INTRODUÇÃO AOS SISTEMAS OPERACIONAIS setembro/2013

SEMANA 09

Escalonadores.
Conceito, escalonamento, filas de escalonamento e tipos de escalonadores.

1 - Introdução

A entidade responsável pelo escalonamento é o escalonador [*scheduler*], que é quem determina qual processo deve sair ou ir para a CPU em determinado momento. É ele o elemento responsável pela alocação de processos a processadores, definindo a sua ordem de execução. A política de escalonamento [*scheduling*] é um problema complexo e depende do tipo de sistema suportado e da natureza das aplicações.

Em sistemas do tipo lote [*batch*], o escalonamento era feito simplesmente selecionando o próximo processo na fila de espera, já em sistemas multiusuário de tempo compartilhado geralmente combinados a sistemas em lote, o algoritmo de escalonamento deve ser mais complexo em virtude da existência de diversos usuários interativos solicitando serviços, e da execução de tarefas em segundo plano [*background*].

Um escalonador poderia funcionar da seguinte forma: quando a CPU torna-se disponível, o primeiro elemento da fila de processos prontos é retirado e inicia a sua execução. Caso seja bloqueado, este irá para o final da fila de processos bloqueados. Se a sua quota de execução se esgotar, este será retirado da CPU e colocado no final da lista de processos prontos.

Num sistema multitarefa, normalmente vários programas são executados concorrentemente e o número de processos é maior que o de CPUs. Um dos problemas mais difíceis na administração de recursos está relacionado ao fato de muitos processos existirem simultaneamente, ocorrendo em alguns casos uma disputa entre processos pelo uso do mesmo recurso.

2 - Escalonamento

O escalonamento de processos [scheduling] é uma atividade organizacional feita pelo escalonador, que possibilita executar processos concorrentemente. Para que a CPU não fique muito tempo sem executar tarefa alguma, existem técnicas para escalonar processos que não estão em execução.

Os responsáveis por essa tarefa são algoritmos de escalonamento. Existem vários algoritmos

de escalonamento e, para melhor escalonar os processos, o sistema operacional pode usar apenas um ou combinações de vários deles.

O escalonamento de processos é uma tarefa complicada, pois nenhum algoritmo é totalmente eficiente e a prova de falhas. A complexidade aumenta em se tratando de sistemas interativos, pois o usuário espera respostas rápidas e a todo o momento processos são tanto criados quanto interrompidos pelos usuários.

Tanenbaum propõem a seguinte definição: Quando mais de um processo está em execução, o sistema operacional deve decidir qual será executado primeiro. A parte do sistema operacional dedicada a esta decisão é chamada escalonador [scheduler], e o algoritmo utilizado é chamado algoritmo de escalonamento [scheduling algorithm].

Num sistema onde existe um único processador, o escalonamento representa a ordem em que os processos são executados. Já no caso de haver mais de um processador na máquina, o escalonamento de processadores é a forma como os processadores existentes num sistema computacional são utilizados para efetuar o processamento, isto é, como os processos são distribuídos para execução nos processadores.

O escalonador utiliza alguns critérios de escalonamento, como:

- taxa de utilização de CPU: é a fração de tempo durante a qual ela está sendo ocupada;
- throughput (vazão): são números de processos terminados por unidade de tempo;
- turnaround (retorno): é o tempo transcorrido desde o momento em que o processo inicia e o instante em que termina a sua execução;
- tempo de resposta: intervalo entre a chegada ao sistema e início de sua execução;
- tempo de espera: soma dos períodos em que o programa estava no seu estado pronto.

A forma com que se dá o escalonamento é, em grande parte, responsável pela produtividade e eficiência atingidas por um sistema computacional. Mais do que um simples mecanismo, o escalonamento deve representar uma política de tratamento dos processos que permita obter os melhores resultados possíveis.

O projeto de um escalonador leva em consideração uma série de necessidades, envolve uma política de escalonamento e contempla os seguintes objetivos:

- ser justo: todos os processos devem ser tratados igualmente, tendo possibilidades idênticas de uso do processador, devendo ser evitado o adiamento indefinido (*starvation* ou inanição);
- maximizar a produtividade [throughput]: procurar maximizar o número de tarefas processadas por unidade de tempo;
- ser previsível: uma tarefa deve ser sempre executada com aproximadamente o mesmo tempo e custo computacional;

- minimizar o tempo de resposta para usuários interativos;
- maximizar o número possível de usuários interativos;
- minimizar a sobrecarga [overhead]: recursos computacionais não devem ser desperdiçados;
- favorecer processos bem comportados: processos com comportamento adequado podem receber um serviço melhor;
- balancear o uso de recursos: o escalonador deve manter todos os recursos ocupados, ou seja, processos que usam recursos subutilizados devem ser favorecidos;
- exibir degradação previsível e progressiva em situações de intensa carga de trabalho.

