Avisos:

Mudança da P3 – De 30/11 para 07/12 – Motivo Feira das profissões (29 e 30/11).

Mudança da Substitutiva – De 09/12 para 14/12.

Dia 28/10 – Não haverá aula. Dia do servidor público.

SO33B - Sistemas Operacionais

Parte 3 – Processos e Threads

Gerência do Processador

Matheus F. Mollon
21/10/2022

Sumário

- Introdução
- Funções Básicas
- Critérios de escalonamento
- Escalonamentos não-preemptivos e preemptivos
- Escalonamento FIFO
- Escalonamento SJF
- Escalonamento Cooperativo
- Escalonamento Circular (Round Robin)
- Escalonamento Por Prioridades

Sumário

- Escalonamento Circular Com Prioridades
- Escalonamento Por Múltiplas Filas
- Escalonamento Por Múltiplas Filas Com Realimentação
- Política de Escalonamento em Sistemas de Tempo Compartilhado
- Política de Escalonamento em Sistemas de Tempo Real

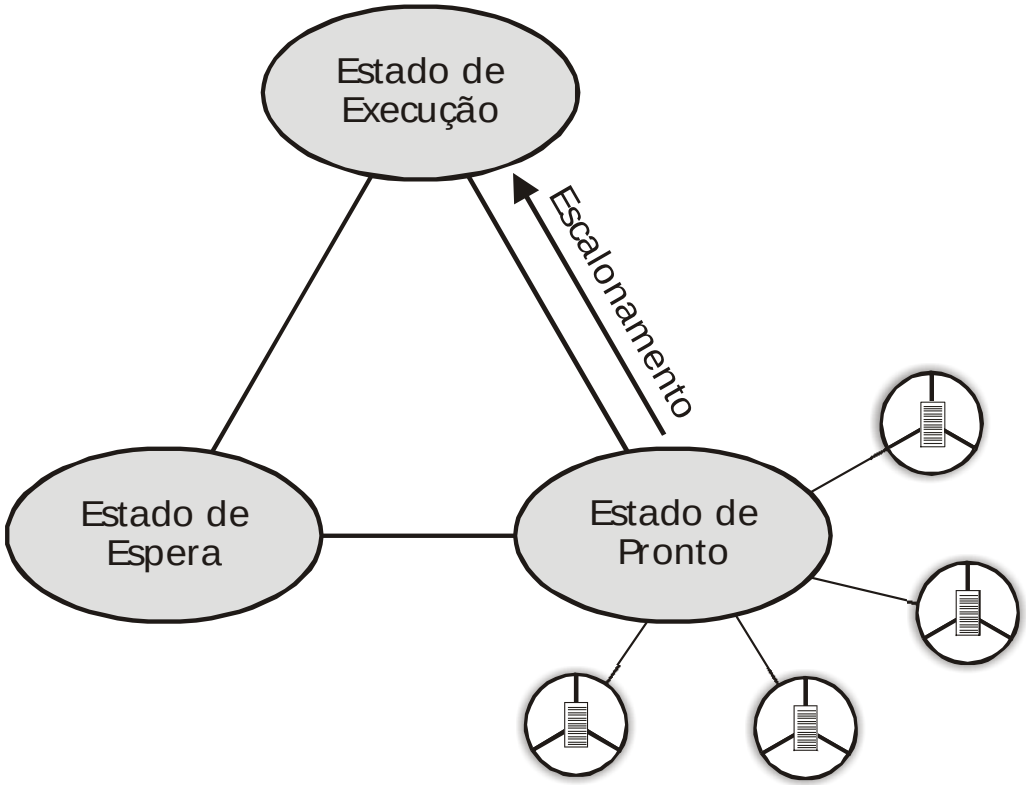
Introdução

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

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

Introdução

- 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

- Manter a UCP ocupada a maior parte do tempo
- Balancear o uso da UCP entre processos
- Privilegiar a execução de aplicações críticas
- Maximizar o *throughput* (processos executados em um intervalo de tempo)
- Oferecer tempos de resposta razoáveis para usuários interativos

Funções Básicas

- Cada SO possui sua política de acordo com seu propósito e suas características.

- Sistemas de tempo compartilhado têm requisitos de escalonamento diferentes dos sistemas de tempo real.

Funções Básicas

- A política de escalonamento é implementada por 2 rotinas principais:
  - O Escalonador (Scheduler), implementa os critérios da política de escalonamento e determina o próximo processo a entrar em execução.
  - O Dispatcher realiza a troca de contexto dos processos conforme definido pelo Scheduler. O tempo gasto para troca dos processos é conhecido como Latência do Dispatcher.

Funções Básicas

- A política de escalonamento é implementada por 2 rotinas principais:
  - Observação: SOs que implementam threads, tratam os processos como unidades de alocação de recursos, e as threads como unidades de escalonamento.

Critérios de Escalonamento

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

- Sistemas de tempo compartilhado → exigem que o escalonamento trate todos os processos de forma igual → evitando o starvation.

- Sistemas de tempo real → o escalonamento deve priorizar a execução de processos críticos em detrimento da execução de outros processos.

Critérios de Escalonamento

- Utilização do Processador: Processador deve permanecer ocupado maior parte do tempo.

- Throughput: Representa o número de processos executados em função do tempo. (processos / intervalo de tempo) → Sua maximização é desejada na maioria dos sistemas.

- Tempo de Processador ou Tempo de CPU: De todo o tempo necessário para executar o processo, é o tempo em que ele ficou efetivamente em estado de execução. Depende apenas do código da aplicação e da entrada de dados.

Critérios de Escalonamento

- Tempo de Espera: tempo total que um processo permanece na fila de pronto durante seu processamento, aguardando para ser executado.

- Tempo de Turnaround: Tempo total de execução, da criação ao encerramento. Leva em conta o tempo de espera, tempo de CPU e na fila de espera (operações de E/S).

- Tempo de Resposta: Tempo entre requisição (tecla digitada) e resposta (exibição no monitor). Crítico em sistemas on-line (rede, Web).

Critérios de Escalonamento

- De maneira geral, qualquer política de escalonamento busca:
- Otimizar a utilização do processador e o throughput.
- Diminuir os tempos de turnaround, espera e resposta.
- Dependendo do tipo de SO, um critério pode ter maior importância do que outros, como nos sistemas interativos, onde o tempo de resposta tem grande relevância.

Tipos de Escalonamento

- Políticas de Escalonamento podem ser classificadas conforme a possibilidade do SO interromper um processo em execução e substituí-lo por outro (Preempção):

- Escalonamento não-preemptivo:
  - Primeiro tipo de escalonamento implementado (para processamento batch).
  - Quando um processo está em execução nenhum evento externo pode ocasionar a perda do uso do processador.
  - Processo só sai do estado de execução ao fim do processamento ou por mudança para estado de espera.

Tipos de Escalonamento

- Políticas de Escalonamento podem ser classificadas conforme a possibilidade do SO interromper um processo em execução e substituí-lo por outro (Preempção):

- Escalonamento preemptivo:
  - O SO pode interromper um processo em execução e passá-lo para o estado de pronto, com o objetivo de alocar outro processo na UCP.
  - Permite uso de prioridades (Tempo Real, por exemplo).
  - Permite compartilhamento uniforme (balanceamento).
  - Mais flexível, é o tipo de escalonamento usado atualmente pela maioria dos SOs.

