

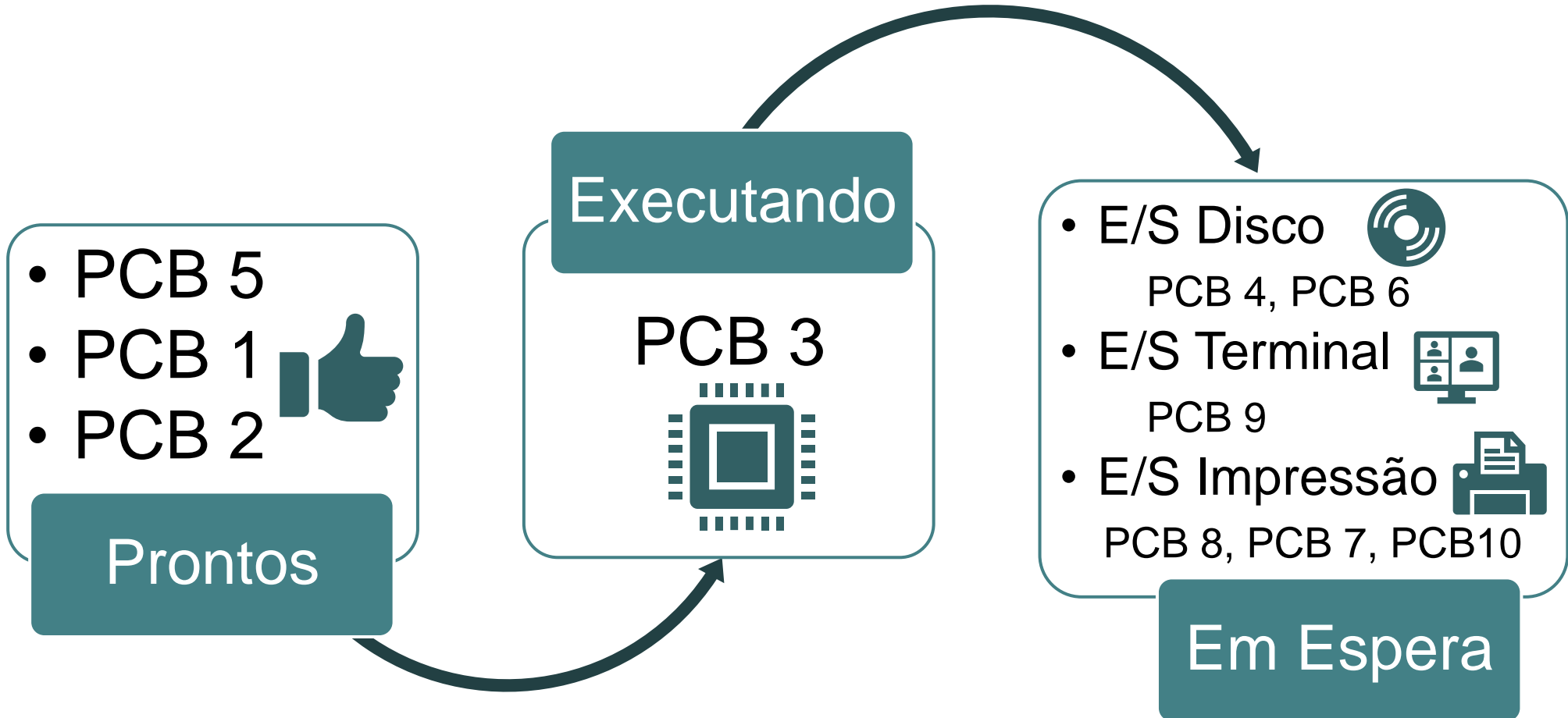
ESCALONAMENTO DE PROCESSOS

A decorative graphic consisting of several horizontal lines of varying lengths and colors (teal, light blue, and white) extending from the left side of the slide towards the right, positioned below the title.

- Embora os SOs sejam multitarefa, processando várias tarefas por vez, o processador só dá conta de um processo por vez, sendo que o mais comum é que seja feito o compartilhamento de tempo do processador para que ele atenda várias tarefas, se dedicando um tempo X a cada processo de cada tarefa.
- Isso dá ao usuário a impressão de dedicação exclusiva, mas não é o que ocorre.
- Executar as tarefas nos computadores sempre no menor tempo possível foi sempre uma prioridade nos estudos de computação.

- O escalonamento dos processos, falando nisso de maneira, simples, é a organização, a priorização das atividades feita pelo **scheduler (escalonador)** do processador, que permite que os processos mais importantes sejam executados primeiro.
- O escalonamento é a base dos sistemas operacionais multitarefa.
- Permite a máxima utilização da CPU.

Para cada estado, existe uma fila que contém os PCB's



Nas transições entre estados, o PCB do processo é movido entre as filas apropriadas

- O escalonamento seleciona um processo, entre os presentes na memória que estão prontos para execução, e o aloca à CPU.
- Consiste em determinar, dentre os processos prontos, qual o próximo processo a ser executado
- **Funções básicas do escalonamento:**
 - Manter o processador ocupado a maior parte do tempo.
 - Balancear o uso de CPU entre processos.
 - Privilegiar a execução de aplicações críticas.
 - Maximizar o throughput (capacidade de processamento) do sistema.

- A principal função do SO é implementar os critérios da política de escalonamento.
- Em um sistema multitarefa o escalonador é fundamental, pois sem ele não é possível o compartilhamento do processador. É por meio dele que o SO consegue gerenciar os processos que fazem uso do processador.

Escalonador vs. despachante

- Distingue-se duas partes:
 - O **escalonador** é responsável pela escolha do processo “eleito” para usar a CPU.
 - O **despachante ou despachador** é responsável pelo lado técnico de:
 - Salvar o processo que estava usando a CPU;
 - Executar o processo eleito na CPU.
 - Isso se chama efetuar a troca de contexto.

Escalonador vs. despachante

Processo A



.....



Processo B

- Existem três tipos de escalonadores:
 - **Escalonador longo prazo**
 - memória secundária → memória principal
 - **Escalonador de médio prazo**
 - Processos que serão removidos de forma parcial ou total da memória para serem suspensos
 - **Escalonador curto prazo**
 - memória principal → processador
- A cadeia de processos do escalonamento faz com que todos os processos consigam passar sobre a execução a partir de uma sequência de prioridade, se tratando de quem pode esperar mais, sendo longo prazo, e quem pode esperar menos, sendo o curto prazo.

