Capítulo 2

Processos e Threads

2.1 Processos
2.2 Threads
2.3 Comunicação interprocesso
2.4 Problemas clássicos de IPC
2.5 Escalonamento

Processos

O Modelo de Processo

Todos os computadores modernos podem executar várias tarefas ao mesmo tempo. Enquanto estiver rodando um programa de usuário, o computador pode ler um disco e imprimir em uma impressora. Em sistemas multiprogramados, o processador pode ser chaveado entre diversos programas, dando a cada um dezenas ou centenas de milissegundos de processamento. Apesar de só haver um programa executando em determinado instante de tempo, no curso de um segundo, o processador pode ter trabalhado para diversos programas, dando a ilusão do paralelismo de execução (pseudo-paralelismo). O tratamento rigoroso de múltiplas atividades realizadas em paralelo pelo sistema operacional não é tarefa fácil. Para um melhor entendimento da função de um Sistema Operacional, é necessário entender alguns conceitos básicos sobre: Processos, Arquivos, Chamadas de Sistema e Interpretador de Comandos.

Processos

O Modelo de Processo

Processo é um programa em execução, sendo constituído do código executável, dos dados referentes ao código, da pilha de execução, do valor do contador de programa, do valor do apontador de pilha, dos demais registradores do HW, além de um conjunto de outras informações necessárias à execução do programa. Um processador é compartilhado por vários processos, estes poderão ser interrompidos pelo sistema operacional, por exemplo, por ter excedido seu tempo do uso do processador compartilhado ou por uma solicitaçàò de um dispositivo de E/S do prórpio processo, sendo o processo temporariamente suspenso, há necessidade de restaurá-lo mais tarde, no exato ponto em que foi interrompido, para isso, todas as informações relativas ao processo precisam ser salvas em algum lugar durante o tempo que durar a suspensão da sua execução. Todas as informações sobre cada um dos processos são armazenadas em um tabela do S.O, denominada Tabela de Processos, um processo suspenso é composto por seu endereçamento e sua entrada na tabela de processo.

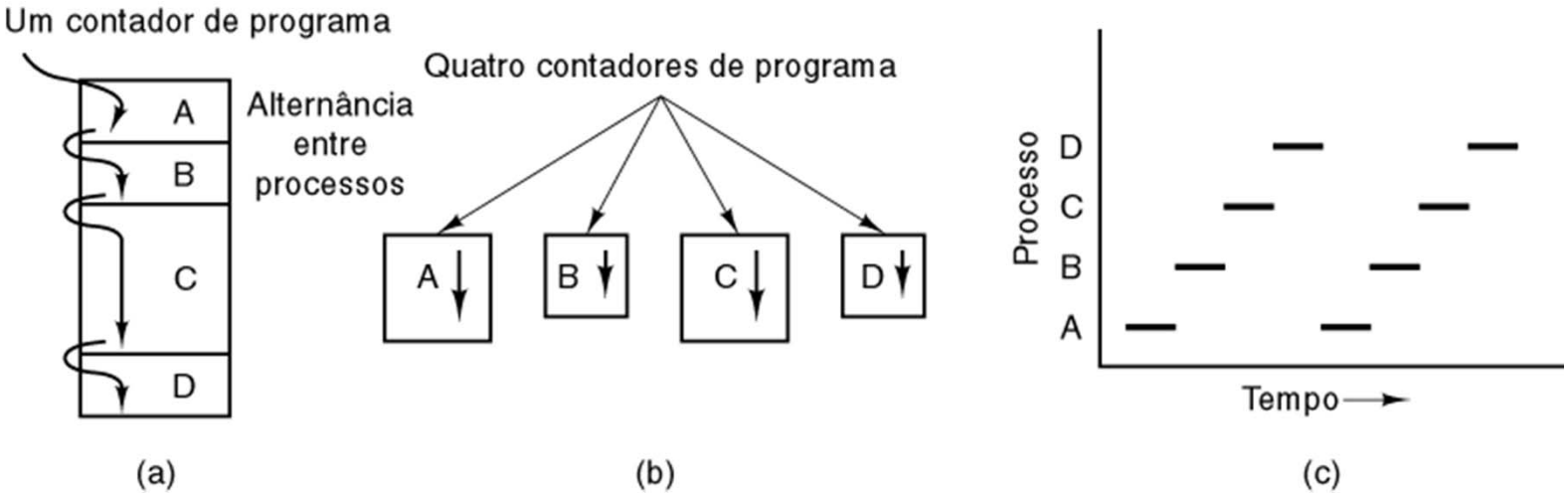
Processos

O Modelo de Processo

Todo software passível de rodar em um computador, muitas vezes incluindo o próprio sistema operacional, é organizado como um conjunto de processos seqüenciais ou simplesmente processos . Um processo nada mais é que um programa em execução, incluindo os valores correntes de todos os registradores do hardware, e das variáveis manipuladas por ele no curso de sua execução. O rápido chaveamento do processador entre diversos programas em execução (processos) é denominado de multiprogramação