Escalonamento FIFO

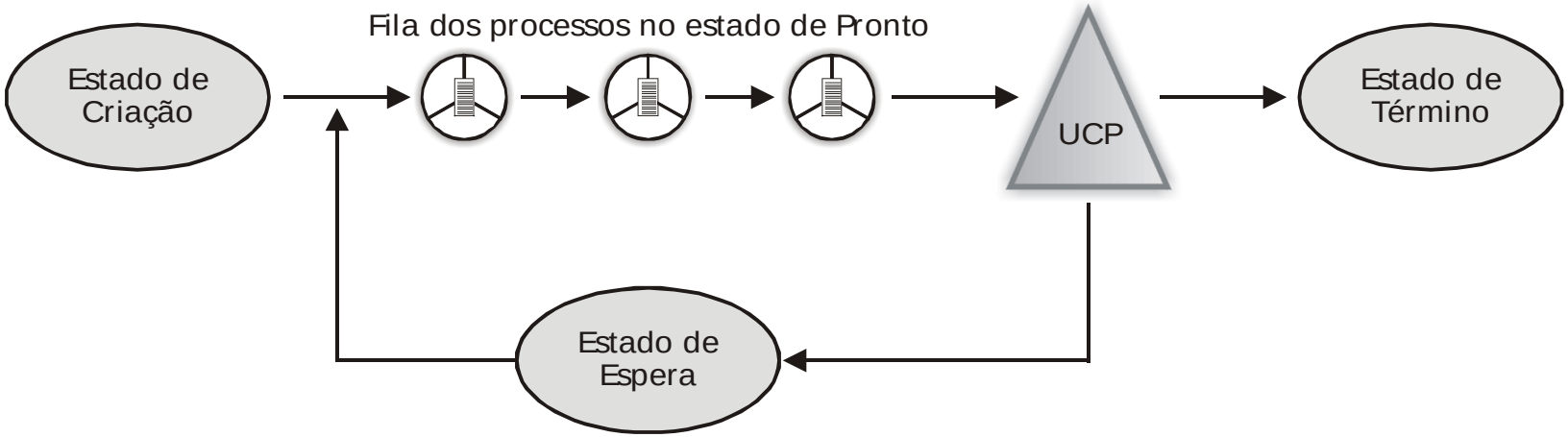
- FIFO (First-In-First-Out) ou FCFS Scheduling (First-Come-First-Served) → processo que chega primeiro ao estado de pronto é o primeiro a ser executado.

- Bastante simples, controlado por uma fila de processos em estado Pronto: processo que passa para Pronto entra no final da fila, e é escalonado quando chega ao início da fila.

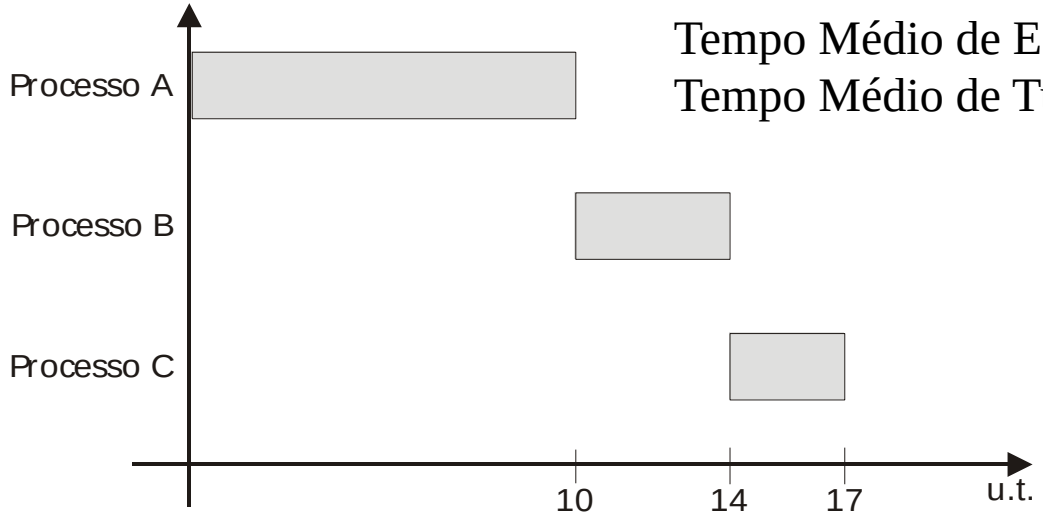
- Quando processo sai de execução (acaba ou entra em estado de espera), o processo no início da fila é escalonado. Ao final da espera, processo volta ao final da fila.

Escalonamento FIFO

- Escalonamento First-In-First-Out

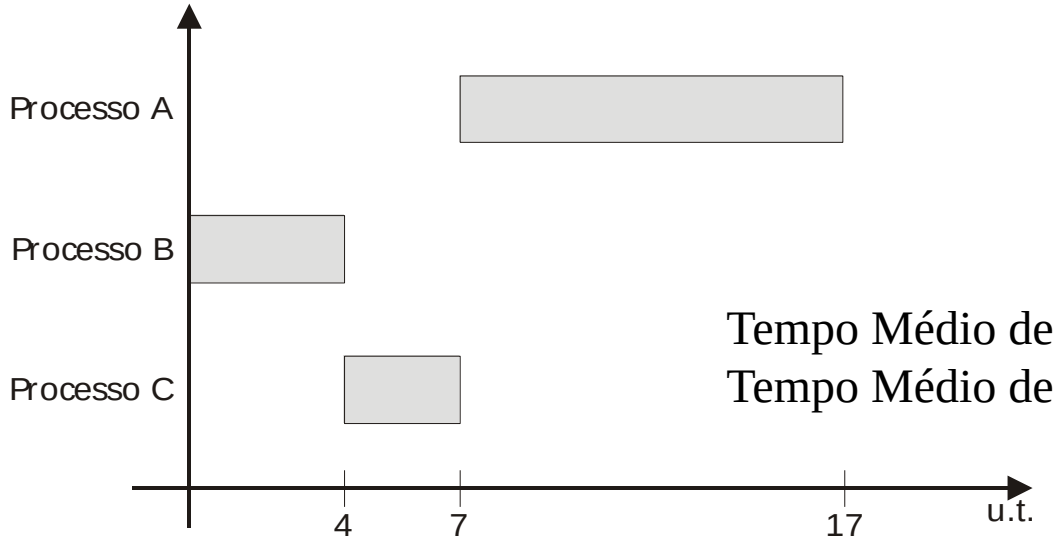


- Exemplo



Tempo Médio de Espera: (0+10+14)/3= 8
Tempo Médio de Turnaround: (10+14+17)/3= 13,7

Processo	Tempo de processador (u.t.)
A	10
B	4
C	3



Tempo Médio de Espera: (7+0+4)/3= 3,7
Tempo Médio de Turnaround: (17+4+7)/3= 9,3

Escalonamento FIFO

- Apesar de possuir implementação simples, apresenta alguns problemas:
  - Não se preocupa em melhorar tempo médio de espera, utilizando apenas a ordem de chegada na fila.
  - Piora Turnaround de processos que precisariam de pouco tempo de CPU.
  - Processos CPU-Bound levam vantagem sobre processos I/O-Bound. Se houver I/O-Bound mais importante, não haverá prioridade.

Escalonamento FIFO

- O escalonamento FIFO é do tipo não preemptivo e foi inicialmente implementado em sistemas com processamento batch.

- É ineficiente se aplicado em sua forma original. Atualmente, é utilizado com variações.