Como funciona essa cadeia de ações?



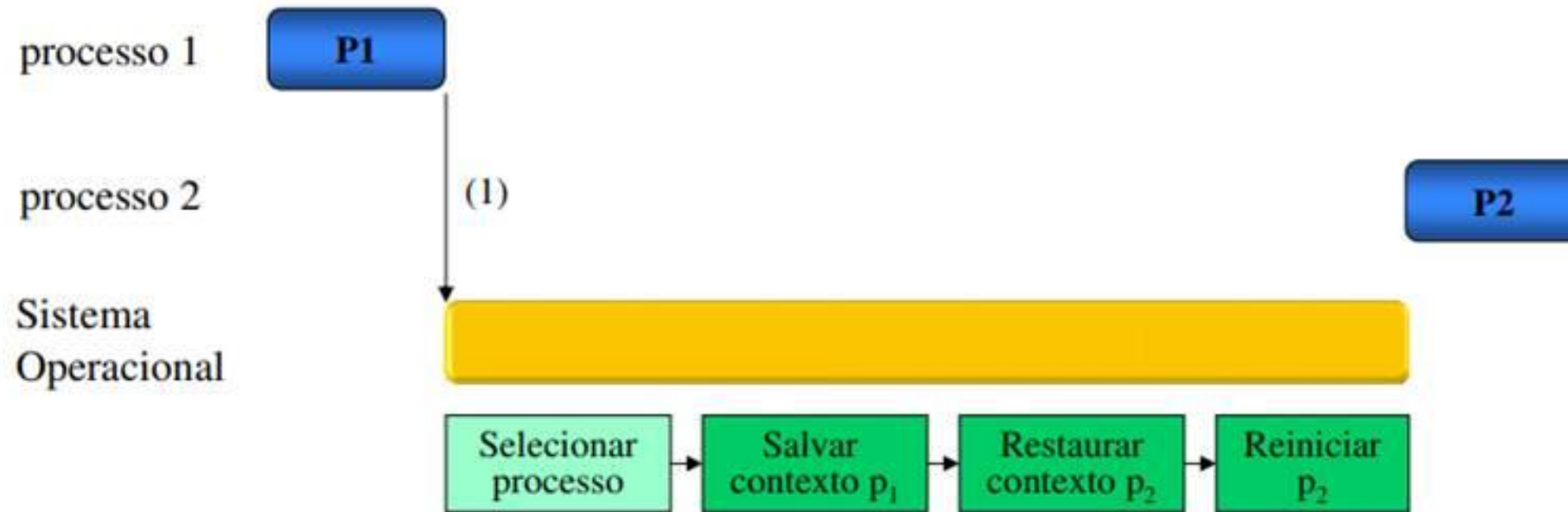
Como funciona a troca de processos?

- A finalização dentro do processo de escalonamento só é dada quando o processo adquire o recurso que necessita do sistema e então ocorre a troca de processos.
- Os recursos necessários são as informações vindo da CPU, sendo processos como **CPU bound**, **I/O bound**, sendo que os processos CPU Bound, que gastam todo o seu tempo em processo de CPU, ocorrem sequencialmente e de maneira rápida dentro do processador, pois estão prontas dentro do sistema esperando apenas o **SPINLOCK** para poder realizar o swap de tarefas.

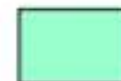
Como funciona a troca de processos?

- O tempo de **SPINLOCK** é um adiamento indefinido até o que o bloqueio da thread seja quebrado, podendo assim iniciar um novo processo.
- Quando uma thread tem em espera execuções de mesma prioridade ela provavelmente irá desperdiçar o **tempo de quantum (ou time-slice: período de tempo durante o qual um processo ou thread usa o processador a cada vez)**, que é o tempo de alocação em que a thread pode ser executada, esperando até que a thread seja liberado de seu bloqueio.

Como funciona a troca de processos?



(1) Finalização do tempo de execução ou o processo se bloqueia à espera de um recurso que necessita



Escalonador

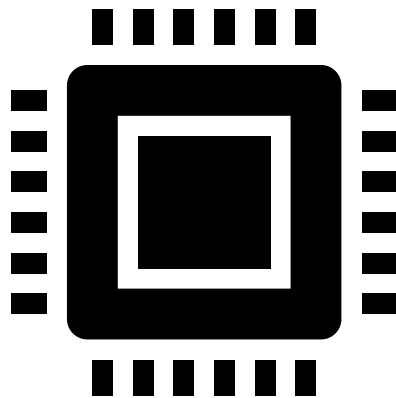


Despachador

Critérios de Escalonamento

- De acordo com as características do SO, determinam-se quais são os atributos para a implementação de um escalonamento adequado.
- **Esses critérios são: uso do processador, throughput, tempo de espera, tempo de turnaround e tempo de resposta**

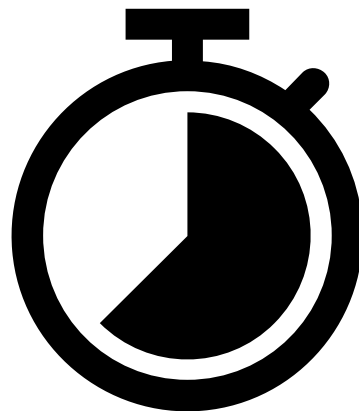
- **Utilização do processador:** é desejável que o processador permaneça a maior parte do seu tempo ocupado.
- Uma utilização na faixa de 30% indica um sistema com uma carga de processamento baixa, enquanto na faixa de 90% indica um sistema bastante carregado, próximo da capacidade máxima



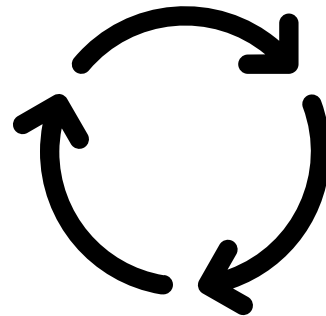
- **Throughput:** representa o número de processos executados em um determinado intervalo de tempo.
- Quanto maior o throughput, maior o número de tarefas executadas em função do tempo. A maximização do throughput é desejada na maioria dos sistemas operacionais



- **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.