A idéia central é que o processo constitui-se de um certo tipo de atividade, ele possui um programa, uma entrada, uma saída e um estado. Um único processador pode ser compartilhado entre vários processos, com o uso de algum algoritmo de escalonamento, para determinar quando o trabalho em um deles deve ser interrompido e qual o próximo processo a ser servido.

Processos
O Modelo de Processo



- Multiprogramação de quatro programas
- Modelo conceitual de 4 processos sequenciais, independentes
- Somente um programa está ativo a cada momento

Criação de Processos

Principais eventos que levam à criação de processos

1. Início do sistema
2. Execução de chamada ao sistema de criação de processos
3. Solicitação do usuário para criar um novo processo
4. Início de um job em lote

Término de Processos

Condições que levam ao término de processos

1. Saída normal (voluntária)
2. Saída por erro (voluntária)
3. Erro fatal (involuntário)
4. Cancelamento por um outro processo (involuntário)

Hierarquias de Processos

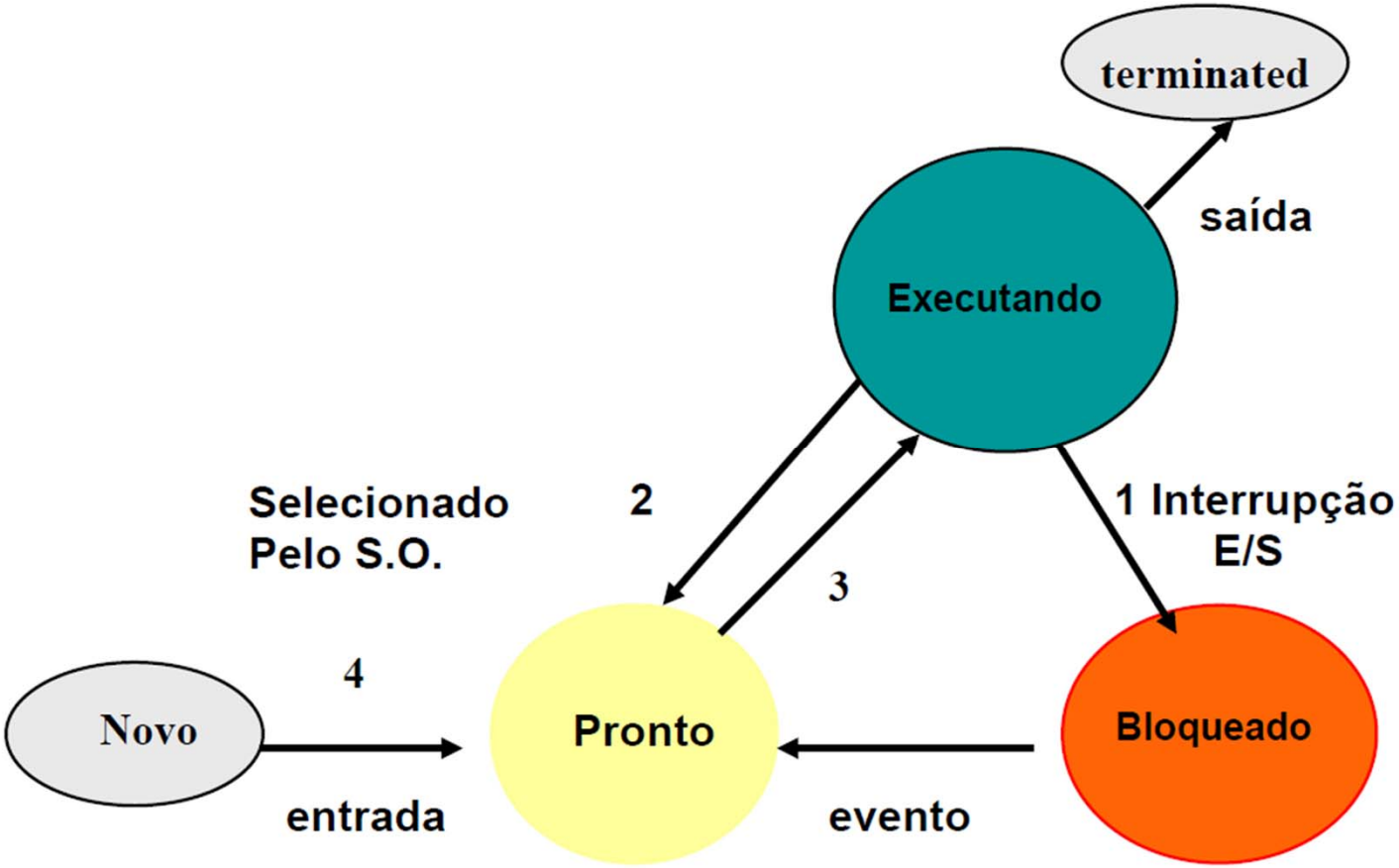
- Pai cria um processo filho, processo filho pode criar seu próprio processo
- Formam uma hierarquia
  - UNIX chama isso de “grupo de processos”
- Windows não possui o conceito de hierarquia de processos
  - Todos os processos são criados iguais

