Sistemas Operacionais

Atividade 04

1. Marque a(s) alternativa(s) INCORRETA(S). (1 Ponto)

( ) Não é possível ocorrer starvation em um sistema monoprocessado baseado em escalonamento não preemptivo.
(X) O throughput proporcionado por um escalonador é medido pelo número de vezes que uma tarefa passa pelo processador em um intervalo de tempo.
( ) Não é possível ocorrer starvation em um sistema multiprocessado baseado em escalonamento preemptivo.
(X) O critério de escalonamento turnaround objetiva maximizar o tempo de execução dos processos do sistema.

2. Marque a(s) alternativa(s) CORRETA(S). (1 Ponto)

(X) No escalonamento baseado em múltiplas filas (com realimentação) as prioridades são dinâmicas.
( ) No escalonamento baseado em múltiplas filas (sem realimentação) os processos curtos tendem a possuir maior timeslice e prioridade.
( ) O sistema operacional Linux utiliza um algoritmo de escalonamento cooperativo, portanto, do tipo preemptivo.
( ) O sistema operacional Windows 10 utiliza um algoritmo de escalonamento cooperativo, tal como o Linux.

3. Normalmente, quem tem maior prioridade, processos CPU-Bound ou IO-Bound? Justifique sua resposta. (1 ponto)

Os processos IO-Bound possuem mais prioridade. Os processos IO-Bound executam mais operações de entrada e saída (E/S), ficando boa parte do tempo em estado de espera. Para compensar o tempo de espera, assim

que um processo IO-Bound se torna pronto, ele tem prioridade para assumir a CPU. O processo CPU-Bound, por fazer menos operações de E/S, podem ser executados enquanto um processo IO-Bound está no estado de espera.

4. O que é processor affinity e quais suas vantagens e desvantagens? (1 Ponto)

Processor affinity é a associação de um processo ou thread a uma CPU ou coleção de CPUs específica. Dessa forma, o processo ou thread será executada sempre na CPU especificada. Ao usar a afinidade de processador, o fato de o processo ou thread utilizar a mesma CPU faz com que a chance dos dados armazenados em cache aumente, acelerando a execução (não precisa buscar na memória principal). Porém, como desvantagem, tem-se que, se a CPU designada estiver ocupada e outra CPU estiver livre, o processo não poderá utilizar a CPU livre, visto que existe a restrição imposta ao processo de executar somente na CPU designada. Processor affinity é a associação de um processo ou thread a uma CPU ou coleção de CPUs específica. Dessa forma, o processo ou thread será executada sempre na CPU especificada. Ao usar a afinidade de processador, o fato de o processo ou thread utilizar a mesma CPU faz com que a chance dos dados armazenados em cache aumente, acelerando a execução (não precisa buscar na memória principal). Porém, como desvantagem, tem-se que, se a CPU designada estiver ocupada e outra CPU estiver livre, o processo não poderá utilizar a CPU livre, visto que existe a restrição imposta ao processo de executar somente na CPU designada.

5. Qual a system call no Linux para alterar a prioridade de um processo. Descreva e explique cada um dos seus argumentos de entrada e valores de retorno, caso existam. (1 Ponto)

Para alterar a prioridade de um processo, pode-se utilizar a system call setpriority( ). Esta system call utiliza 3 parâmetros: int which, id_t who, int prio.

- int which: é o valor que identifica o processo e pode assumir 3 valores: PRIO_PROCESS (ID do processo - campo pid do descritor de processos), PRIO_PGRP (ID do grupo - campo pgrp do descritor de processos) ou PRIO_USER (ID do usuário - campo uid do descritor de processos).
- id_t who: é o valor do campo pid, pgrp ou uid (dependendo do valor do which) e, se who = 0, é atribuído o valor do campo correspondente ao processo atual. - int prio: é o novo valor da prioridade que pode ir de -20 (maior prioridade) a +19 (menor prioridade).

De acordo com o Man Linux , a system call setpriority retorna zero se a operação foi bem sucedida e, em caso de erro, retorna -1 e configura o número do erro (errno) para indicar a causa do erro.