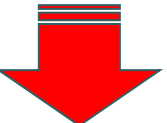
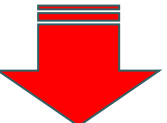



- **Tempo de turnaround:** é o tempo que um processo leva desde a sua criação até seu término, levando em consideração todo o tempo gasto na espera para alocação de memória, espera na fila de pronto, processamento de CPU e na fila de espera, como nas operações de E/S.
- As políticas de escalonamento 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 como o tempo decorrido entre a última tecla digitada pelo usuário e o início da execução do resultado no monitor.



- Uso do processador máximo 
- Throughput máximo 
- Tempo de espera mínimo 
- Tempo de turnaround mínimo 
- Tempo de resposta mínimo 

Escalonamentos Não Preemptivos e Preemptivos

- Podemos classificar os escalonamentos de acordo com a possibilidade ou não da interrupção de um processo.
- Se o sistema operacional consegue interromper um processo em execução e substituí-lo por outro, trata-se de um **escalonamento preemptivo**. Esses sistemas são mais complexos, porém possibilitam políticas de escalonamentos mais flexíveis.

- A vantagem do **escalonamento preemptivo** é que o sistema operacional pode interromper um processo em execução e passá-lo para o estado de pronto, no intuito de alocar outro processo na CPU.
- Assim, é possível priorizar a execução de processos, como nas 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 maneira mais uniforme, distribuindo de forma balanceada o uso da CPU entre os processos, aumentando a produtividade e diminuindo o tempo de turnaround.

- Já no **escalonamento não preemptivo**, quando um processo está em execução, nenhum evento externo irá ocasionar a perda do uso do processador, isso apenas acontecerá se o processo concluir seu processamento ou executar instruções do próprio código que promovam uma mudança de estado de espera

Escalonamento First-Come-First-Served (FCFS)

- O primeiro que chega é o primeiro a ser servido. Conhecido também como **FIFO (First-In-First-Out: o primeiro a entrar é o primeiro a sair)**.
- Nesse tipo de algoritmo de escalonamento, o processo que chegar primeiro ao estado de pronto é escolhido para execução.
- É necessária apenas uma fila onde os processos que passam para o estado de pronto entram no final e são escolhidos quando chegam ao seu início.

- Se um processo vai para o estado de espera, o primeiro que estiver na fila de pronto é escalonado.
- O grande problema é prever quando um processo terá sua execução iniciada, pois isso depende dos que estão na frente da fila.
- Fácil implementação e gerenciamento.
- **Esse escalonamento é do tipo não preemptivo e não pode ser usado em sistemas de tempo compartilhado.**

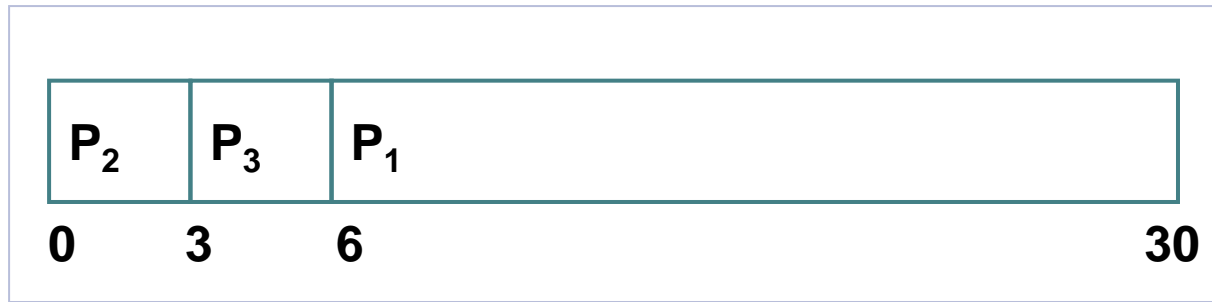
- Processo

P_1
 P_2
 P_3

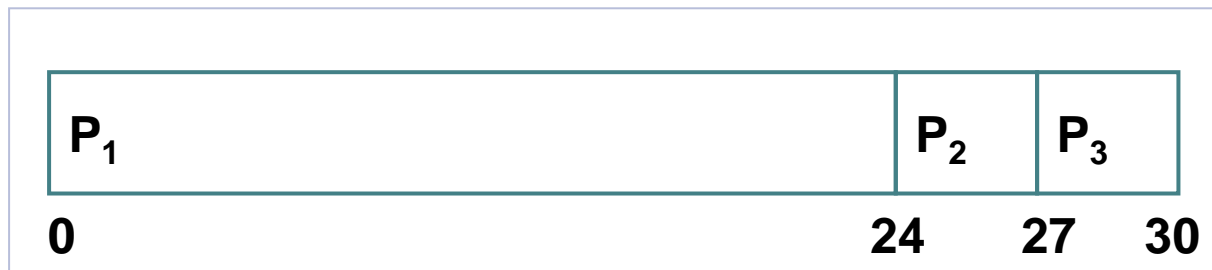
- Duração dos Processos

24 ms
3 ms
3 ms

- Ordem de Chegada: P_2, P_3, P_1



- Ordem de Chegada: P_1, P_2, P_3



- Tempo de Espera

$$P_1 = 6 \quad P_2 = 0 \quad P_3 = 3;$$

- Tempo de Espera Médio

$$(6 + 0 + 3) / 3 = \mathbf{3 \text{ ms}}$$

- Tempo de Espera

$$P_1 = 0 \quad P_2 = 24 \quad P_3 = 27$$

- Tempo de Espera Médio

$$(0 + 24 + 27) / 3 = \mathbf{17 \text{ ms}}$$

Escalonamento Shortest Job First (SJF)

- Processo menor primeiro.
- Esse algoritmo de escalonamento atende primeiramente os processos menores, que usam menos o processador.
- **Esse tipo de escalonamento pode ser preemptivo e não preemptivo.**

Processo

Duração (ms)