Estados de Processos (1)

1. O processo bloqueia aguardando uma entrada
2. O escalonador seleciona outro processo
3. O escalonador seleciona esse processo
4. A entrada torna-se disponível

- Possíveis estados de processos
  - em execução
  - bloqueado
  - pronto
- Mostradas as transições entre os estados

Estados de Processos (1)



Grafo de estados de um processo: **rodando**, **bloqueado** e **pronto**.
Grafo de transição de um processo: **(1)** Processo bloqueado para fazer entrada de dados. **(2)** Escalonador escolhe o novo processo. **(3)** Escalonador entrega o processador a esse novo processo. **(4)** Entrada de dados concluída.

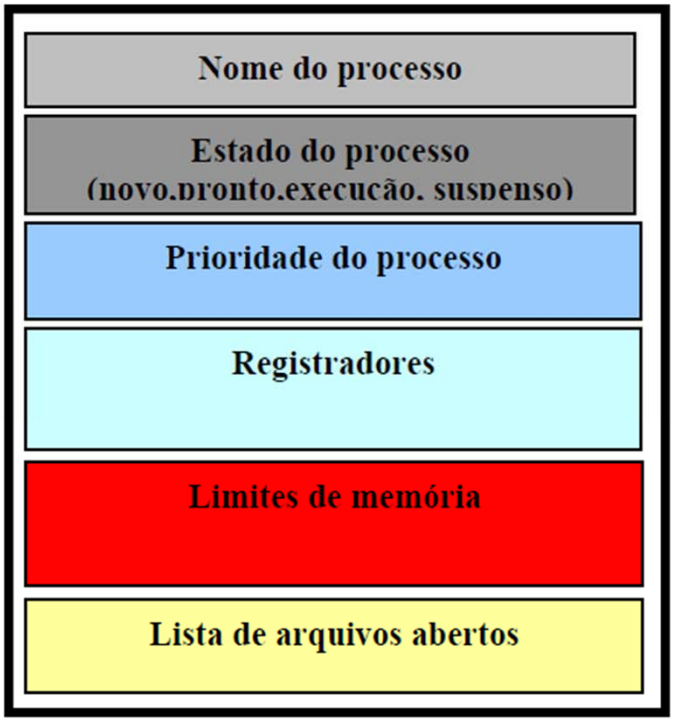
Estados de Processos (1)

Os estados de um processo são:
Running (Rodando).- usando o processador neste instante.
Blocked (Bloqueado).- impedido de rodar até que ocorra um determinado evento externo ao processo (bloqueado por uma solicitaçào de Entrada/Saída).
Ready (Pronto).- em condições de rodar, mas bloqueado temporariamente, pelo sistema operacional, para dar vez a um outro processo.

As transições de um processo são:
(1) Ocorre quando o processo descobre que não tem mais condições lógicas de prosseguir em seu processamento (em alguns sistemas, tal processo executa uma chamada de sistema, BLOCK, indo para o estado bloqueado, quando o processo precisa ler ou gravar)
(2) e (3) são causadas pelo escalonador de processos, uma rotina do sistema operacional, sem que o processo tome conhecimento delas. (2) ocorre quando o escalonador decide que o processo corrente já ocupou o processador por tempo suficiente, sendo portanto o momento de deixar que outro processo execute. (3) ocorre quanto todos os demais processos já tiveram oportunidade de rodar, sendo então a vez de o primeiro processo pronto rodar novamente. O escalonador de processos decide qual processo vai rodar e por quanto tempo.
(4) Ocorre quando acontece um evento externo pelo qual o processo estava aguardando, por exemplo a conclusão de uma operação de entrada/saída. Se nenhum outro processo estiver rodando neste momento, a transição 3 é acionada imediatamente, e o processo que aguardava a condição vai rodar. Caso contrário, ele vai para o estado de pronto até que o processador fique disponível e o escalonador escolha para rodar.

