilize a seguinte nomenclatura de estados: *Pronto* (P); *Rodando* (R); *Bloqueado* (B); *Terminado* (T). Por exemplo:

- "P2" indica que o processo está no estado *Pronto* com 2 créditos.
- "R0" indica que o processo está no estado *Rodando* consumindo o seu último crédito

23. Com relação ao escalonador tradicional do Unix:
    a) Por que ele favorece processos I/O bound?
    b) Por que ele não é adequado a processos de tempo real?
    c) Por que ele não é escalável?
24. Em algumas implementações do UNIX, o kernel é não-preemptivo. O que isto significa? Quais as vantagens e desvantagens desta abordagem?
25. No UNIX, um processo pode encontrar-se no estado *kernel running* enquanto o sistema pode apresentar dois contextos de execução, *process context* e *system context*. Explique a diferença entre eles.