P_1 6 ms

P_2 8 ms

P_3 7 ms

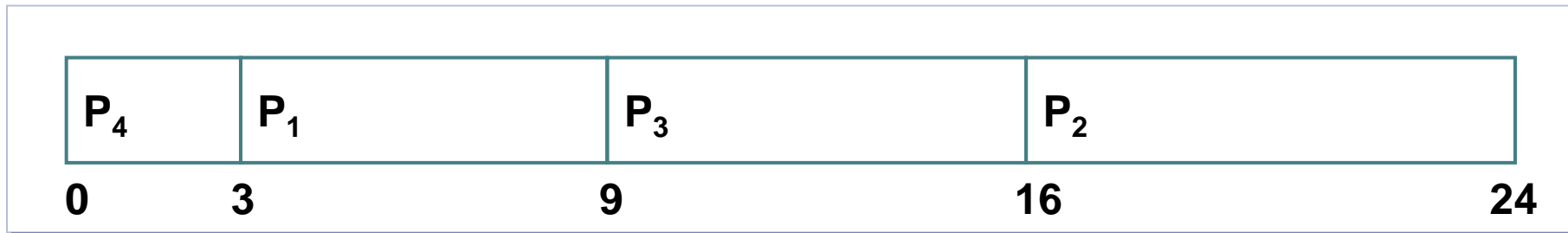
P_4 3 ms

Tempo de Espera

$$P_1 = 3 \quad P_2 = 16 \quad P_3 = 9 \quad P_4 = 0$$

Tempo de Espera Médio

$$(3 + 16 + 9 + 0) / 4 = 7 \text{ ms}$$



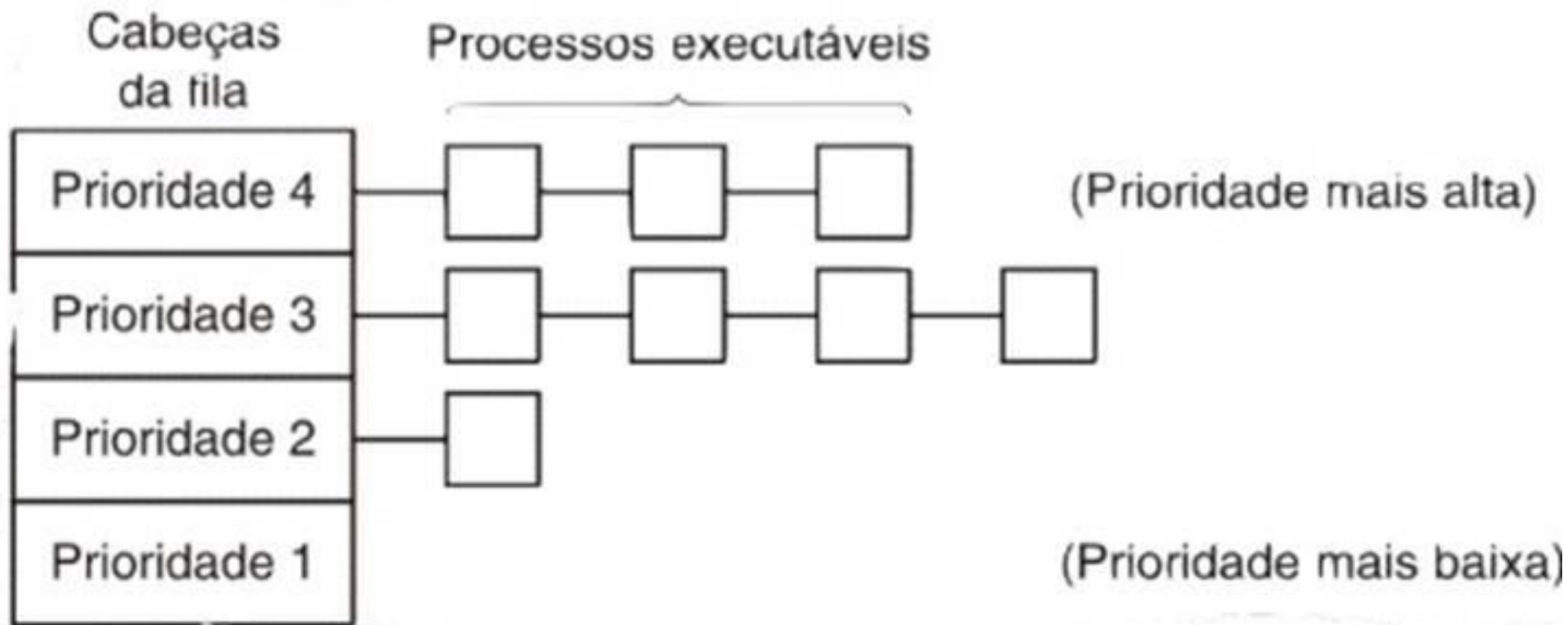
- Escalonamento **SJF não Preemptivo**: o processo faz uso do processador até o final da sua execução sem interrupção. Ao concluir o processamento, verifica-se na fila de pronto qual é o próximo “menor processo” a ser executado.
- Escalonamento **SJF Preemptivo**: caso chegue um processo menor que o processo que está sendo executado, o “maior” (que utiliza mais CPU) será interrompido, para atender o “menor” que chegou.