Escalonamento SJF

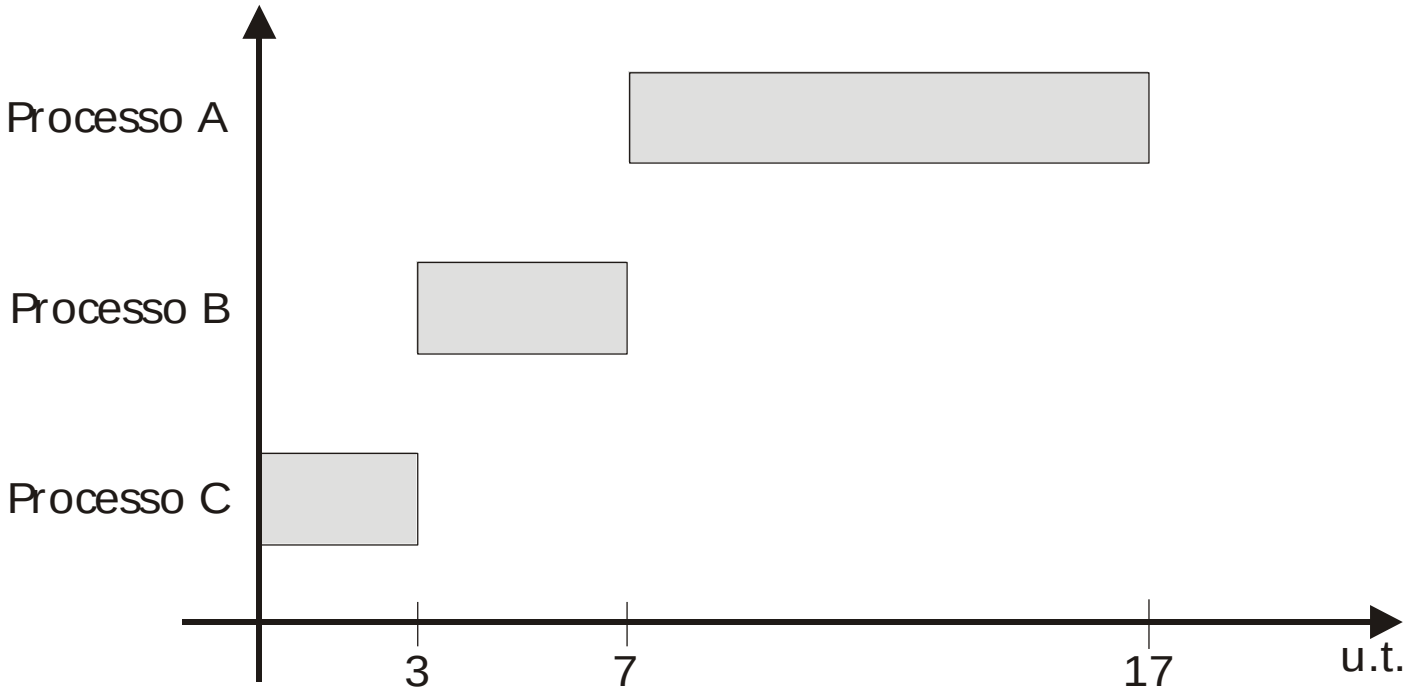
- Escalonamento shortest-job-first (SJF scheduling), também conhecido como shortest-process-next (SPN scheduling).

- O algoritmo de escalonamento seleciona o processo que tiver o menor tempo de processador ainda por executar.

- Desta forma, o processo em estado de pronto que necessitar de menos tempo de UCP para terminar seu processamento é selecionado para execução.

Escalonamento SJF

- Exemplo: Processos A, B e C com tempos de processador 10, 4 e 3 u.t. (unidades de tempo) e mesmo instante de criação.



Tempo Médio de Espera: (7+3+0)/3= 3,3
Tempo Médio de Turnaround: (17+7+3)/3= 9

Escalonamento SJF

- Tempo médio de espera diminui, pois processos mais rápidos são executados primeiro.

- Para cada novo processo é associado um tempo de processador ao seu contexto de software. Como não é possível precisar o tempo de processador previamente, o mesmo é estimado com base em análise de execuções passadas dos programas.

- Caso tempo estimado fosse muito inferior ao real, processo era interrompido

Escalonamento SJF

- Problema: impossibilidade de estimar o tempo para processos interativos.
- Na sua concepção inicial, o SJF é um escalonamento não preemptivo.
- Reduz tempo médio de turnaround, porém pode haver starvation para processos com tempo muito longo ou CPU-Bound.
- Variação: SJF com Preempção (SRT Scheduling-Shortest Remaining Time) → Se há um processo em pronto com tempo estimado menor que o processo em execução, são trocados.

Escalonamento Cooperativo

- Busca o aumento do grau de multiprogramação em sistemas não-preemptivos.

- Processo em execução pode voluntariamente liberar o processador, retornando à fila de pronto → possibilita que um novo processo seja escalonado → melhor distribuição do uso da UCP.

- A liberação do processador, portanto, é feita pelo processo em execução, que fica verificando se há outros na fila de pronto, se houver, o processo cooperativamente libera a CPU.

Escalonamento Cooperativo

- Podem ocorrer problemas pela falta de controle do SO: se um processo não verificar a fila, os demais não terão chance de ser executados até a liberação da UCP pelo processo em execução.

- Problema: um programa pode permanecer por um longo período de tempo alocando o processador.

- Era utilizado nas primeiras versões do Windows, com o nome de Multitarefa Cooperativa.

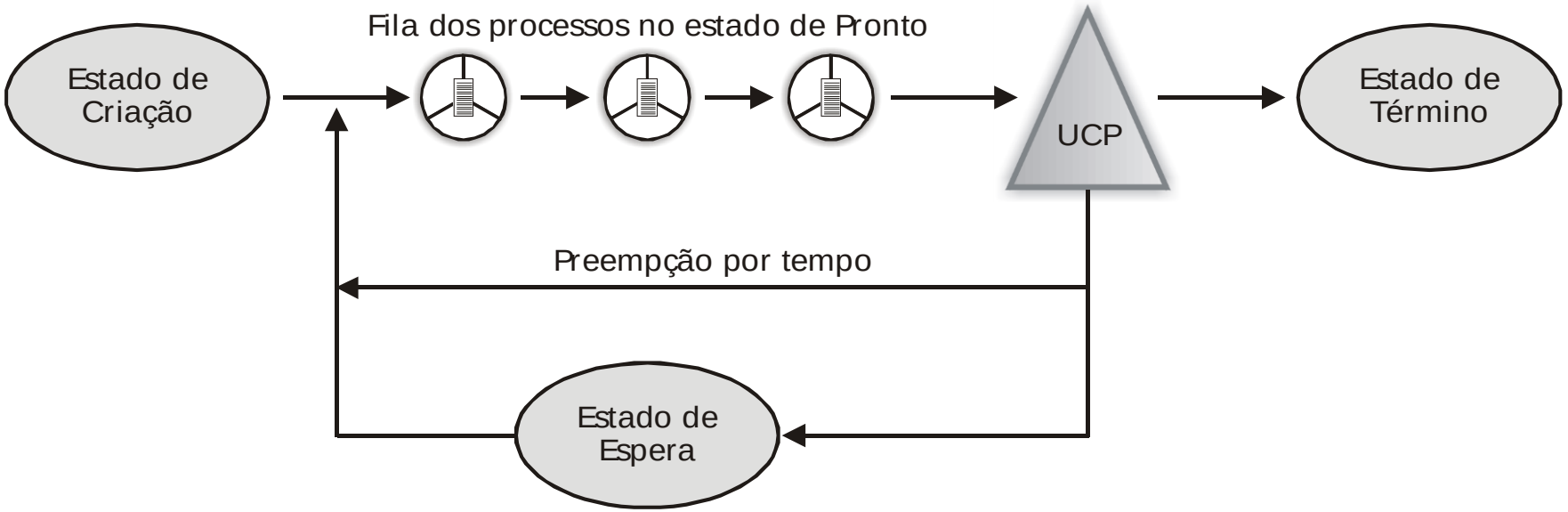
Escalonamento Circular

- Round Robin Scheduling → Projetado para sistemas de tempo compartilhado. Método Preemptivo.

- Semelhante ao FIFO, com a diferença de que utiliza time-slice (fatia de tempo) ou quantum.

