Universidade Federal de Itajubá – UNIFEI

Instituto de Engenharia de Sistemas e Tecnologias da Informação – IESTI
CCO 004 – Sistemas Operacionais – Prof. Edmilson Marmo Moreira

Capítulo 4 – Gerência do Processador

"O que sabemos é uma gota, o que ignoramos é um oceano."
Isaac Newton

4.1 Introdução

Com o surgimento dos sistemas multiprogramáveis, onde múltiplos processos poderiam permanecer na memória principal compartilhando o uso da CPU, a gerência do processador tornou-se uma das atividades mais importantes em um sistema operacional.

A partir do momento em que diversos processos podem estar no estado de pronto, devem ser estabelecidos critérios para determinar qual processo será escolhido para fazer uso do processador.

Os critérios utilizados para esta seleção compõem a chamada *política de escalonamento*, que é a base da gerência do processador e da multiprogramação em um sistema operacional.

Funções Básicas

A política de escalonamento de um sistema operacional possui diversas funções básicas, tais como:

- Manter o processador ocupado a maior parte do tempo;
- Balancear o uso da CPU entre os processos;
- Privilegiar a execução de aplicações críticas;
- Maximizar o *throughput* do sistema;
- Oferecer tempos de resposta razoáveis para usuários interativos.
- etc.

Cada SO possui sua política de escalonamento adequada ao seu propósito e às suas características.

A rotina do SO que tem como principal função implementar os critérios da política de escalonamento é denominada escalonador (*scheduler*).

Em um sistema multiprogramável, o escalonador é fundamental, pois todo o compartilhamento do processador é dependente dessa rotina.

Outra rotina importante na gerência do processador é conhecida como *dispatcher,* responsável pela troca de contexto dos processos após o escalonador determinar qual processo deve fazer uso do processador.

O período de tempo gasto na substituição de um processo em execução por outro é denominado latência do *dispatcher.*

Em ambientes que implementam apenas processos, o escalonamento é realizado com base nos processos prontos para execução. Em sistemas que implementam *threads,* o escalonamento é realizado considerando os *threads* no estado de pronto, independente do processo.

Critérios de Escalonamento

As características de cada sistema operacional determinam quais são os principais aspectos para a implementação de uma política de escalonamento adequada.

Por exemplo, sistemas de tempo compartilhado exigem que o escalonamento trate todos os processos de forma igual, evitando, assim a ocorrência de *starvation*, ou seja, que um processo fique indefinidamente esperando pela utilização do processador.

Já em sistemas de tempo real, o escalonamento deve priorizar a execução de processos críticos em detrimento da execução de outros processos.

Os principais critérios que devem ser considerados em uma política de escalonamento são:

- **Utilização do Processador:** Na maioria dos sistemas, é desejável que o processador permaneça ocupado a maior parte do tempo. Uma utilização na faixa de 30% indica um sistema com uma carga de processamento baixa, enquanto que na faixa de 90% indica um sistema bastante carregado.

- ***Throughput*:** Representa o número de processos executados em um determinado intervalo de tempo. Quanto maior o *throughput,* maior o número de tarefas executadas em função do tempo. A maximização do *throughput* é desejada na maioria dos sistemas.

- **Tempo de Processador / Tempo de CPU:** é o tempo que um processo leva no estado de execução durante o seu processamento. As políticas de escalonamento não influenciam o tempo de processador de um processo, sendo este tempo função apenas do código da aplicação e da entrada de dados.

- **Tempo de Espera:** é o tempo total que um processo permanece na fila de pronto durante seu processamento, aguardando para ser executado. A redução do tempo de espera dos processos é desejada pela maioria das políticas de escalonamento.

- **Tempo de *Turnaround*:** é o tempo que um processo leva desde a sua criação até ao seu término, levando em consideração todo o tempo gasto na espera para alocação de memória, espera na fila de pronto (tempo de espera), processamento na CPU (tempo de processador) e na fila de bloqueados, ou seja, nas operações de E/S. As políticas de escalonamento buscam minimizar o tempo de *turnaround*.

- **Tempo de Resposta:** é o tempo decorrido entre uma requisição ao sistema ou à aplicação e o instante em que a resposta é exibida. Em sistemas interativos, pode-se entender o tempo de resposta como o tempo decorrido entre a última tecla digitada pelo usuário e o início da exibição do resultado no monitor. Em geral, o tempo de resposta não é limitado pela capacidade de processamento do sistema computacional, mas pela velocidade dos dispositivos de E/S.