Escalonamento com prioridades

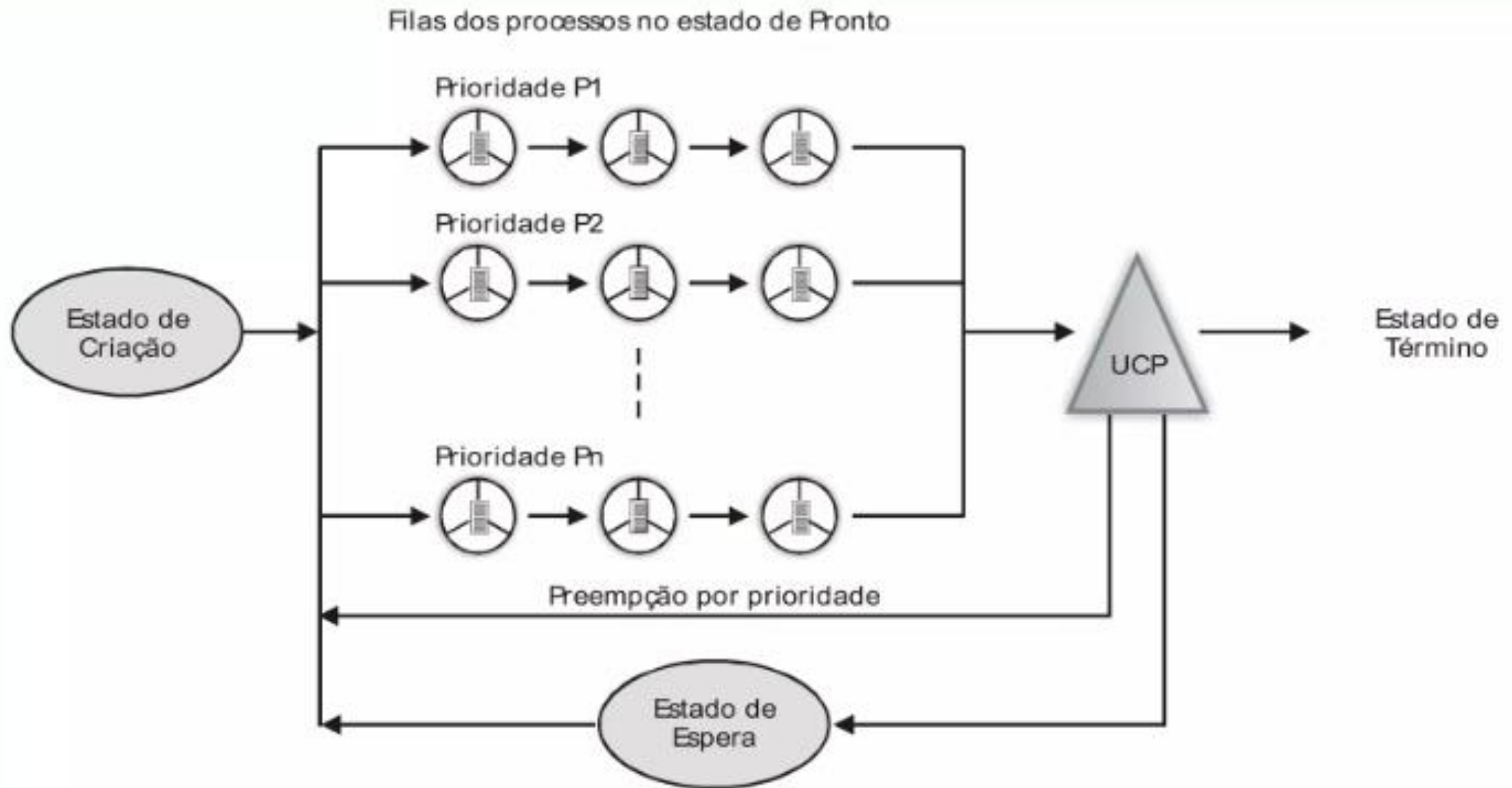
- A prioridade, quando estamos abordando escalonamento serve para os processos receberem um tratamento próprio. Logo após a criação de um processo, o mesmo recebe um grau de prioridade, e no momento onde o escalonador precisar escolher um processo pra ser executado, apontará para o que tiver maior prioridade.
- Para cada vez que este processo é executado, o escalonador decrementa a sua prioridade, até que seu grau de prioridade se torne inferior a outro processo, quando isso ocorre, ele é interrompido e o próximo processo será executado.
- Existe duas maneiras de ligar uma prioridade a um processo:

- **Prioridade estática:** Possui simples implementação, é associado no momento da criação do processo, e não é alterada durante sua existência. O seu principal problema é que pode gerar tempos de respostas elevados.
- **Prioridade dinâmica:** A prioridade pode ser modificada pelo escalonador, durante a execução do processo, e o grau de sua prioridade é dado de acordo com estatísticas geradas da execução deste processo anteriormente.
- **Os processos I/O-bounds (ligado à E/S – Entrada e Saída) terão sua prioridade aumentada, para compensar o tempo no estado de espera. São realizados sem o uso da CPU.**
- **Os processos CPU-bounds (ligado à CPU) podem ser executados enquanto os mesmos aguardam a realização de alguma ação.**

- Toda programação tem de ser pensada para executar corretamente as tarefas, para evitar a condição de corrida dentro das execuções.
- **Condição de corrida (race condition)** ocorre quando duas ou mais operações dependem da sequência ou do tempo de execução umas das outras, podendo gerar resultados imprevisíveis.
- Em sistemas operacionais, a ordem de execução de tarefas pode variar, e isso pode causar problemas, como inconsistências de dados ou comportamentos inesperados, caso as operações não sejam corretamente sincronizadas.