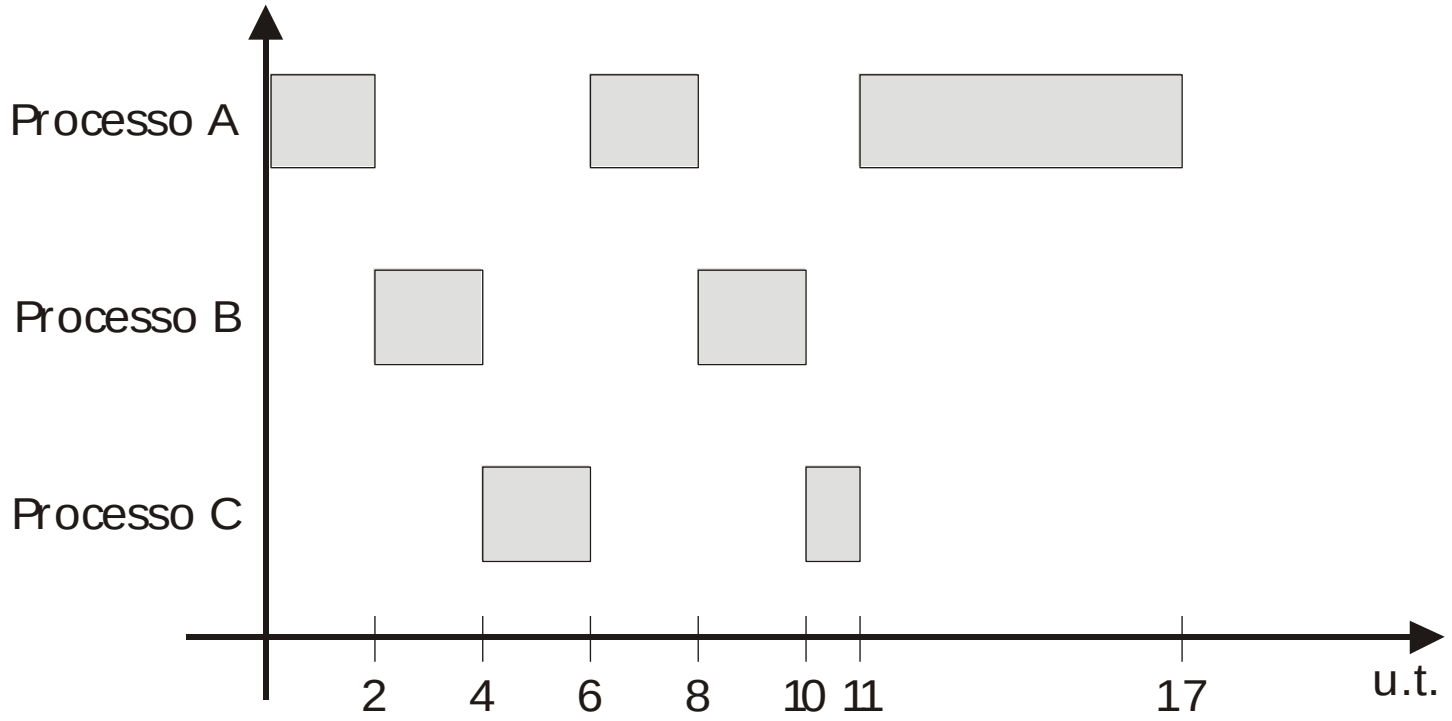
- Quando processo entra em execução, recebe um tempo limite para uso contínuo do processador. Quando tempo se esgota, processo é interrompido pelo SO, seu contexto é salvo e o processo volta para a fila de prontos (preempção por tempo).

Escalonamento Circular



Escalonamento Circular

- Exemplo - Fatia de Tempo = 2 u.t.



Escalonamento Circular

- O valor da fatia de tempo depende do SO, mas geralmente varia de 10 a 100 milissegundos.

- Este valor afeta diretamente o desempenho: valor muito alto provoca tendência do Round Robin funcionar como o FIFO, valor muito baixo gera muitas preempções, o que faz com que a latência do dispatcher afete o tempo de turnaround.

- Vantagem: não permite que um processo monopolize a UCP, o que é adequado para sistemas de tempo compartilhado.

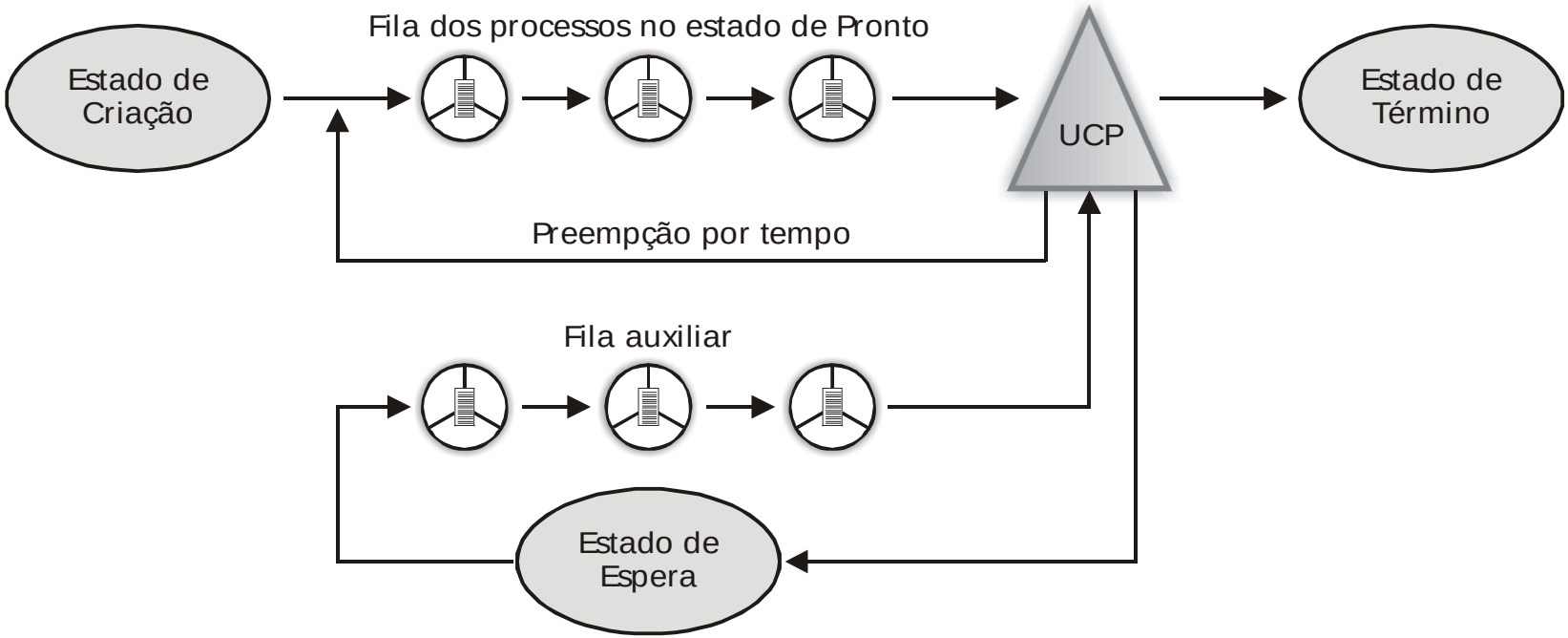
Escalonamento Circular

- Problema da política: CPU-Bound são beneficiados em detrimento dos I/O-Bound, pois CPU-Bound aproveitam toda a fatia de tempo, e I/O-Bound geralmente passam para o estado de espera antes da preempção por tempo.

- Estas características distintas ocasionam um balanceamento desigual no uso do processador entre os processos.