É notório que alguns destes objetivos são contraditórios. A quantidade de tempo disponível para processamento [tempo do processador] e também os recursos computacionais são finitos, então só tem como favorecer um processo à custa do prejuízo de outro.

O maior problema no projeto de algoritmos de escalonamento está relacionado à natureza imprevisível dos processos, pois não tem como prever se um processo utilizará intensamente o processador, se precisará alocar uma grande quantidade de memória ou se necessitará efetuar um número muito grande de acessos a dispositivos de I/O.

3 - Filas de escalonamento

A partir do momento em que é criado um novo processo, este é colocado numa fila de processos para aguardar o seu momento de execução [rodar]. É importante notar que nos sistemas modernos, normalmente o processo recebe um tempo de processamento [slice ou fatia de tempo] menor que o necessário para completar o seu tempo de vida. Desse modo, enquanto durar o processo este permanece alternando entre momentos de uso do processador [rodando] ou momentos de permanência em alguma fila de espera.

Os processos que estão prontos para serem executados estão alocados numa fila de espera pelo processador. Não há filas para os estados novo, exit [finalizado] e rodando.

Quando o tempo de uso do processador termina, o processo é alocado na fila de processos prontos para aguardar ser selecionado de novo. Se o processo está bloqueado esperando por um evento, ele fica numa fila de espera até que o evento ocorra. Ocorrendo o evento, o processo é realocado para a fila de processos prontos [fila de espera do processador].

Existe também a fila para espera pelo uso de dispositivos de I/O. O processo é colocado numa fila de espera, onde é identificado o dispositivo requerido, e terá que aguardar caso um outro processo já esteja usando o dispositivo.

O esquema básico das filas de escalonamento de processos é ilustrado na figura 1. Nessa figura, cada caixa retangular representa uma fila com duas categorias: processos prontos [fila de

espera do processador] e de dispositivos [fila de espera do sispositivos de I/O]. Os círculos são os recursos e as elipses eventos. As setas representam os fluxos. Todo novo processo é alocado na fila de espera do processador onde aguarda ser selecionado para execução.

No diagrama de filas de espera, podem ocorrer os seguintes eventos:

- o processo emite uma requisição para uso de um dispositivo de I/O e é alocado na fila do mesmo;
- o processo pode criar um processo filho, e espera o seu término;
- o processo espera pela ocorrência de uma interrupção;
- o processo pode ser removido da CPU como resultado de uma interrupção e ser alocado na fila de processos prontos;

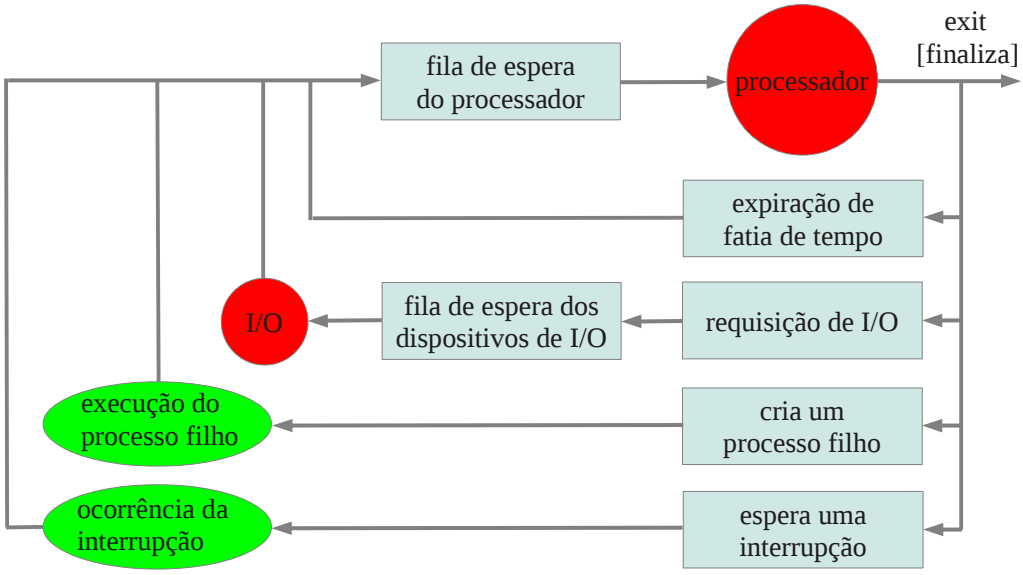

Fig 1 – Diagrama de filas de escalonamento

A figura 1 também mostra que um processo pode migrar entre as várias filas de escalonamento durante o seu tempo de vida. É tarefa do gerenciador de processos selecionar os processos nessas filas para usar os recursos computacionais. A parte do gerenciamento de processos que é responsável por essa seleção é o escalonador de processos, que seleciona com base em algum método ou algoritmo.