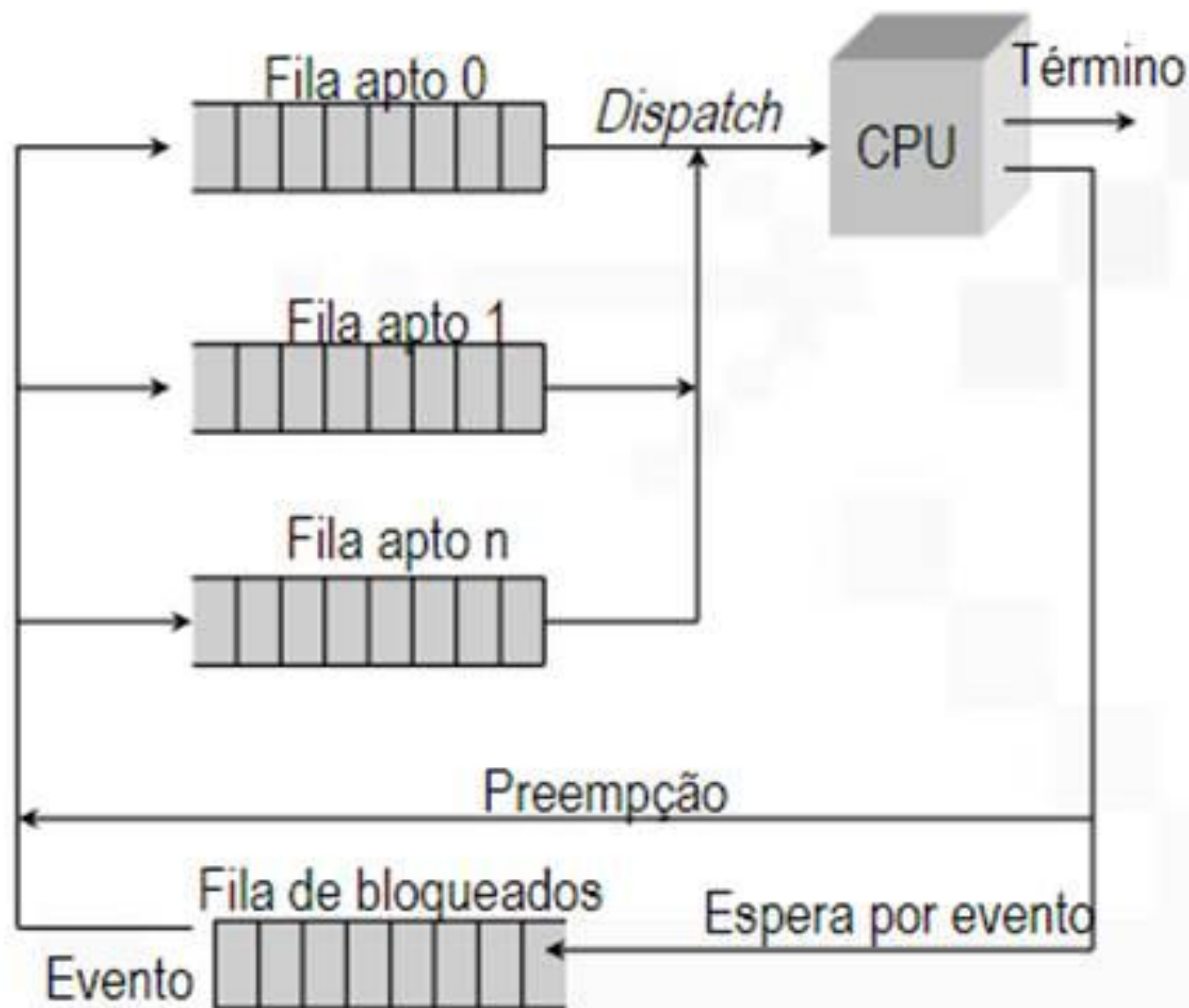
- Dentro do Escalonamento com Prioridade, encontramos o Escalonamento Circular com Prioridade, que implementa o conceito de fatia de tempo e de grau de prioridade de execução ligados em cada processo.
- Neste caso, um processo só entrará em estado de espera, ou sofrerá uma preempção por tempo ou prioridade (causar mudança de uma aplicação para outra).
- A sua vantagem está atrelada a otimização do balanceamento no uso do processador em sistemas tempo compartilhado.



Problemas com o escalonamento com prioridades

- Há como problema os processos que possuem baixa prioridade, pois podem não ser executados, criando assim um adiamento indefinido (**starvation**).
- Os processos com prioridade estática podem estar mal analisados, e assim sendo favorecidos ou penalizados em relação aos outros processos. Bem comum em processos que durante sua execução mudam de padrão de comportamento (I/O bound para CPU bound e vice versa).
- A solução para estes, seria o uso de múltiplas filas com realimentação, que é baseado em prioridade dinâmicas, o sistema de envelhecimento evita o adiamento indefinido.

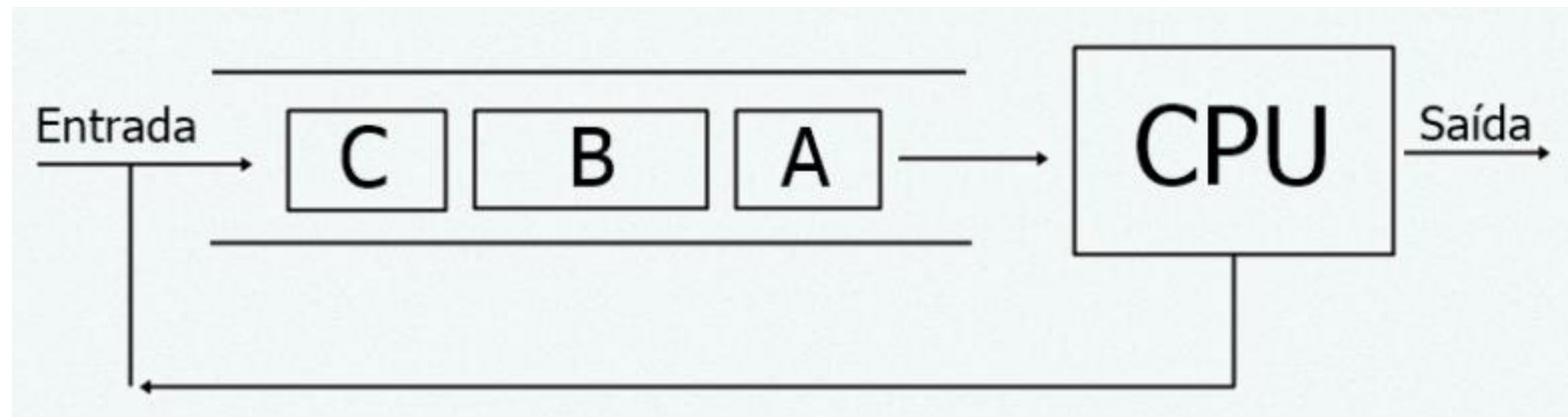
Possibilidade de
trocar de fila



Escalonamento circular (Round-Robin)