De uma maneira geral, qualquer política de escalonamento busca otimizar a utilização do processador e o *throughput,* enquanto tenta diminuir os tempos de *turnaround,* espera e resposta.

Entretanto, as funções que uma política de escalonamento deve possuir são muitas vezes conflitantes. Dependendo do tipo de sistema operacional, um critério pode ter maior importância do que outros, como nos sistemas interativos onde o tempo de resposta tem grande relevância.

4.2 Escalonamentos Não-Preemptivos e Preemptivos

As políticas de escalonamento podem ser classificadas segundo a possibilidade de o sistema operacional interromper um processo em execução e substituí-lo por um outro, atividade conhecida como *preempção*.

Escalonamento não-preemptivo

Foi o primeiro tipo de escalonamento implementado nos sistemas operacionais multiprogramáveis, onde predominava o processamento *batch.*

Nesse tipo de escalonamento, um processo que está em execução não pode perder o uso do processador devido a algum evento externo.

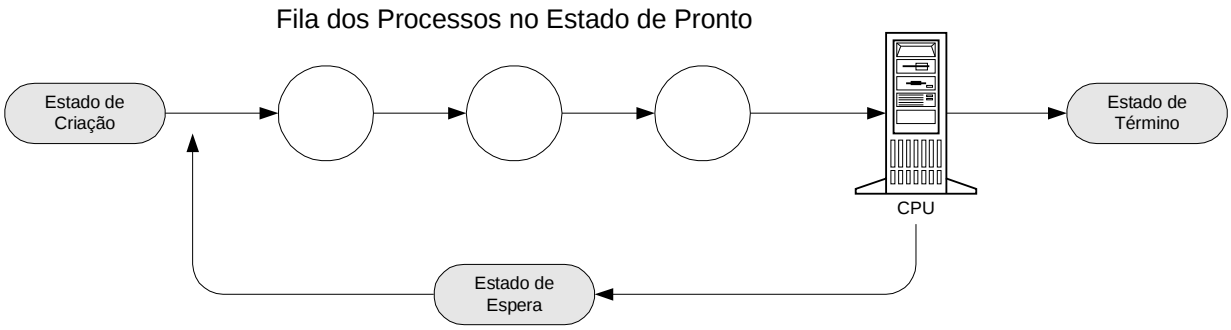
O processo somente sai do estado de execução caso termine seu processamento ou execute instruções do próprio código que ocasionem uma mudança para o estado de espera.

Os algoritmos não-preemptivos mais conhecidos são: FIFO (*First In First Out*), SJF (*Shortest Job First*) e HRN (*Highest Response Next*).

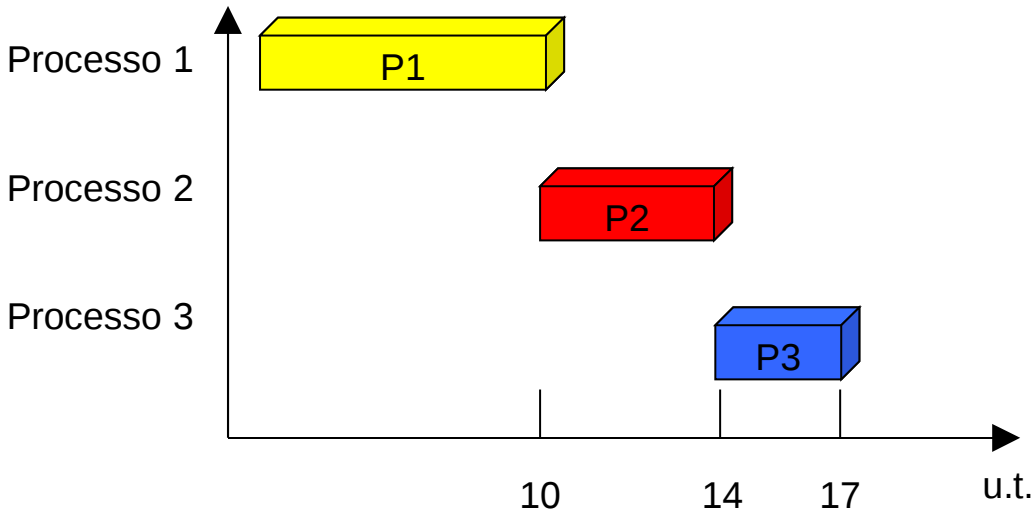
Escalonamento *First In First Out* (FIFO)