Preempção é a interrupção forçada de um processo para que outro processo possa usar o processador, e é usada em sistemas multiprogramados para garantir que todos os processos possam receber a CPU e progredir uniformemente.

A preempção é implementada através de um relógio de tempo (clock) que interrompe o processador em intervalos de tempo regulares, para que o escalonador possa fazer uma realocação de prioridades e, possivelmente, escalonar outro processo. Este intervalo é a unidade básica de alocação de tempo do processador, e é denominado quantum.

A preempção possui um custo, pois para escalonar um processo é necessário fazer troca de contexto, que gera overhead [é carga para o hardware e não contabiliza como andamento dos processos].

4 - Tipos de escalonadores

Quando se considera a frequência e a complexidade das operações envolvidas, existem três níveis distintos de escalonamento em um sistema computacional: **escalonador de curto prazo** [também conhecido por escalonamento de baixo nível], **escalonador de médio prazo** [ou escalonamento de nível intermediário] e **escalonador de longo prazo** [ou escalonamento de alto nível].

- escalonador de curto prazo: seleciona quais processos no estado pronto devem ser executados em seguida, e reserva o processador para executar esse processamento. O escalonador de curto prazo faz decisões de escalonamento com muito mais frequência que os de médio e de longo prazo. O prazo típico para o escalonador de curto prazo ser invocado é da ordem de milisegundos. É também conhecido como expedidor ou despachante [dispatcher]. Toma decisões de baixo nível e define o próximo processo a ir para o estado rodando. Representa a troca de contexto e o chaveamento do processador entre os processos ativos;

- escalonador de médio prazo: a ideia básica é swapping. Este escalonador temporariamente remove os processos da memória primária e os coloca na memória secundária [swap] fazendo as operações de *swapping in* e *swapping out*. Mais tarde, estes processos podem ser reintroduzidos na memória primária e continuar as suas execuções a partir do ponto onde haviam parado. O objetivo do *swapping* é liberar a memória primária para os outros processos. As decisões deste escalonador são baseadas na necessidade de gerenciar o grau de multiprogramação do sistema, e que leva em consideração os requisitos de memória dos processos que são removidos para o sistema de arquivo [swap]. Para este escalonador, o gerenciamento da memória é importante. O grau de multiprogramação está relacionado ao número de processos na memória primária, quanto mais processos couber, maior é a eficiência;

- escalonador de longo prazo: é invocado com pouca frequência, tipicamente em tempos da ordem de segundos a minutos. Seleciona os processos que utilizam por menos tempo o processador. Também determina quais tarefas serão admitidas para processamento no sistema, e que se tornarão processos. Toma decisões de alto nível.

Ainda sobre o escalonador de longo prazo, em sistemas do tipo batch normalmente existem mais processos submetidos do que podem ser executados pelo sistema. Estes processos são mantidos em um dispositivo de armazenamento de massa para serem executados posteriormente. O escalonador de longo prazo seleciona dentre estes processos quais podem ser carregados para a memória e inseridos na fila de processos prontos para executar.

O escalonador de longo prazo controla o grau de multiprogramação do sistema. Em sistemas de tempo compartilhado modernos como os sistemas da família Unix, não existe mais um escalonador de longo prazo, e todos os processos são colocados na memória primária ou swap.

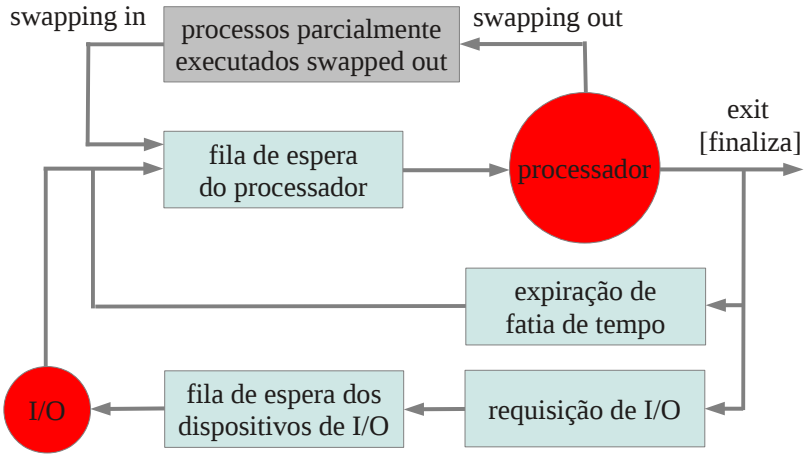

Fig 2 – Escalonamento de médio prazo

As decisões sobre o escalonamento do processo podem ser tomadas nos seguintes momentos:

i. quando um processo sai do estado "rodando" para "bloqueado";

ii. quando um processo sai do estado "rodando" para "pronto";

iii. quando um processo sai do estado "bloqueado" para "pronto";

iv. quando um processo é finalizado [exit].

