Escalonamento da CPU

O objetivo do escalonamento da CPU é designar os processos a serem executados pela(s) CPUS(s) ao longo do tempo, tendo como base os objetivos definidos para o sistema operacional e sua política de escalonamento, o escalonador divide o tempo da CPU entre diversos processos ativos nos sistema (processos de usuário/sistema)
Existem três tipos básicos para o escalonamento:

- Escalonamento de longo prazo (“long-term scheduling”) - adiciona processos ao pool de processos a serem executados, determinando quais programas serão admitidos para o processamento do sistema,´permitindo o controle da carga do sistema (new→ready);
- Escalonamento de médio prazo ("medium-term scheduling")- controla o swap in e swap out de processos, atendendo às flutuações de carga do sistema, ligado a gerência e de memória, fazendo parte da função de troca de processos (swapping) entre a memória principal e a memória secundária.
- Escalonamento de curto prazo ("short-term scheduling" ou "dispatcher")- responsável pela troca de contexto e alocação efetiva da CPU ao processo selecionado, executando uma decisão sobre qual será a próxima tarefa a ser executado (ready→running), deve ser rápido e eficiente;

Critérios de Escalonamento

- Maximizar a taxa de utilização da UCP;
- Maximizar a vazão (“throughput”) do sistema;
- Minimizar o tempo de execução (“turnaround”);
- Turnaround: tempo total para executar um processo;
- Minimizar o tempo de espera (“waiting time”);
- Waiting time: tempo de espera na fila de prontos;
- Minimizar o tempo de resposta (“response time”);
- Response time: tempo entre requisição e resposta (processos interativos).

Políticas de Escalonamento

Preemptivas

A posse da CPU pode ser tomada do processo em execução a qualquer momento, ou seja, o processo pode perdê-la compulsoriamente na ocorrência de certos eventos, como fim de fatia de tempo ou processo mais prioritário torna-se pronto para execução, não permitindo a monopolização da CPU.

Não Preemptivas

O processo em execução só perde a posse da UCP caso termine ou a devolva deliberadamente, isto é, uma vez no estado running, ele só muda de estado caso conclua a sua execução ou bloqueie a si mesmo (ex. chamada de sistema para fazer uma operação de E/S.)

Algoritmos de Escalonamento

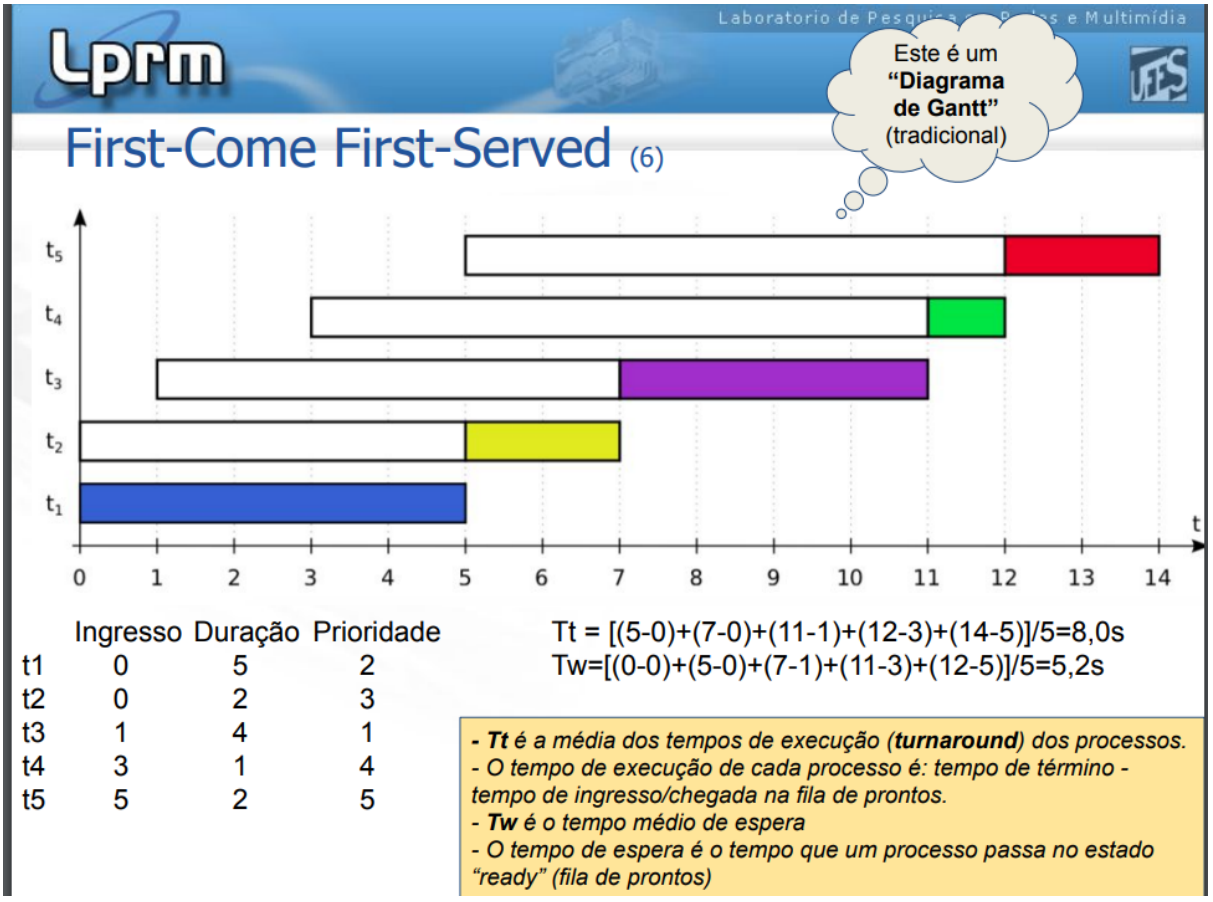
FIFO (First-In First-Out) ou FCFS (First-Come First-Served)