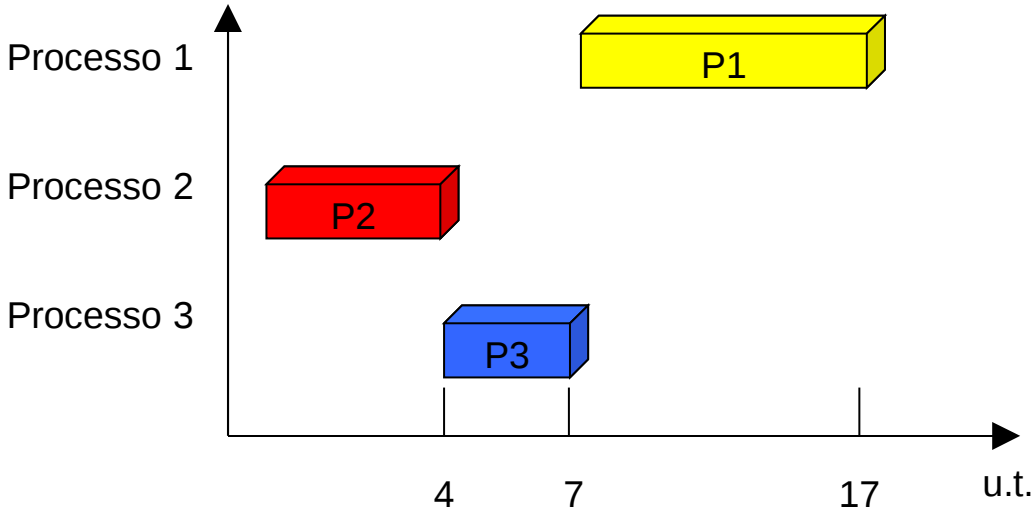
É também conhecido como FCFS (*First Come First Served*). O processo que chegar primeiro ao estado de pronto é o selecionado para execução.

Este algoritmo é bastante simples e utiliza apenas uma fila, onde os processos que passam para o estado de pronto entram no seu final e são escalonados quando chegam ao seu início.



Na figura abaixo é possível comparar o uso do escalonamento FIFO em duas situações distintas, onde os processos 1, 2 e 3 são criados no instante de tempo 0, com os tempos de processador 10, 4 e 3, respectivamente.





A diferença entre os exemplos é o posicionamento dos processos na fila de pronto. O tempo médio de espera dos três processos no primeiro exemplo é igual a (0 + 10 + 14) / 3 = 8 u.t., enquanto que no segundo exemplo é de aproximadamente 3,7 u.t.

A diferença entre as médias é bastante significativa, justificada pela diferença dos tempos de processador entre os processos.

Apesar da simplicidade o escalonamento FIFO apresenta algumas deficiências. O principal problema é a impossibilidade de se prever quando um processo terá sua execução iniciada, já que isso varia em função do tempo de execução dos demais processos.

Este algoritmo não se preocupa em melhorar o tempo médio de espera dos processos, utilizando apenas a ordem de chegada dos processos à fila de pronto.

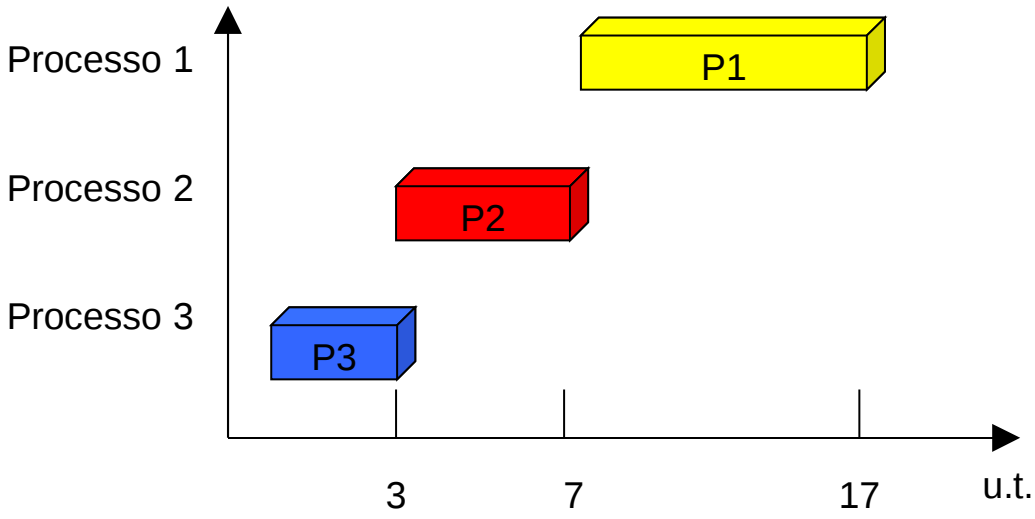
Este problema pode ser percebido nos tempos de *turnaround* dos processos que demandam menor tempo de CPU.