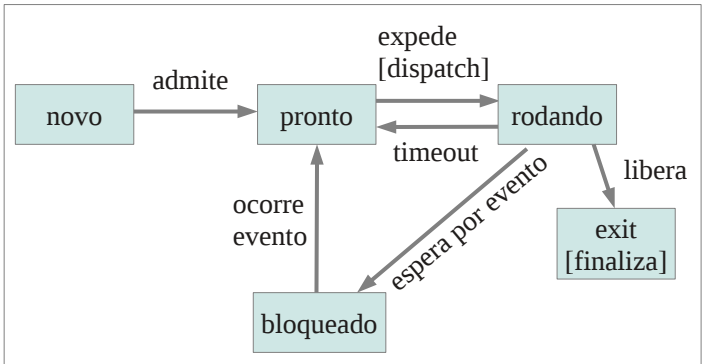

Fig 3 – Modelo de processo com 5 estados

Se um sistema operacional realiza escalonamento somente nas situações (i) e (iv), ele é dito não-preemptivo ou cooperativo. Caso ele também realize escalonamento nas outras situações, o sistema é dito preemptivo.

O escalonamento *não-preemptivo* ocorre quando o processo que está executando não pode ser interrompido. Esse tipo de escalonamento estava presente nos primeiros sistemas multiprogramáveis, onde predominava o processamento em *batch*. As políticas que implementam escalonamento não-preemptivo não são aplicáveis a sistemas de tempo compartilhado. Os primeiros sistemas de programas em lote [*batch*] inicialmente liam todas as intruções sequencialmente e depois as executavam uma após a outra.

Já no escalonamento *preemptivo*, o processo pode ser retirado do processador que está executando. Isso permite atenção imediata aos processos mais prioritários [tempo real], melhores tempos de resposta [tempo compartilhado] e compartilhamento uniforme do processador.

Um algoritmo de escalonamento é dito não-preemptivo quando o processador designado para o processo não pode ser retirado deste até que o processo seja finalizado. Analogamente, um algoritmo de escalonamento é considerado preemptivo quando o processador designado para um processo pode ser retirado deste em favor de outro processo.

Algoritmos preemptivos são mais adequados para sistemas em que múltiplos processos requerem atenção do sistema, ou seja, no caso de sistemas multiusuário interativos [sistemas de tempo compartilhado] ou em sistemas de tempo real. Nestes casos, a preemptividade representa a troca do processo em execução, onde o processador interrompe o seu trabalho, é retirado de um processo e designado para outro. Para essa troca de processo em execução é necessário a troca de contexto. Usualmente os algoritmos preemptivos são mais complexos dada a natureza imprevisível dos processos.

Por sua vez, os algoritmos não-preemptivos são mais simples e adequados para o processamento não interativo, como é o caso dos antigos sistemas batch. Os algoritmos não preemptivos são geralmente mais eficientes e previsíveis quanto ao tempo de entrega de suas tarefas.

Existe também a multitarefa *cooperativa* onde não existe a figura do escalonador, pois nesse caso são os aplicativos que cooperativamente se revezam no uso dos recursos de CPU e memória. Porém, nesse caso, se determinado aplicativo apresentar algum problema trava o sistema. Era a multitarefa cooperativa que dava aos aplicativos escritos para Windows 3.X a impressão de ser multitarefa, muito embora todos eles rodassem sobre o DOS, um sistema monotarefa. Era também esse o principal motivo das "travadas" frequentes.

No escalonamento cooperativo os processos não são interrompidos, mas a preempção ocorre em duas situações bem definidas: quando o processo efetua uma operação de I/O e quando o processo é finalizado. Também ocorre de um processo voluntariamente ceder o processador, em favor de outros processos. A preempção voluntária pode auxiliar numa distribuição mais equitativa da capacidade de processamento do sistema, porém depende demais da generosidade do programador que criou a aplicação.

Por fim, a preemptividade parte do fato de que o processador e a memória são naturalmente recursos preemptivos, ou seja, um processo pode tanto ser retirado do processador quanto da memória principal e posteriormente ser devolvido sem prejuízo. Mas recursos como arquivos e impressoras não podem sofrer preempção, e em muitos casos não podem ser retirados de um processo sem que ocorra prejuízo para este.