Escalonamento Circular

- Escalonamento circular virtual
  - Processos na fila auxiliar têm prioridade aos processos na fila de pronto.



Escalonamento por Prioridades

- É um escalonamento do tipo preemptivo.
- Baseado em um valor de prioridade (denominado prioridade de execução) associado a cada processo.
- Processo com maior prioridade na fila de prontos é escolhido para execução. Processos com valores iguais são escalonados por FIFO.
- Não há fatia de tempo, portanto, não há preempção por tempo.

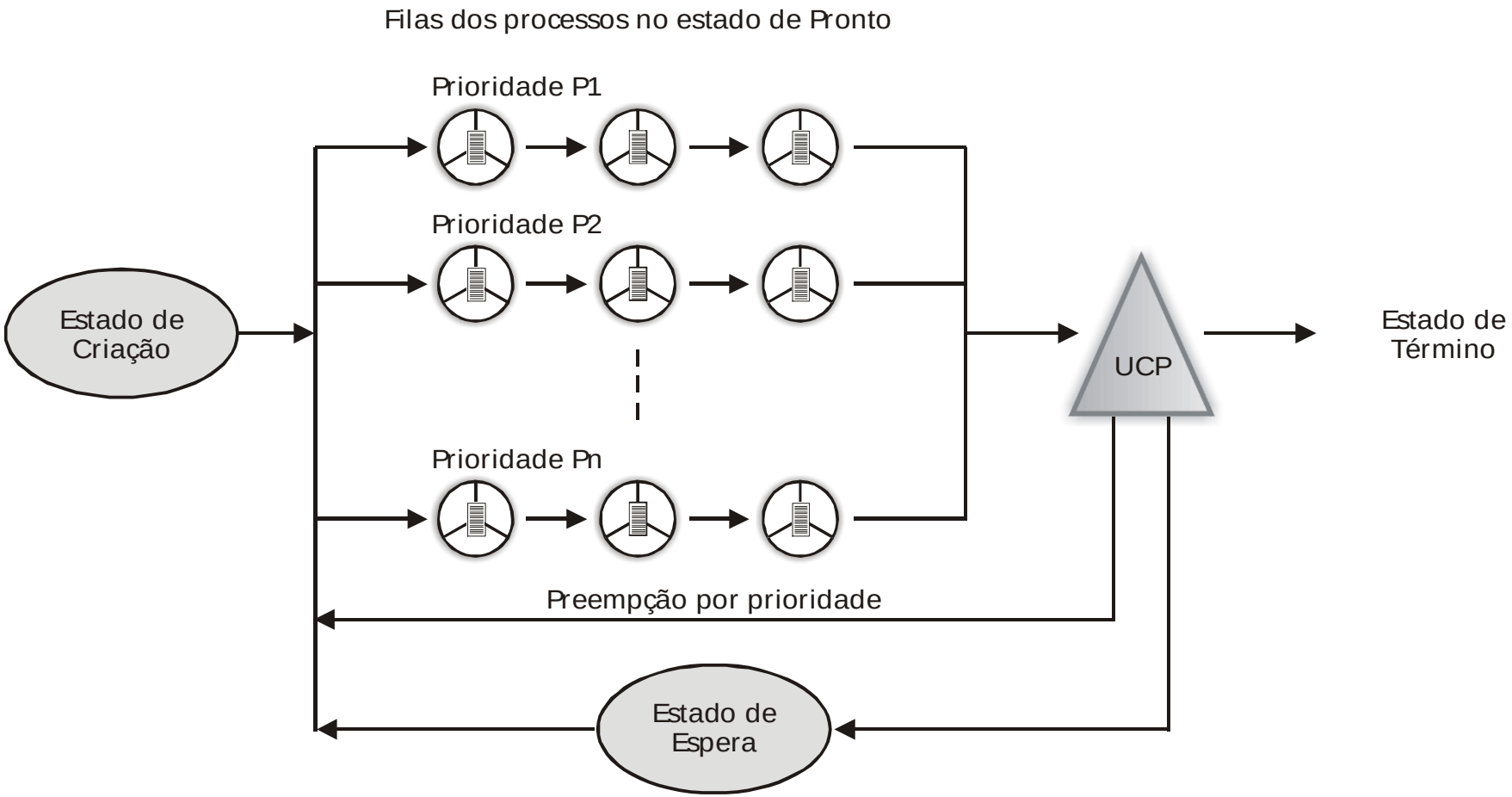
Escalonamento por Prioridades

- A perda do processador ocorre por mudança para estado de espera ou quando processo de maior prioridade passa para o estado de pronto (preempção por prioridade).

- É implementado por interrupção de clock, que faz com que de tempos em tempos o escalonador avalie as prioridades dos processos em estado de pronto.

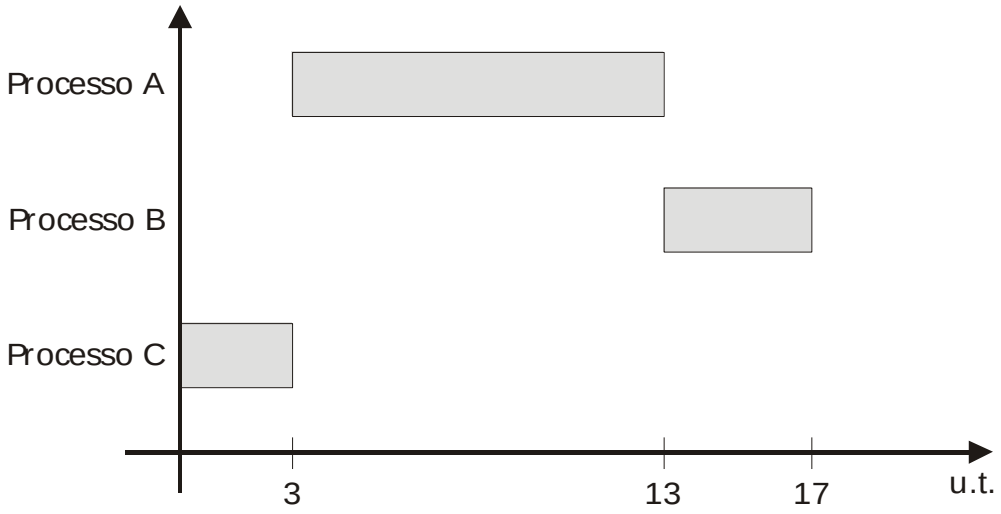
- Para cada prioridade existe uma fila de prontos, tratada como uma fila circular. Execução começa pela fila com prioridade mais alta.

Escalonamento por Prioridades



Escalonamento por Prioridades

- Exemplo com processos de prioridade diferentes (A-2, B-1, C-3), onde 5 é o valor de prioridade mais alta.



Processo	Tempo de processador (u.t.)	Prioridade
A	10	2
B	4	1
C	3	3

Escalonamento por Prioridades

- Cada SO implementa sua formatação para os valores de prioridade. Alguns associam valores altos às maiores prioridades, outros associam valores baixos.

- Prioridade faz parte do contexto de software do processo, e pode ser:

  - Estática: valor não muda durante a existência do processo.

  - Dinâmica: valor pode ser alterado durante execução, permitindo ajustar o critério de escalonamento em função do comportamento do processo.

Escalonamento por Prioridades

- Problema da política: Possibilidade de Starvation para processos de baixa prioridade.

- Solução: com prioridade dinâmica pode haver Aging, que incrementa a prioridade do processo conforme seu tempo na fila de prontos.

- Política de escalonamento por prioridades é bastante útil em sistemas de tempo real, e também em sistemas de tempo compartilhado, em que, às vezes, é necessário priorizar o escalonamento de determinados processos.