Outro problema nesse tipo de escalonamento é que processos CPU-*bound* levam vantagem no uso do processador sobre processos I/O-*bound.*

Escalonamento *Shortest Job First* (SJF)

É também conhecido como SPN (*Shortest Process Next*). É selecionado o processo que tiver o menor tempo estimado de execução para terminar o seu processamento.

O mesmo exemplo do escalonamento FIFO, ficaria conforme figura abaixo no escalonamento SJF.



O tempo médio de espera dos três processos, igual a (7 + 3 + 0) / 3 = 3,3 u.t., é inferior ao apresentado no escalonamento FIFO.

Essa implementação foi utilizada nos primeiros sistemas operacionais com processamento exclusivamente *batch.*

Para cada novo processo admitido no sistema, um tempo de processador era associado ao seu contexto de software.

Como não é possível precisar previamente o tempo de processador para cada processo, uma estimativa era realizada com base em análises estatísticas de execuções passadas dos programas.

O tempo estimado era informado ao sistema operacional. Caso o tempo informado fosse muito inferior ao tempo real, o sistema operacional poderia interromper a execução do processo.

O principal problema nesta implementação é a impossibilidade de estimar o tempo de processador para processos interativos, devido à entrada de dados ser uma ação imprevisível.

Outro problema com o algoritmo SJF é a possibilidade de provocar *starvation* em alguns processos do sistema, ou seja, processos com tempo estimados de execução grande tendem a ficar mais tempo na fila de pronto.

Na sua concepção inicial, o escalonamento SJF é um escalonamento não-preemptivo. Uma implementação do escalonamento SJF com preempção é conhecida como escalonamento *Shortest Remaining Time* (SRT *scheduling*). Nessa política, toda vez que um processo no estado de pronto tem um tempo de processador estimado menor do que o processo em execução, o sistema operacional realiza uma preempção, substituindo-o pelo novo processo.

Semelhante ao SJF, o sistema operacional deve ser o responsável por estimar os tempos de processador dos processos, mas o risco de *starvation* permanece.

Escalonamento *Highest Response Next* (HRN)

Uma variação do algoritmo SJF é a política, também não-preemptiva, HRN.

O objetivo do escalonamento *Highest Response Next* é evitar o starvation, ainda assim, priorizando os processos com tempo estimado de execução pequeno.

A principal característica desse mecanismo é selecionar os processos com base na seguinte fórmula de prioridade:

$$\text{Prioridade} = \frac{\text{Tempo de Execução + Tempo de Espera}}{\text{Tempo de Execução}}$$

Neste mecanismo, todos os processos iniciam com prioridade 1, e a prioridade vai aumentado com o passar do tempo, devido ao valor do tempo de espera ser utilizado na fórmula.

Escalonamento Cooperativo

O escalonamento cooperativo é uma implementação que busca aumentar o grau de multiprogramação em políticas de escalonamento que não possuam mecanismos de preempção, como o FIFO e o SJF.

Neste caso, um processo em execução pode voluntariamente liberar o processador retornando à fila de pronto, possibilitando que um novo processo seja escalonado e, assim, permitir melhor distribuição no uso do processador.

A principal característica do escalonamento cooperativo está no fato de a liberação do processador ser uma tarefa realizada exclusivamente pelo processo em execução, que de uma maneira cooperativa libera a CPU para um outro processo.

Neste mecanismo, o processo em execução verifica periodicamente uma fila de mensagens para determinar se existem outros processos na fila de pronto.

Como a interrupção do processo em execução não é responsabilidade do sistema operacional, algumas situações indesejadas podem ocorrer. No caso de um processo não verificar a fila de mensagens, os demais não terão chance de ser executados até a liberação da CPU pelo processo em execução.

Um exemplo deste tipo de escalonamento pode ser encontrado nos primeiros sistemas operacionais da família Windows, sendo conhecido como multitarefa cooperativa.

Escalonamento Preemptivo

O escalonamento preemptivo é caracterizado pela possibilidade do sistema operacional interromper um processo em execução e passá-lo para o estado de pronto, com o objetivo de alocar outro processo na CPU.

Com o uso da preempção, é possível ao sistema priorizar a execução de processos, como no caso de aplicações de tempo real onde o fator tempo é crítico.

Outro benefício é a possibilidade de implementar políticas de escalonamento que compartilhem o processador de uma maneira mais uniforme, distribuindo de forma balanceada o uso da CPU entre os processos.