É um algoritmo com baixa complexidade (não preemptiva) e muito utilizado em sistemas Batch, Processos que se tornam aptos para execução são inseridos no final da fila de processos prontos e o primeiro processo da fila é selecionado para a execução, como é não preemptiva, irá executar o processo até que termine ou realize uma chamada de sistema.

Ele oferece maior apoio a processos com CPU-bound, para os processos I/O-bound tem que esperar muito até que os de CPU-bound termine a sua execução.

Como problema, há o “Convoy effect”, que é quando um processo pequeno tem que esperar um longo período de tempo até que seja executado, caso os processos dá fila (que estão à frente dele) sejam longos, e é problemático para sistemas de tempo compartilhado, em que os usuários precisam da CPU com intervalos regulares.

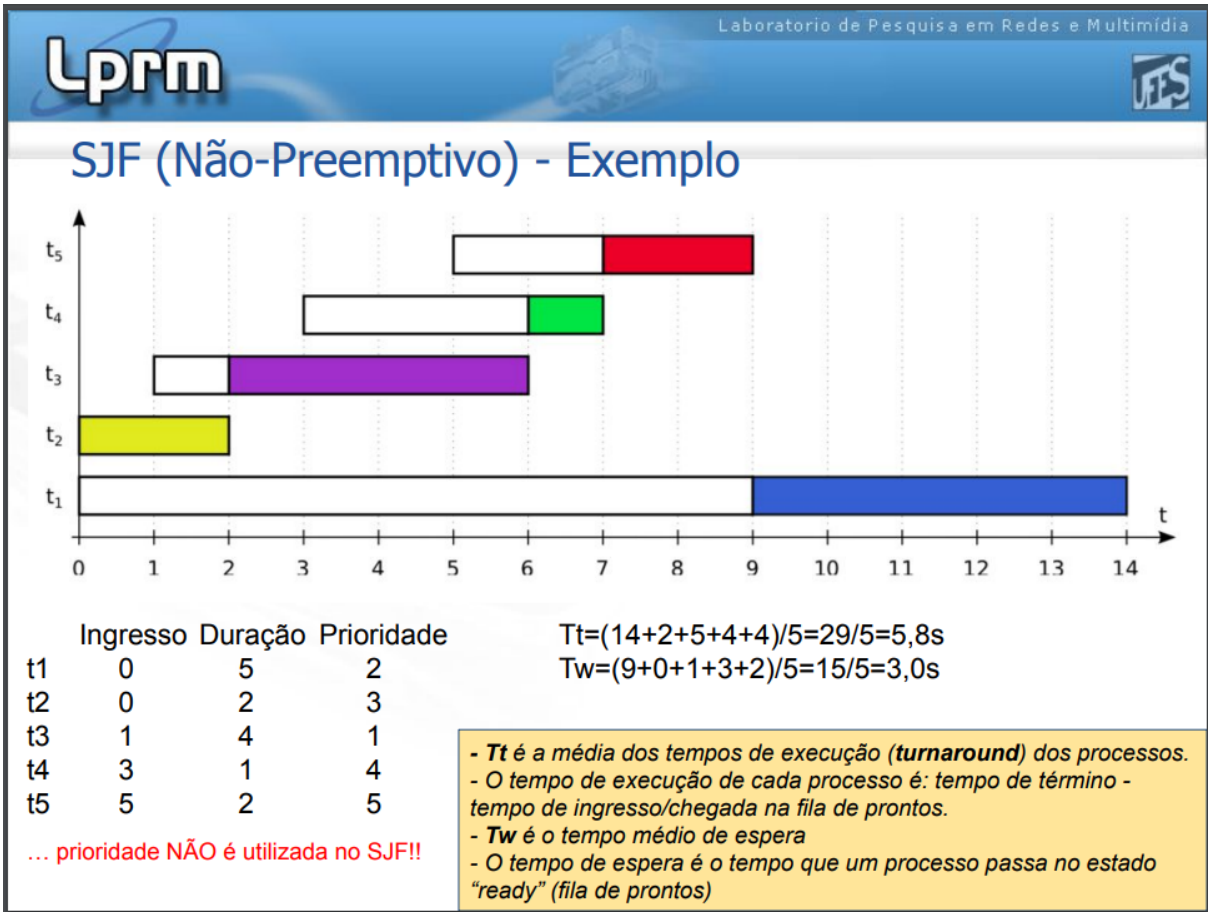
Formula:

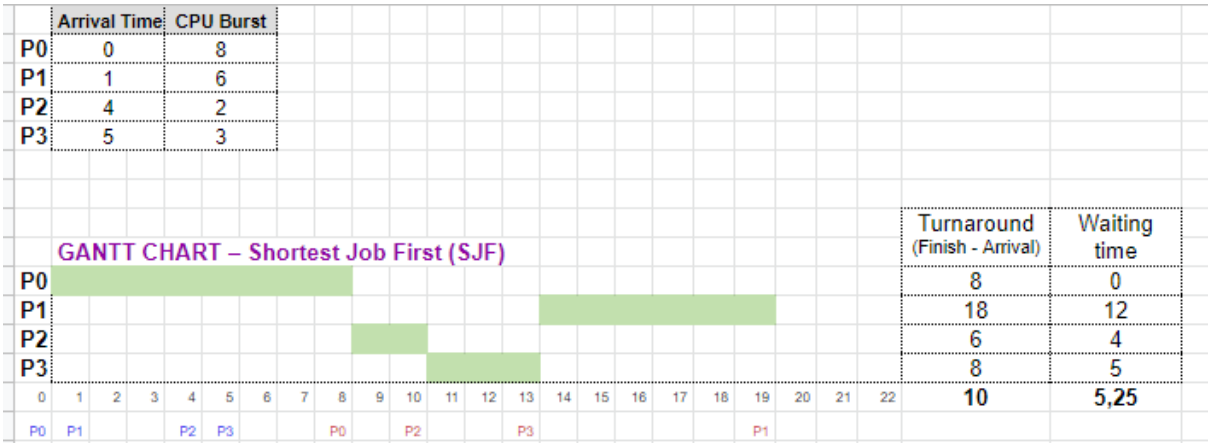


Shortest Job First - SJF

Se baseia no fato de que privilegiando processos pequenos o tempo médio de espera decresce, sendo assim, o tempo de espera dos processos pequenos decresce mais do que o aumento do tempo de espera dos processos longos.

SJF Não-preemptivo – uma vez a CPU alocada a um processo ela não pode ser dada a um outro antes do término do CPU burst corrente.
O menor processo tem prioridade e como ele não sabe qual é o menor processo para se iniciar, ele pressupõe que o menor processo seja o que chegou primeiro e o que será executado em seguida, será o que tem menor tempo entre os processos.

