Rafaela Carneiro

Matrícula 2011483

1) Vantagem: podemos fazer os processos rodarem em paralelo e garantir que eles não acessem a região crítica ao mesmo tempo.

Desvantagens: desperdiça tempo de CPU, porque um processo pode ficar muito tempo na espera; e o problema da inversão de prioridade: considerando dois processos A e B, onde A tem maior prioridade do que B, dependendo das regras de escalonamento, A é executado sempre que estiver em estado pronto. Nesse caso, se B entra na região crítica e A fica pronto para executar, A começa espera ocupada e, considerando que B não é escalonado enquanto A estiver executando, A fica em um laço infinito.

3)

Com uso de semáforos:

Caso 1: programa 1 roda primeiro

Programa 1 faz a função down(s), bloqueando a entrada de outros programas na região crítica; coloca valor 1 na variável x; imprime p1, 1 e 3; e faz a função up(s) para liberar a entrada na região crítica e para de executar. Em seguida, como agora o programa 2 consegue entrar na região crítica, ele executa down(s) bloqueando a entrada do prog1 na região crítica e coloca o valor de x+y na variável y. Como nesse momento x=1 e y=3, agora y=4. Programa 2 imprime p2, 1 e 4 e executa up(s) para liberar a entrada na região crítica.

Caso 2: programa 2 roda primeiro

Prog2 faz a função down(s), bloqueando a entrada do prog1 na região crítica; coloca valor x+y na variável y; como x=0 e y=3, y=3; imprime p2, 0 e 3 e faz a função up(s) para liberar a entrada de prog1 na região crítica. Agora, prog1 consegue entrar na região crítica e executa down(s) (para bloquear a entrada de prog2 na memória compartilhada), coloca valor 1 na variável x, imprime p1, 1 e 3 e faz a função up(s) para liberar a entrada na região crítica.

Sem uso de semáforos, os dois programas podem entrar na região crítica ao mesmo tempo, de modo que um programa pode sobrescrever o valor que o outro programa atribuiu à variável da memória compartilhada antes deste ser impresso. Por exemplo: prog2 faz y=x+y no momento que x=0 e y=3; logo em seguida prog1 acessa a memória compartilhada e faz x=1; quando a print() é executada por prog2 o valor impresso seria p2, 1 e 3, por que foi sobrescrito valor 1 sobre o valor 0 de x. Também pode acontecer de o prog2 sobrescrever o valor a ser impresso por prog1: nesse caso, se prog1 começa a executar e faz x=1 e logo em seguida prog2 entra na região crítica também, ele faz

y=x+y (e y fica com valor 4). O valor a ser impresso por prog1 será p1, 1 e 4, diferente do esperado com semáforos, porque o valor da variável y foi sobrescrito pelo prog2.

4) Um desses métodos é o escalonamento por múltiplas filas com feedback. Nesse caso, dependendo da quantidade de tempo que um processo precisa, o quantum dele aumenta nas próximas vezes que ele é escalonado.

5) a) Quick fit: porque não tem que percorrer a lista toda vez que for alocar.

b) Worst fit: porque sempre preenche o máximo do espaço que ele precisa, sem deixar várias porções de pequenas fragmentações.

c) Best fit: porque sempre é escolhido o espaço com tamanho que melhor se adequa.

6)

c) o sistema faz swap out do programa da memória e swap in no disco

7) Utilizaria processos, memória compartilhada, exclusão mútua e semáforos.

Um processo representa o produtor: ele produz um item; usa os semáforos para verificar se pode colocar o item na memória compartilhada (se tem espaço) e se ele pode acessar a memória compartilhada (ou se algum processo já está acessando); coloca o item na memória compartilhada, incrementa o semáforo que indica quantos produtos têm na memória compartilhada e libera o acesso a ela.

O outro processo representa o consumidor: ele usa os semáforos para ver se têm produto na memória compartilhada e se o acesso a ela está liberado; se conseguir acessar, ele consome produto, incrementa o semáforo que conta a quantidade de espaço vazio na memória compartilhada e libera o acesso a ela.

Para isso são precisos 3 semáforos: 1 contador para controlar quanto de espaço da memória está preenchido, 1 contador para controlar quanto de espaço da memória está vazio e 1 mutex para controlar a entrada dos processos na região crítica.

A utilização da memória compartilhada é feita para armazenar os produtos; a utilização da exclusão mútua (semáforo mutex) para impedir que dois processos acessem a memória ao mesmo tempo; e os semáforos contadores para impedir que o consumidor tente consumir um produto quando não tem nada pronto, ou um produtor tente produzir sem ter espaço para armazenar o produto. Seriam utilizados processos lugar de threads por conta da maior facilidade de implementação (o escalonamento dos processos é feito pelo sistema operacional).