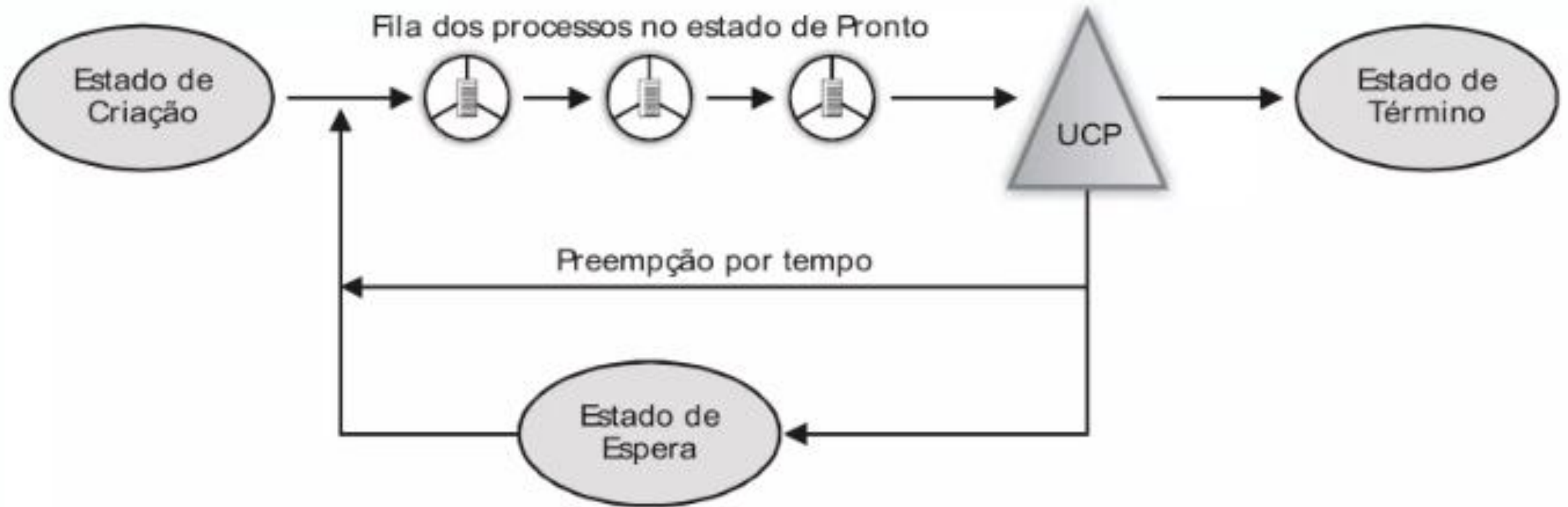
- É o tipo de escalonamento preemptivo mais simples e consiste em repartir uniformemente o tempo da CPU entre todos os processos prontos para a execução.
- Os processos são organizados numa fila circular, alocando-se a cada um uma fatia de tempo da CPU.
- Caso um processo não termine dentro de sua fatia de tempo, ele é colocado no fim da fila e uma nova fatia de tempo é alocada para o processo no começo da fila.

Escalonamento circular (Round-Robin)



- O escalonamento circular é muito simples, mas pode trazer problemas se os tempos de execução são muito discrepantes entre si.
- Quando existirem muitas tarefas ativas e de longa duração no sistema, as tarefas curtas terão o seu tempo de resposta degradado, pois as tarefas longas reciclarão continuamente na fila circular, compartilhando de maneira igual a CPU com tarefas curtas.

Escalonamento circular (Round-Robin)



Múltiplas Filas

- Prioridade preemptiva.
- Nesta abordagem, os processos são divididos em várias filas de acordo com suas características ou prioridades. Cada fila pode ter seu próprio algoritmo de escalonamento. Por exemplo, pode haver uma fila para processos de tempo real, uma fila para processos interativos e uma fila para processos em lote.
- Cada fila pode ter um escalonador diferente que decide qual processo deve ser executado a seguir, com base em critérios específicos daquela fila.

Múltiplas Filas

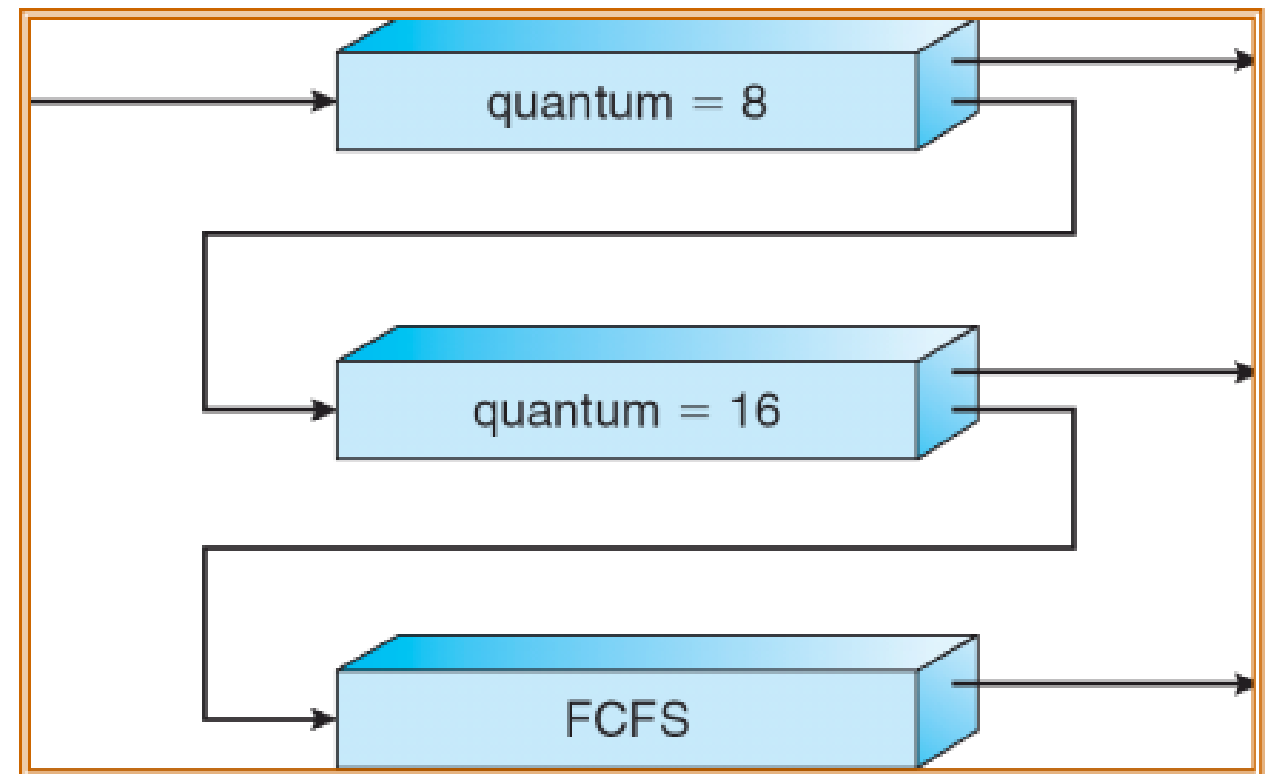
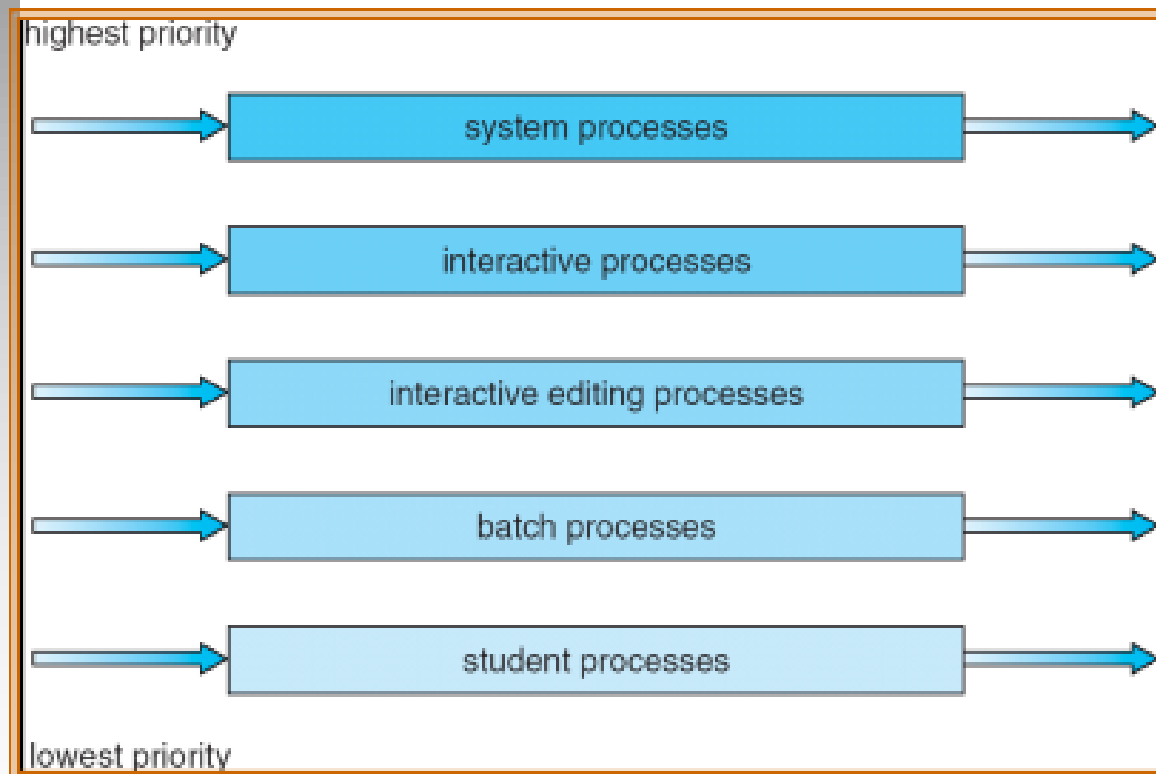
- No escalonamento com múltiplas filas, os processos em uma fila com uma prioridade específica podem ser tratados de forma diferente dos processos em outras filas, mas isso não implica necessariamente que os processos em uma fila com prioridade mais alta serão sempre executados antes dos processos em filas com prioridades mais baixas.