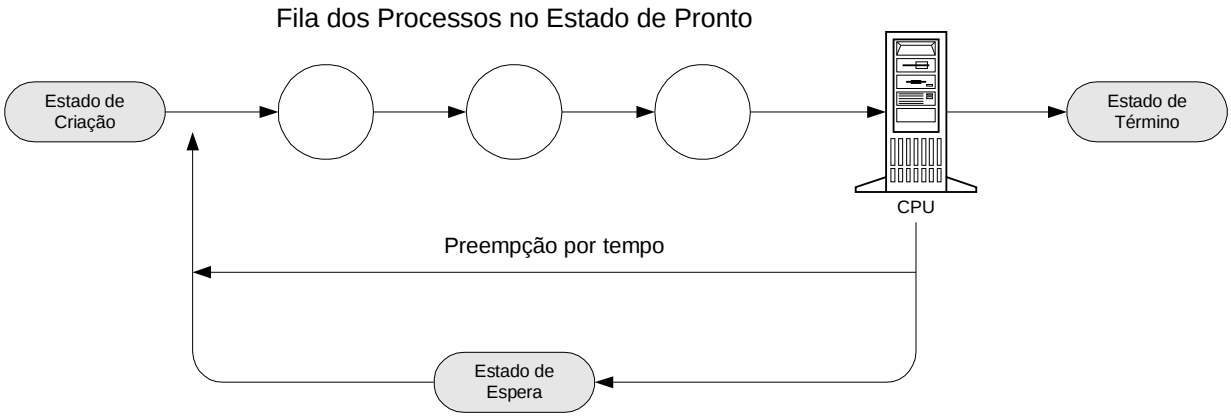
Escalonamento Circular

O escalonamento circular (*round robin scheduling*) é um escalonamento projetado especialmente para sistemas de tempo compartilhado.

Esse algoritmo é a versão preemptiva da política FIFO.

Quando um processo passa para o estado de execução, existe um limite para o uso contínuo do processador denominado fatia de tempo (*time-slice*) ou *quantum*.

No escalonamento circular, toda vez que um processo é escalonado para execução, uma nova fatia de tempo é concedida. Caso a fatia de tempo expire, o sistema operacional interrompe o processo em execução, salva seu contexto e direciona-o para o final da fila de pronto. Esse mecanismo é conhecido como preempção por tempo.