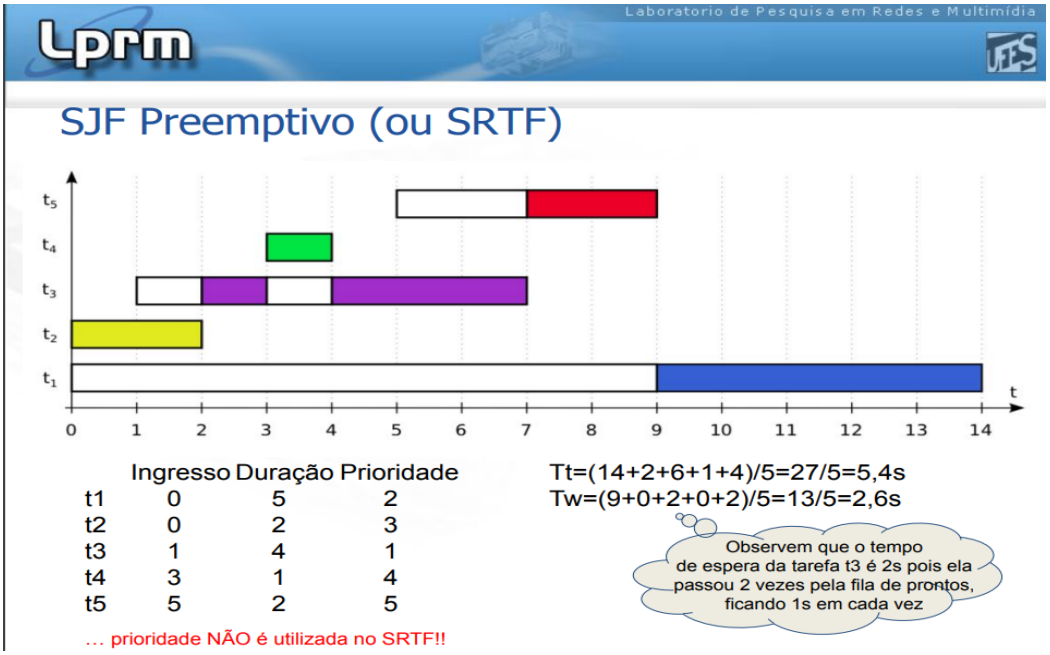


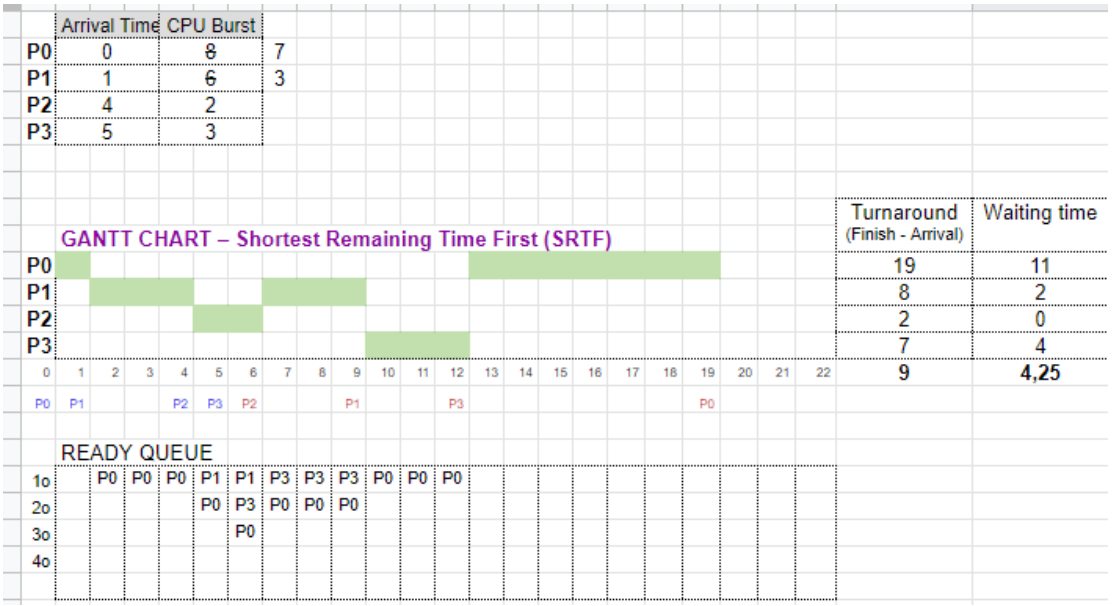


Turnaround = Final (Na Gantt Chart) - Arrival Time
Waiting Time = Inicio (Na Gantt Chart) - Arrival Time

Preemptivo – se chega um novo processo com CPU burst menor que o tempo remanescente do processo corrente ocorre a preempção. Esse esquema é conhecido como Shortest-Remaining-Time-First (SRTF)

Tomamos o primeiro como início até alcançar o tempo de chegada dá unidade do menor da fila de processos, interrompendo em X unidades, e o processo que parou irá voltar ao estado running quando alcançar Y unidades, ou seja, quando alcançar o tempo de execução daquele processo.