- A fila de prontos é separada em várias filas:
 - Fila para processos interativos
 - Fila para processos em lote
- Cada fila tem sua política de escalonamento
 - RR para processos interativos
 - FCFS para processos em lote
- Deve haver um escalonamento entre as filas
 - Prioridade: primeiro a fila de processos interativos possibilidade de abandono de processos (starvation)
 - Fatia de tempo para cada fila

Transferência entre filas

- Processos podem ser transferidos de fila
 - **Envelhecimento (aging):** processo vai ganhando prioridade com o tempo
- Escalonador com múltiplas filas pode ser definido através dos seguintes parâmetros:
 - Número de filas
 - Algoritmos de escalonamento para cada fila
 - Método utilizado para aumentar a prioridade do processo
 - Método utilizado para diminuir a prioridade do processo
 - Método utilizado para determinar qual fila o processo ficará quanto o mesmo solicita um serviço

Transferência entre filas



Múltiplas Filas

- Logo que surgir um processo com maior prioridade que o que está executando, ele “preempta” (retira de execução) o mesmo que volta para a fila dos “prontos”.
- Caso haja mais de um processo com uma dada prioridade, se aplica um segundo algoritmo de desempate.
 - Tipicamente Round-Robin;
 - Também pode ser um FIFO ou SJF.
- Neste caso, obtém-se uma lista por nível de prioridades.
 - Quando uma fila está vazia, considera-se a lista de prioridade inferior.
 - Múltiplas filas, com realimentação.
 - Pode ter um algoritmo distinto de desempate em cada fila.

Definição das prioridades

- **Estática:** a prioridade é dada na criação do processo, pelo sistema operacional, pelo usuário...
 - Problema: há risco de adiamento indefinido (**starvation**) para um processo com baixa prioridade.
- **Dinâmica:** a prioridade evolui durante o ciclo de vida do processo. Começa com um valor estático
 - Evolui depois:
 - Aumenta a medida que o processo usa a CPU;
 - Aumenta proporcionalmente à fração do quantum que não usou.
 - Assim, os processos I/O bound voltam na fila de espera com alta prioridade!

Escalonamento Garantido

- Visa garantir um nível mínimo de recursos ou tempo de CPU para determinados processos ou threads. Isso é especialmente relevante em sistemas que suportam aplicações de tempo real ou críticas em que é necessário garantir que certas operações sejam concluídas dentro de prazos específicos.
- Para implementar o escalonamento garantido, os sistemas operacionais podem usar algoritmos de escalonamento específicos que reservam recursos para processos garantidos e ajustam dinamicamente a alocação de recursos com base nas necessidades atuais do sistema e nas prioridades dos processos.

Escalonamento Lotérico

- Escalonamento lotérico (ou "lottery scheduling" em inglês) é uma estratégia de escalonamento utilizada em sistemas operacionais multitarefa. Nessa abordagem, cada processo recebe bilhetes virtuais que são inseridos em uma loteria. O escalonador então seleciona aleatoriamente um bilhete da loteria para determinar qual processo será executado em seguida.
- A ideia principal por trás do escalonamento lotérico é que a probabilidade de um processo ser escolhido para execução é diretamente proporcional ao número de bilhetes que ele possui. Isso significa que os processos com mais bilhetes têm uma maior probabilidade de serem escolhidos, enquanto os processos com menos bilhetes têm uma probabilidade proporcionalmente menor.