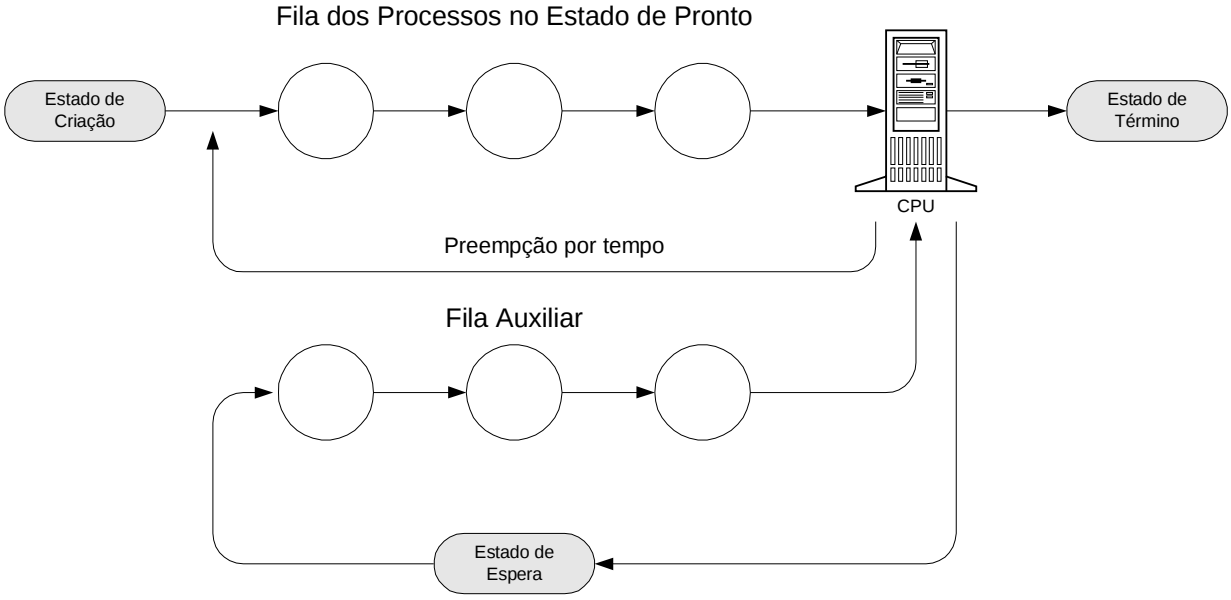


O valor da fatia de tempo depende da arquitetura de cada sistema operacional e, em geral, varia entre 10 e 100 milissegundos. Este valor afeta diretamente o desempenho da política de escalonamento circular. Caso a fatia de tempo tenha um valor muito alto, este escalonamento tenderá a ter o mesmo comportamento do escalonamento FIFO. Caso o valor do *time-slice* seja pequeno, a tendência é que haja um grande número de preempções, o que ocasionaria excessivas mudanças de contexto, prejudicando o desempenho do sistema e afetando o tempo de *turnaround* dos processos.

A principal vantagem do escalonamento circular é impedir que um processo monopolize a CPU.

Um problema presente nessa política, da mesma forma que na política FIFO, é que processos CPU-*bound* são beneficiados no uso do processador em relação aos processos I/O-*bound.*

Uma variação do escalonamento circular é conhecido como Escalonamento Circular Virtual.



Nesse esquema, processos que saem do estado de espera vão para uma fila de pronto auxiliar. Os processos da fila auxiliar possuem preferência no escalonamento em relação à fila de pronto, e o escalonador só seleciona processos na fila de pronto quando a fila auxiliar estiver vazia.

Quando um processo é escalonado a partir da fila auxiliar, sua fatia de tempo é calculada como sendo o valor da fatia de tempo do sistema menos o tempo de processador que o processo utilizou na última vez em que foi escalonado a partir da fila de pronto.

Estudos comprovam que, apesar de maior complexidade na implementação, o balanceamento do uso do processador neste esquema é mais equilibrado.

Escalonamento por Prioridades

O escalonamento por prioridades é um escalonamento do tipo preemptivo realizado com base em um valor associado a cada processo denominado prioridade de execução.

O processo com maior prioridade no estado de pronto é sempre o escolhido para execução, e processos com valores iguais são escalonados seguindo o critério FIFO.

Neste escalonamento, o conceito de fatia de tempo não existe; conseqüentemente, um processo em execução não pode sofrer preempção por tempo.

Assim, a perda do uso do processador só irá ocorrer no caso de mudança voluntária para o estado de espera ou quando um processo de prioridade maior passa para o estado de pronto.

Neste caso, o sistema operacional deverá interromper o processo corrente, salvar seu contexto e colocá-lo no estado de pronto. Esse mecanismo é conhecido como preempção por prioridade. Após isso, o processo de maior prioridade é escalonado.

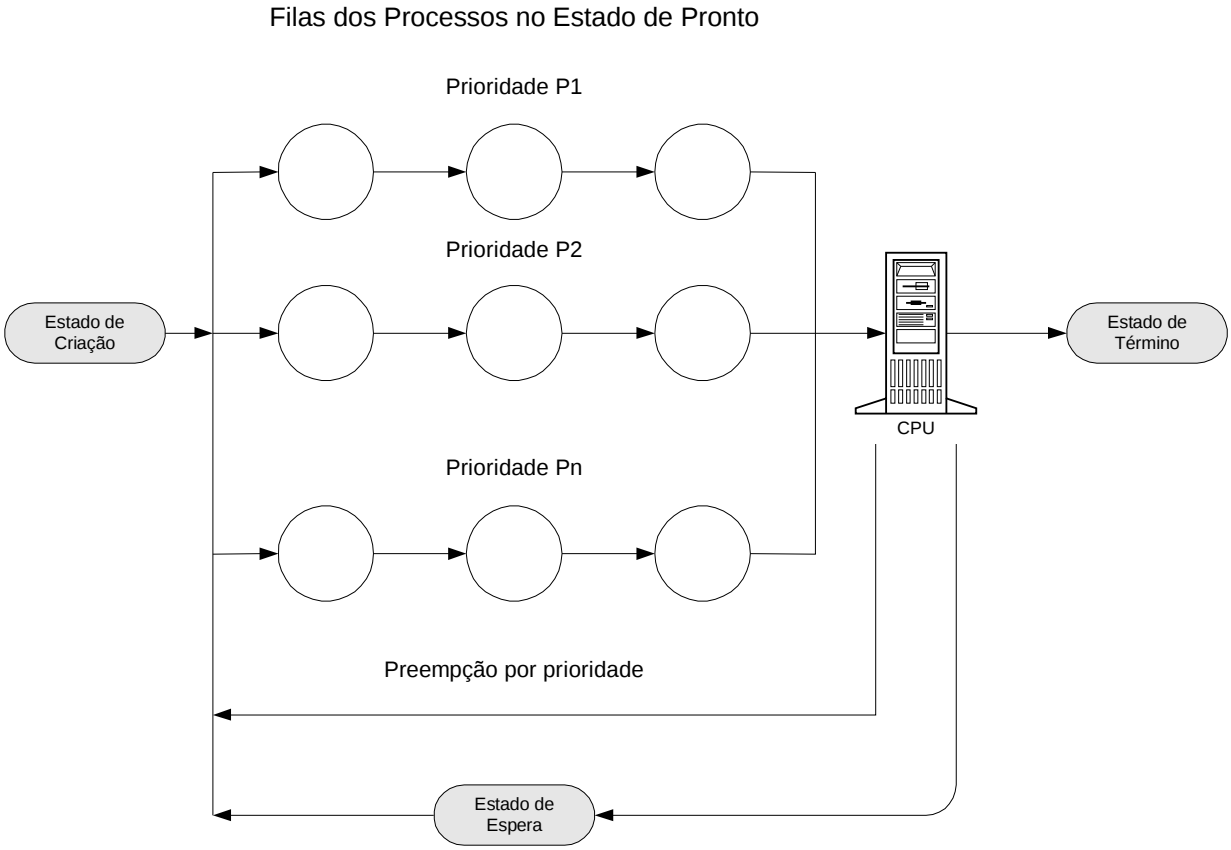
A preempção por prioridade é implementada através de um interrupção de relógio, gerada em determinados intervalos de tempo, para que a rotina de escalonamento reavalie as prioridades dos processos no estado de pronto.