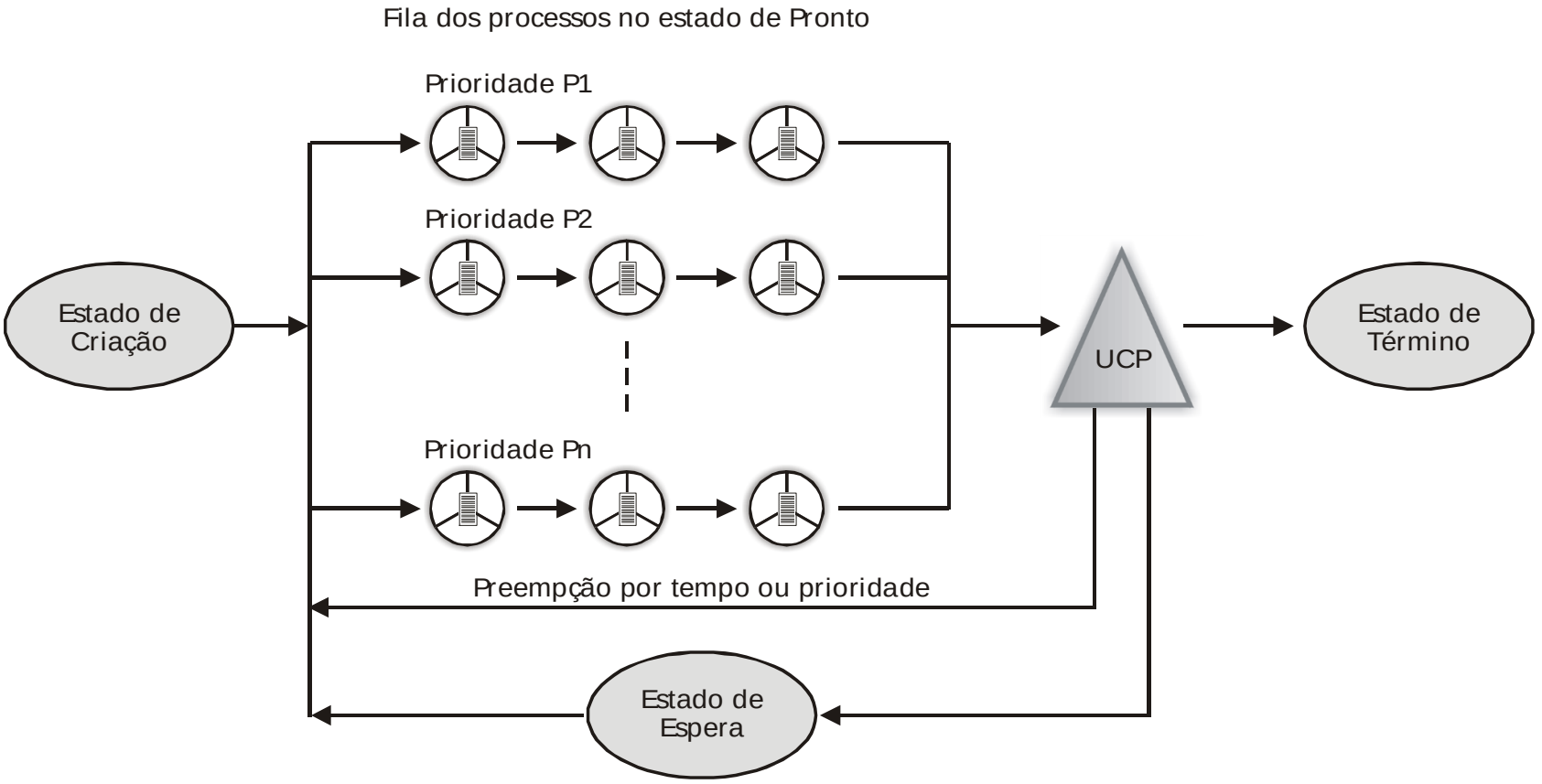
Escalonamento Circular com Prioridades

- Implementa o conceito de fatia de tempo e de prioridade de execução associada a cada processo.

- Neste tipo de escalonamento, um processo permanece no estado de execução até que:

  - Termine seu processamento;
  - Voluntariamente passe para o estado de espera;
  - Sofra uma preempção por tempo ou prioridade.

Escalonamento Circular com Prioridades



Escalonamento Circular com Prioridades

- A principal vantagem deste escalonamento é permitir o melhor balanceamento no uso do processador em sistemas de tempo compartilhado.

- Processos com o perfil I/O-bound devem receber do administrador do sistema prioridades com valores maiores que as dos processos CPU-bound.

- Isso permite ao sistema operacional praticar uma política compensatória entre processos de perfis distintos, compartilhando o processador de forma mais igualitária.

Escalonamento Circular com Prioridades

- O escalonamento circular apresenta duas variações:
  - Prioridades estáticas: a prioridade definida no contexto de software de cada processo permanece inalterada ao longo de sua existência.
  - Prioridades dinâmicas: é possível que a prioridade de um processo seja alterada dinamicamente pelo administrador do sistema ou, em algumas políticas, pelo próprio sistema operacional.

Escalonamento por Múltiplas Filas

- O Multilevel queue scheduling → há várias filas de processos em estado pronto, cada uma com prioridade específica.

- Processos são associados às filas em função de algumas características, como importância, tipo de processamento, área de memória necessária, etc.

- De acordo com o tipo de fila é escolhido o tipo de escalonamento, o que permite a implementação de mecanismos de escalonamento diferentes em um mesmo SO.

Escalonamento por Múltiplas Filas

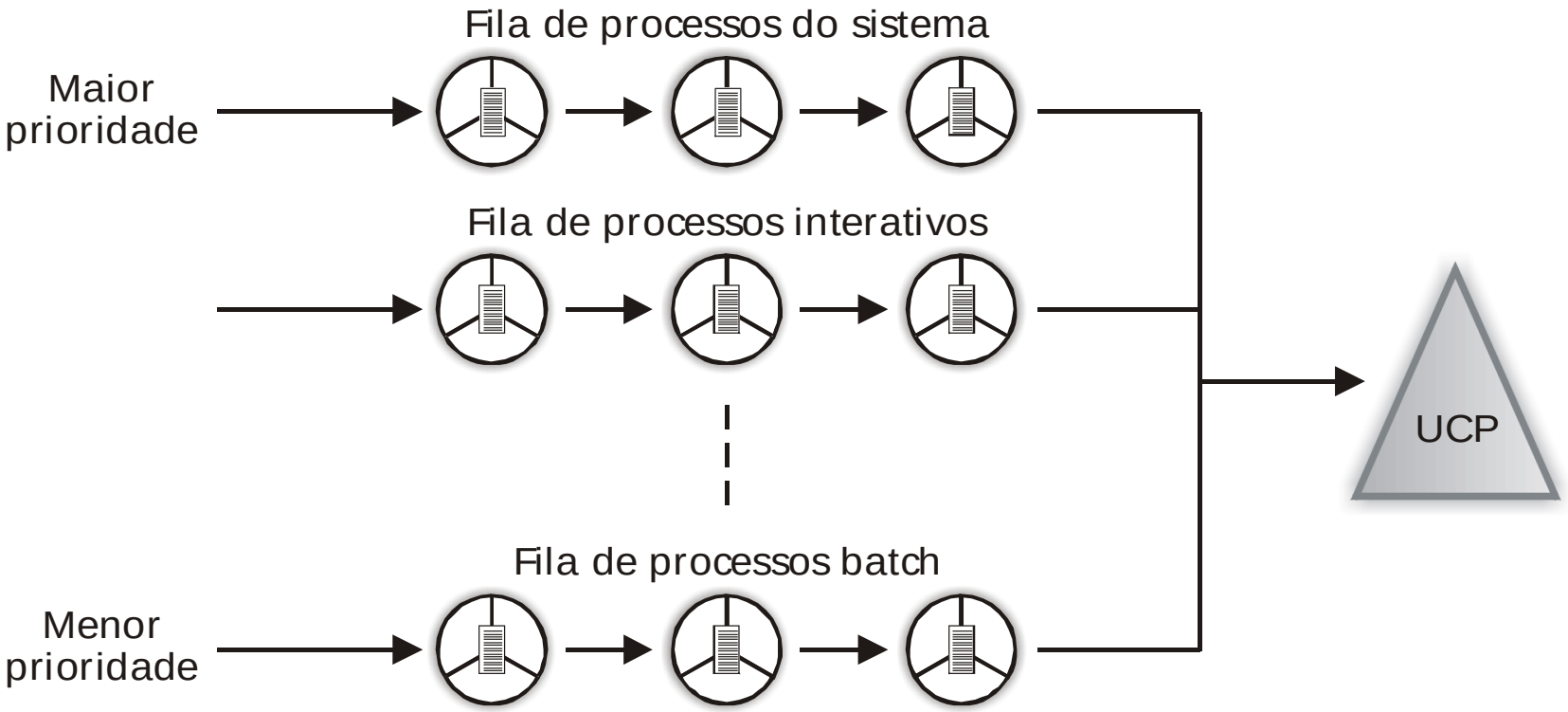
- Permitindo que alguns processos sejam escalonados pelo mecanismo FIFO, enquanto outros pelo circular.