Turnaround = Final (Na Gantt Chart) - Arrival Time
Waiting Time = Início (Na Gantt Chart) - Arrival Time (Processo normal)
Waiting Time = n vezes que passou pela fila - Arrival Time (Processo normal)

Algumas perguntas referente a aula:

Explique as funções dos escalonadores de curto, médio e longo prazo.
Resposta:

Escalonamento a longo prazo: Determina que programas são admitidos para processamento no sistema;

Escalonamento a médio prazo: faz parte da função de troca de processos (swapping) entre a memória principal e a memória secundária;

Escalonamento a curto prazo: executa uma decisão sobre qual será a próxima tarefa a ser executada;

Qual a relação entre Tempo de Espera e Tempo de Resposta?
Resposta:

Tempo de Espera é o tempo total perdido pela tarefa na fila de tarefas prontas, aguardando o processador, esse tempo não inclui os tempos de espera em operações de entrada/saída, e o tempo de resposta, é o tempo decorrido entre a chegada de um evento ao sistema e o resultado imediato de seu processamento. Essa medida de desempenho é típica de sistemas interativos, como sistemas desktop e de tempo real; ela depende sobretudo da rapidez no tratamento das interrupções de hardware pelo núcleo e do valor do quantum de tempo, para permitir que as tarefas interativas cheguem mais rápido ao processador quando saem do estado suspenso, logo a sua relação com o tempo de espera está relacionado com os valores de quantum baixos, para que cada tarefa pronta receba rapidamente o processador, promovendo assim, maior interatividade.

O que significa um processo sofrer preempção?
Resposta:

É quando o SO se admite interrupção de uma ordem/processo de em execução, para processamento de outra, características do escalonamento preemptivo

Há dois tipos principais de preempção, por tempo e por prioridade. Na preempção por tempo o processo é retirado do processador porque foi atingido o tempo determinado. Já na preempção por prioridade o processo sai para entrar outro que está na fila de pronto que possui prioridade maior.

A maioria dos escalonadores Round Robin 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.
Resposta:

Quantum pequenos gera muito overhead devido às sucessivas trocas de contexto baixando a eficiente o processador, mas auxilia nos processos que necessitam de chamadas freqüentes.Já os quantum maiores tende a FIFO seriam interessantes para aqueles processos que não necessitam de chamadas freqüentes de execução, pois possui um tempo de resposta ruim para usuários interativos.

Considere o seguinte algoritmo de alocação de CPU (escalonamento) 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 α; quando está em execução, sua prioridade muda segundo uma taxa β. Todos os processos têm a mesma prioridade quando são criados. Valores diferentes para os parâmetros e podem determinar muitos algoritmos de alocação diferentes.

a) Qual algoritmo é obtido com α > β > 0?

Resposta:

Algoritmo RR, a maior prioridade iria está no processo que está na fila de prontos, o α , e a menor prioridade estaria no processo que está em execução, no caso o β , isso fará com que o processo que está em execução [β] seja preemptado a todo momento funcionando como um RR.

b) Qual algoritmo é obtido com β > α > 0?

Resposta:

Algoritmo FIFO, a maior prioridade em execução seria o [β], pois o β é maior que α é maior que zero, o processo estaria com a priori*dade mais alta que os processos prontos que estão aguardando na fila, funcionando como um algoritmo FIFO.*

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), processos de um mesmo tipo para um mesmo cálculo e não precisa esperar evento externo, e interativos (I/O bound)

a) Por que esse sistema é deficiente?

Resposta:

Em I/O Bound , ficam a maior parte do tempo no estado de espera esperando um evento externo. No caso deste algoritmo ficariam ocupando o processador e seus registradores sem realizar nenhum tipo de cálculo.