Caso haja processos na fila de pronto com maior prioridade do que o processo em execução, o sistema operacional realiza a preempção.

A figura a seguir ilustra o escalonamento por prioridades, onde para cada prioridade existe uma fila de processos em estado de pronto que é

tratada como uma fila circular. O escalonamento é realizado alocando o processador ao primeiro processo da fila de prioridade mais alta.

O processo permanecerá no estado de execução até que termine seu processamento, voluntariamente passe para o estado de espera ou sofra uma preempção por prioridade.



O escalonamento por prioridades também pode ser implementado de uma maneira não-preemptiva. Neste caso, processos que passem para o estado de pronto com prioridade maior do que a do processo em execução não ocasionam preempção, sendo apenas colocados no início da filha de pronto.

Cada sistema operacional implementa sua faixa de valores para as prioridades de execução.

A prioridade de execução é uma característica do contexto de software de um processo e pode ser classificada como estática ou dinâmica.

A prioridade estática não tem seu valor alterado durante a existência do processo, já a prioridade dinâmica pode ser ajustada de acordo com critérios definidos pelo sistema operacional.

A possibilidade de alterar o valor da prioridade de um processo ao longo de seu processamento permite ajustar o critério de escalonamento em função do comportamento de cada processo no sistema.

Um dos principais problemas no escalonamento por prioridades é o *starvation.* Processos de baixa prioridade podem não ser escalonados, permanecendo indefinidamente na fila de pronto.

Uma solução para o *starvation*, em ambientes que implementam a prioridade dinâmica, é a técnica de *aging.* Este mecanismo incrementa gradualmente a prioridade de processos que permanecem por muito tempo na fila de pronto. Esta técnica é utilizada no política HRN.

Escalonamento por Múltiplas Filas

Esta política de escalonamento também é conhecida como Filas Multiníveis (*Multilevel Queues Scheduling*).

Existem diversas filas de processos no estado de pronto, cada qual com uma prioridade específica.

Os processos são associados às filas em função de características próprias, como importância para a aplicação, tipo de processamento ou área de memória necessária.

Como os processos possuem características de processamento distintas, é difícil que um único mecanismo de escalonamento seja adequado a todos.

A principal vantagem de múltiplas filas é a possibilidade da convivência de mecanismos de escalonamento distintos em um mesmo sistema operacional.