- Característica de prioridade não está associada ao processo, mas sim à fila. Processo sofre preempção caso 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.

Escalonamento por Múltiplas Filas

Exemplo: processo de sistema, interativos ou batch. Fila de processos de sistema pode implementar escalonamento baseado em prioridades, enquanto as outras filas utilizam escalonamento circular.



Escalonamento por Múltiplas Filas

- Uma desvantagem deste escalonamento é que no caso de um processo alterar seu comportamento no decorrer do tempo, o processo não poderá ser redirecionado para uma outra fila mais adequada.

- A associação de um processo à fila é determinada na criação do processo, permanecendo até o término do seu processamento.

Escalonamento por Múltiplas Filas com Realimentação

- Multilevel Feedback Queues Scheduling → semelhante ao escalonamento por múltiplas filas, mas os processos podem trocar de fila durante o processamento.

- Utiliza recurso de ajuste dinâmico conhecido como Mecanismo Adaptativo: SO identifica dinamicamente o comportamento do processo e o direciona para a fila com prioridade mais adequada.

- É bastante complexo, mas atende às necessidades de diversos tipos de processamento.

Políticas em Sistemas de Tempo Compartilhado

- Caracterizam-se pelo processamento interativo, no qual usuários interagem com as aplicações exigindo tempos de respostas baixos.

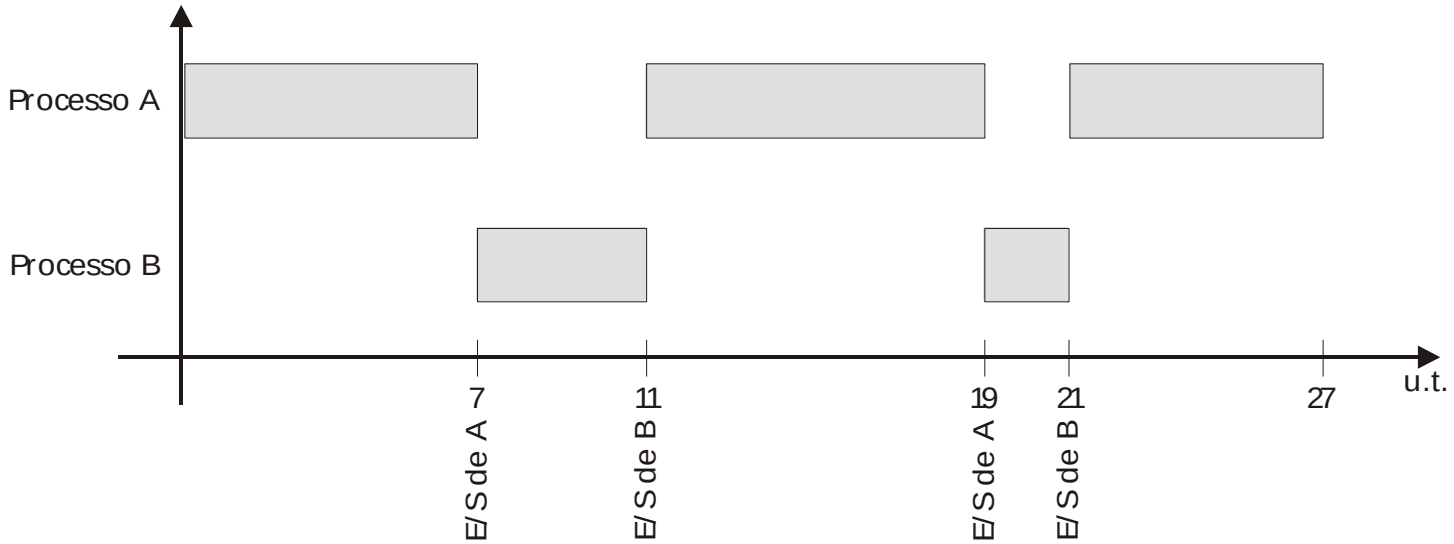
- A escolha de uma política de escalonamento para atingir este propósito deve levar em consideração o compartilhamento dos recursos de forma equitativa para possibilitar o uso balanceado da UCP entre processos.

- Para analisar a aplicação das Políticas de Escalonamento serão analisados dois processos (um CPU-Bound e um I/O-Bound) nos principais tipos de escalonamento apresentados.

Políticas em Sistemas de Tempo Compartilhado

- Escalonamento FIFO (exemplo)