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

Resposta:

Proporia preemptar processos do tipo I/O bound , assim os recursos computacionais seriam utilizados de forma mais eficientes

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 (início da execução até a finalização) dos processos. Ignore o overhead causado pela troca de contexto. Se não especifica o tempo de chegada, considera todos no tempo de chegada =0

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

Resposta*: P1=20, P2=18, P3=8, P4=10, P5=20*

(20+18+8+10+20+) / 5 = 15,2

b) Escalonamento com prioridade

Resposta*:*

Por prioridade interna: P2=0 - 6-6, P5=6 – 14-8, P1=14 – 24-10, P3=24 – 26-2, P4=26 – 30-4

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

Resposta*: P1=0, P2=10, P3=16, P4=18, P5=22*

(0+10+16+18+22) / 5 = 13,2

d) SJF organizado pela ordem do burstTime – Melhor desempenho!

Resposta*: P3=0, P4=2, P2=6, P5=12, P1=20*

(0+2+6+12+20) / 5 = 8

TempoMédioTurnAroun = 14

Considere um sistema operacional cuja máquina de estados inclui os estados Ready e Ready-Suspended (em disco). Suponha que seja hora do S.O. despachar (escalonar) um processo e que existam neste 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 que tente balancear prioridade e desempenho.

Resposta*:*

Uma possível política intermediária seria continuar executando os da fila ready até carregar o de maior prioridade da fila readySuspended para assim preemptar o processo da ready e executar o da fila readySuspended.

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), em que p é o índice da tabela de descritores (i.e. PCBs) de processos e que possui um campo (entre outros campos) 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.

Resposta*:*

O procedimento é chamado quando é interrompido sua execução pelo término do quantum do tempo compartilhado ou quando termina a execução de um processo. O procedimento insere (p) é chamado quando um novo processo é criado. O procedimento seleciona (p) é chamado quando solicitado uma determinada fila de processos

Algumas questões interessantes da internet:

Em que circunstâncias as decisões de escalonamento podem acontecer?

R*:* O escalonamento pode acontecer quando:

Um processo em execução passa para o estado de espera por I/O

Um processo em espera por I/O passa para o estado de pronto

Um processo em execução sofre uma interrupção para atender uma chamada de sistema e passa para o estado de pronto.

Um processo é encerrado

Diferencie os tempos do processador, espera, turnaround e resposta.

R*:*Tempo de processador é o tempo que um processo precisa para realizar sua tarefa

Tempo de espera é a soma do tempo que um processo fica na fila aguardando para iniciar sua execução

Turnround é o tempo gasto desde a entrada do processo em memória até sua conclusão

Tempo de resposta é o tempo gasto desde a entrada do processo em memória até o momento da primeira resposta.

Qual a diferença entre os escalonamentos FIFO e circular

R*:* O escalonamento FIFO procura terminar o processo primeiro antes de começar outro, enquanto o escalonamento circular reserva um determinado tempo para que o processo circule compartilhando o mesmo momento com outros processos. O Circular e do tipo Preemptivo, enquanto o FIFO e first in first out

Descreva o escalonamento SJF e o escalonamento por prioridades

R*:*As prioridades são baseadas nos processos que tiver o menor tempo de processador e desta forma sempre serão executados nesta ordem. Esta politica e preemptiva e está baseada na prioridade de execução.

Qual a diferença entre preempção por tempo e preempção por prioridade?

R*:* Preempção por tempo de vista interromper os processos baseados no menor tempo de processo, enquanto quando baseado na prioridade visa processar a fila através do nível de prioridade.

Descreva as ações tomadas pelo kernel para fazer a troca de contexto entre processos.

R:A troca de contexto exige que o estado do processo antigo (a sair do processamento) seja salvo e que o estado do processo novo (a entrar no processamento) seja carregado. O contexto é representado no PCB que inclui o valor dos registradores

Explique o que são os anéis de execução. Qual a diferença entre código executando no nível 0 e em outros níveis.

R:São extensões dos processadores que possibilitam separar os códigos sendo executados na CPU por camadas. Nos CHIPs Intel vão de RING0 (Kernel), passado por RING 1 (Drivers), RING 2 (Drivers) até RING3 (Aplicativos). O Kernel do Linux e do Windows XP usam somente o RING0 e RING3. O código executado em RING0 é o que tem mais privilégios (ou seja mais acesso ao hardware)

Qual a relação que há entre programa e processo?

R: Tanto o processo quanto o programa são uma entidade, um código fonte armazenado na memória, mas no caso do programa é uma entidade passiva, não necessariamente em execução. Já o processo se intitula como uma entidade ativa, em execução. Deixa de ser programa par ser processo.