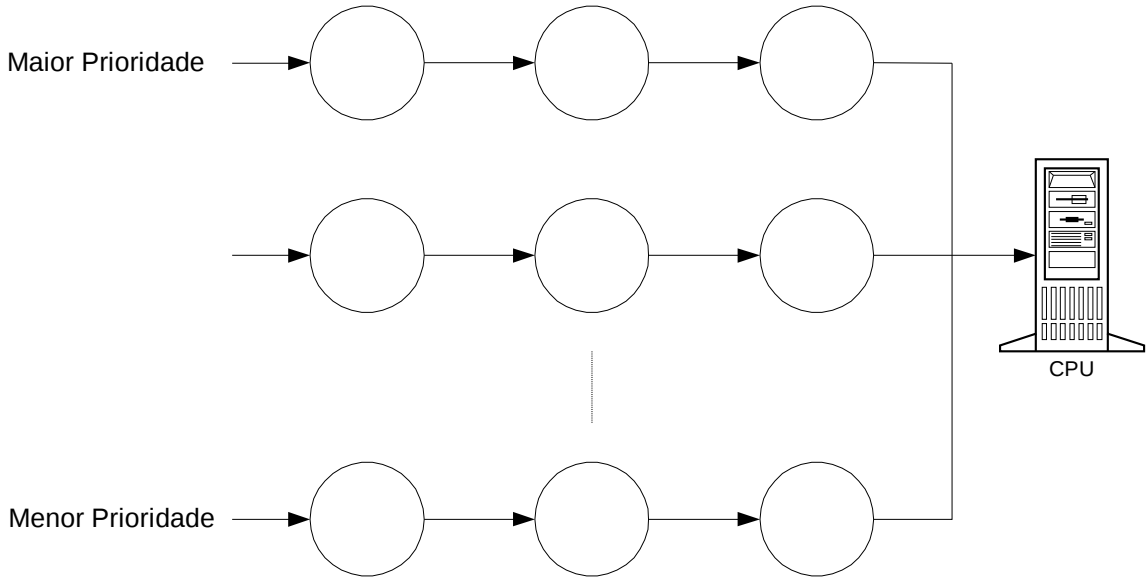
Cada fila possui um mecanismo próprio, permitindo que alguns processos sejam escalonados pelo mecanismo FIFO, enquanto outros pelo circular.

Neste mecanismo, o processo não possui prioridade, ficando esta característica associada à fila.

O processo sofre preempção caso um outro processo entre em uma fila de maior prioridade. O sistema operacional só pode escalonar processos

de uma determinada fila caso todas as outras filas de maior prioridade estejam vazias.

Uma boa prática é classificar os processos em função do tipo de processamento realizado e associá-los adequadamente às respectivas filas.



Uma desvantagem deste escalonamento é que, no caso de um processo alterar seu comportamento durante o seu processamento, ele não poderá ser redirecionado para uma outra fila mais adequada.

Escalonamento por Múltiplas Filas com Realimentação

Esta política de escalonamento também é conhecida como Filas Multiníveis com Realimentação (*Multilevel Feedback Queues Scheduling*).

É semelhante ao escalonamento por múltiplas filas, porém os processos podem trocar de fila durante o seu processamento.

Sua grande vantagem é permitir ao sistema operacional identificar dinamicamente o comportamento de cada processo.

Esse esquema permite que os processos sejam redirecionados entre as diversas filas, fazendo com que o sistema operacional implemente um mecanismo de ajuste dinâmico denominado mecanismo adaptativo.

Um mecanismo FIFO adaptado com fatia de tempo é implementado para escalonamento em todas filas, com exceção da fila de menor prioridade que, geralmente, utiliza o escalonamento circular. O escalonamento de um processo em fila ocorre apenas quando todas as outras filas de prioridade mais altas estiverem vazias.

A fatia de tempo em cada fila varia em função da sua prioridade, ou seja, quanto maior a prioridade da fila, maior sua fatia de tempo.

Um processo, quando criado, entra no final da fila de maior prioridade, porém durante sua execução, a cada preempção por tempo, o processo é redirecionado para uma fila de menor prioridade.

Este escalonamento atende às necessidades dos diversos tipos de processos. No caso de processos I/O-*bound,* um tempo de resposta adequado é obtido, já que esses processos têm prioridade mais altas por permanecerem a maior parte do tempo nas filas de maior prioridade, por dificilmente sofrerão preempção por tempo.

Por outro lado, em processos CPU-*bound* a tendência é de que, ao entrar na fila de mais alta prioridade, o processo ganhe o processador, gaste sua fatia de tempo, e seja direcionado para uma fila de menor prioridade.

Dessa forma, quanto mais tempo um processo se utiliza do processador, mais ele vai caindo para filas de menor prioridade.

O escalonamento por múltiplas filas com realimentação é um algoritmo de escalonamento generalista, podendo ser utilizado em qualquer tipo de sistema operacional.

Um dos problemas encontrados nessa política é que a mudança de comportamento de um processo CPU-*bound* para um I/O-*bound* pode comprometer seu tempo de resposta.

Outro aspecto a ser considerado é sua complexidade de implementação, ocasionando um grande *overhead* ao sistema.