Processo A → CPU-Bound
Processo B →I/O-Bound

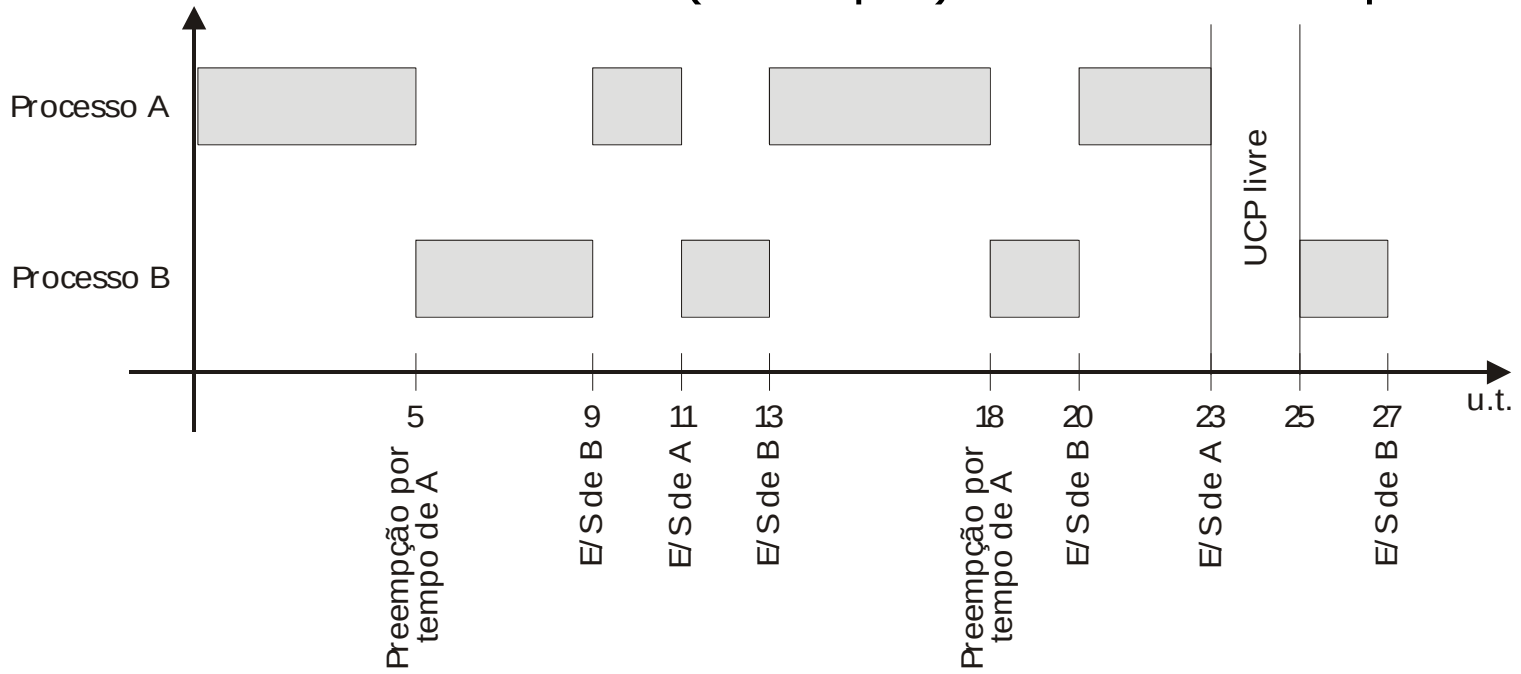


Processo	Tempo de processador (u.t.)	Característica
A	21	CPU-bound
B	6	I/O-bound

Não está balanceado, Processo B (I/O-Bound) permanece maior parte do tempo em espera.

Políticas em Sistemas de Tempo Compartilhado

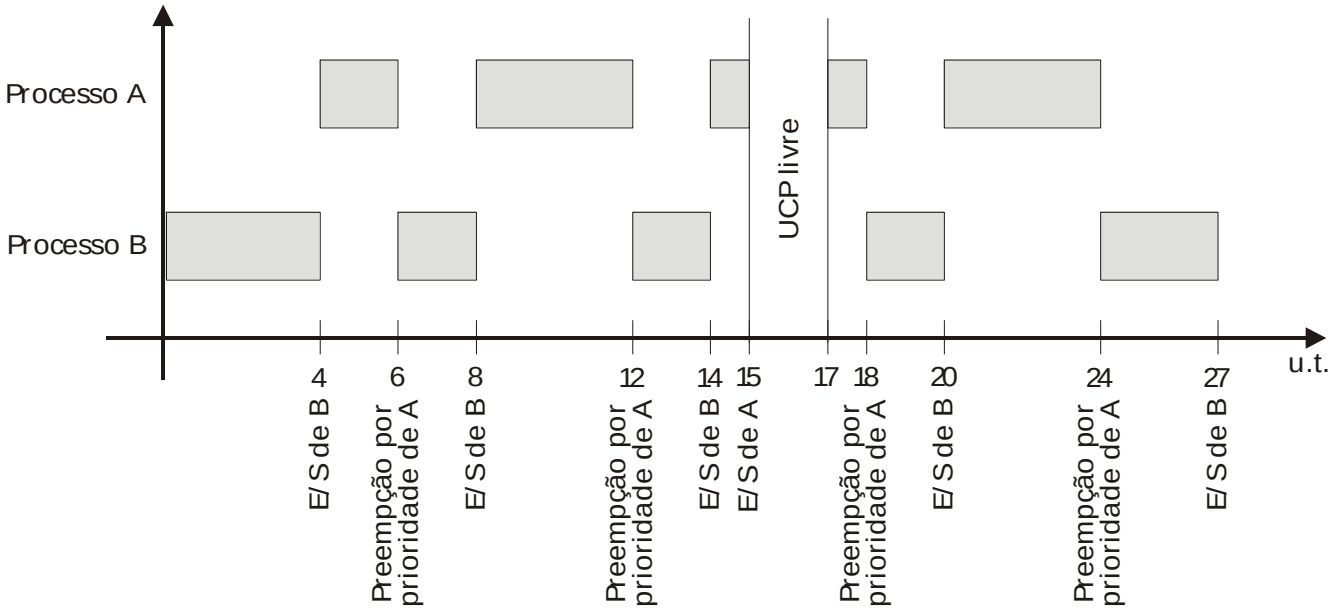
- Escalonamento Circular (exemplo) – Fatia de tempo = 5 u.t



Melhor distribuição da UCP, mas ainda não é a ideal, → trata todos os processos da mesma maneira, o que não é desejável (CPU-Bound tem mais oportunidades de uso do processador).

Políticas em Sistemas de Tempo Compartilhado

- Escalonamento circular com prioridades (exemplo)



Processo	Tempo de processador (u.t.)	Característica	Prioridade
A	12	CPU-bound	Baixa
B	13	I/O-bound	Alta

Políticas em Sistemas de Tempo Compartilhado

- Sistemas com prioridade dinâmica são mais complexos, porém o aumento de eficiência compensa. A maioria dos sistemas atuais de tempo compartilhado utiliza essa política de escalonamento.

Políticas em Sistemas de Tempo Real

- Em Sistemas de Tempo Compartilhado, as aplicações não são comprometidas pela variação do tempo de resposta.

- Para aplicações que exigem resposta imediata, devem ser utilizados Sistemas de Tempo Real, que garantem a execução de processos dentro de limites rígidos de tempo.

- Exemplos: Aplicações de controle de processos, como sistemas de controle de produção de bens industriais e controle de tráfego aéreo.

Políticas em Sistemas de Tempo Real

- Para esse tipo de sistema, o escalonamento por prioridades é o mais adequado, com cada processo recebendo uma prioridade de acordo com sua importância dentro da aplicação.

- Não há fatia de tempo e a prioridade deve ser estática.

.

Orientações Prova 2

Processos

- Estrutura do processo
- Estados do processo
- Mudanças de estado do processo
- Criação e eliminação de processos
- Processos independentes, subprocessos e threads
- Processos foreground e background
- Processos do sistema operacional
- Processos CPU-bound e I/O-bound
- Sinais

.

Threads

- Ambiente monothread
- Ambiente multithread
- Arquitetura e implementação

Sincronização de Processos

- Definição de concorrência
- Implementação de concorrência
- Problemas de Compartilhamento
- Exclusão Mútua
  - Soluções de Hardware e
  - Soluções de Software
- Sincronização Condicional
- Semáforos
- Monitores
- Troca de Mensagens (Não)
- Deadlocks

Exercício

Processo 1 (Cliente A)	Processo 2 (Cliente B)
/* saque em A */	/*saque em A */
1a. x := saldo_do_cliente_A;	2a. y := saldo_do_cliente_A;
1b. x := x - 200;	2b. y := y - 100;
1c. saldo_do_cliente_A := x;	2c. saldo_do_cliente_A := y;
/* deposito em B */	/* deposito em B */
1d. x := saldo_do_cliente_B;	2d. y := saldo_do_cliente_B;
1e. x := x + 100;	2e. y := y + 200;
1f. saldo_do_cliente_B := x;	2f. saldo_do_cliente_B := y;

Exercício

Supondo que os valores dos saldos de A e B sejam, respectivamente, 500 e 900, antes de os processos executarem, pede-se:

b. Quais os valores corretos esperados para os saldos dos clientes A e B após o término da execução dos processos?

c. Quais os valores finais dos saldos dos clientes se a sequência temporal de execução das operações for: 1a, 2a, 1b, 2b, 1c, 2c, 1d, 2d, 1e, 2e, 1f, 2f?