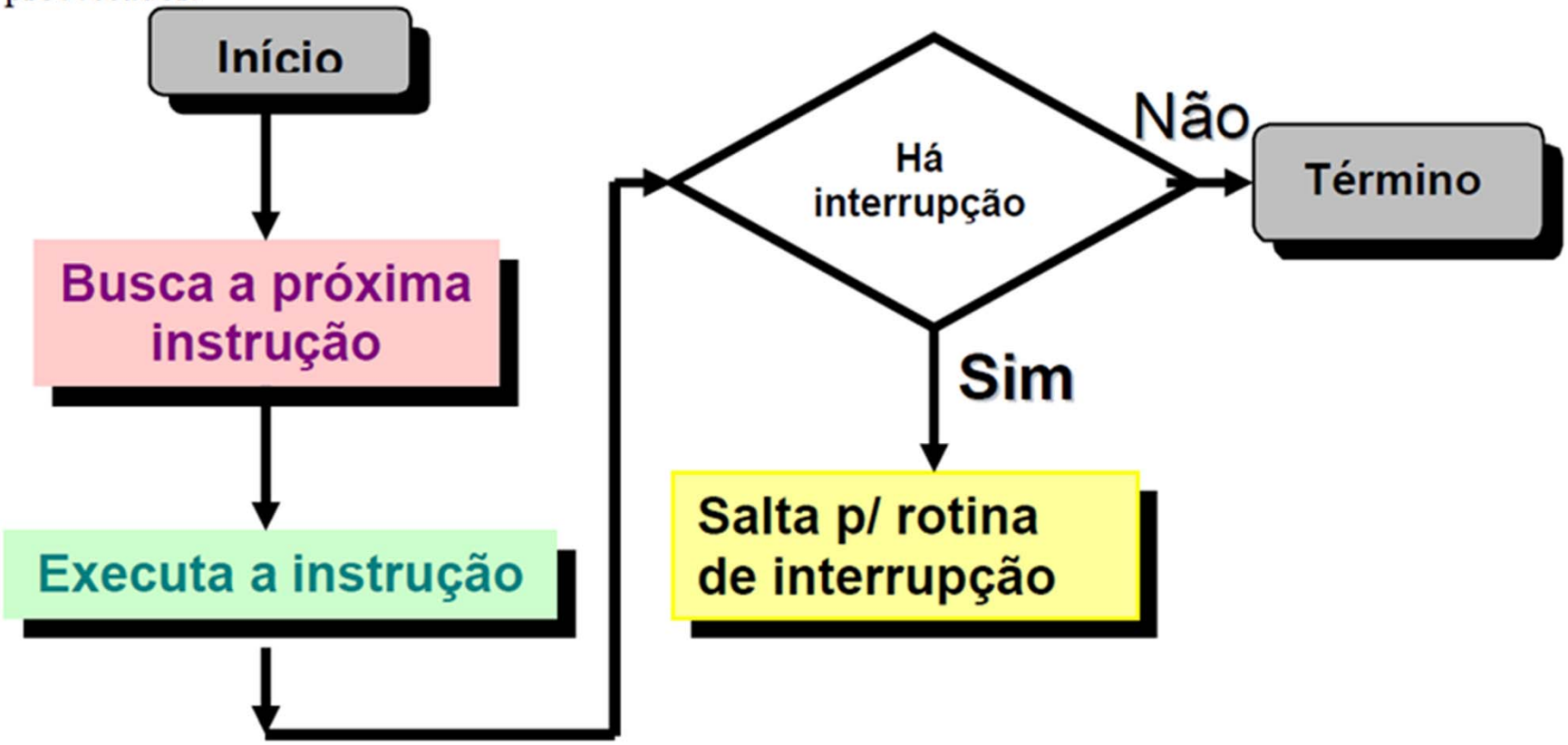
Estados de Processos (1)

O sistema Operacional materializa o **processo** através de uma estrutura chamada de bloco de controle de processo (BCP). A partir do BCP , o sistema mantém todas as informações sobre o processo, como sua identificação, prioridade, estado corrente, recursos alocados e informações sobre o programa em execução.

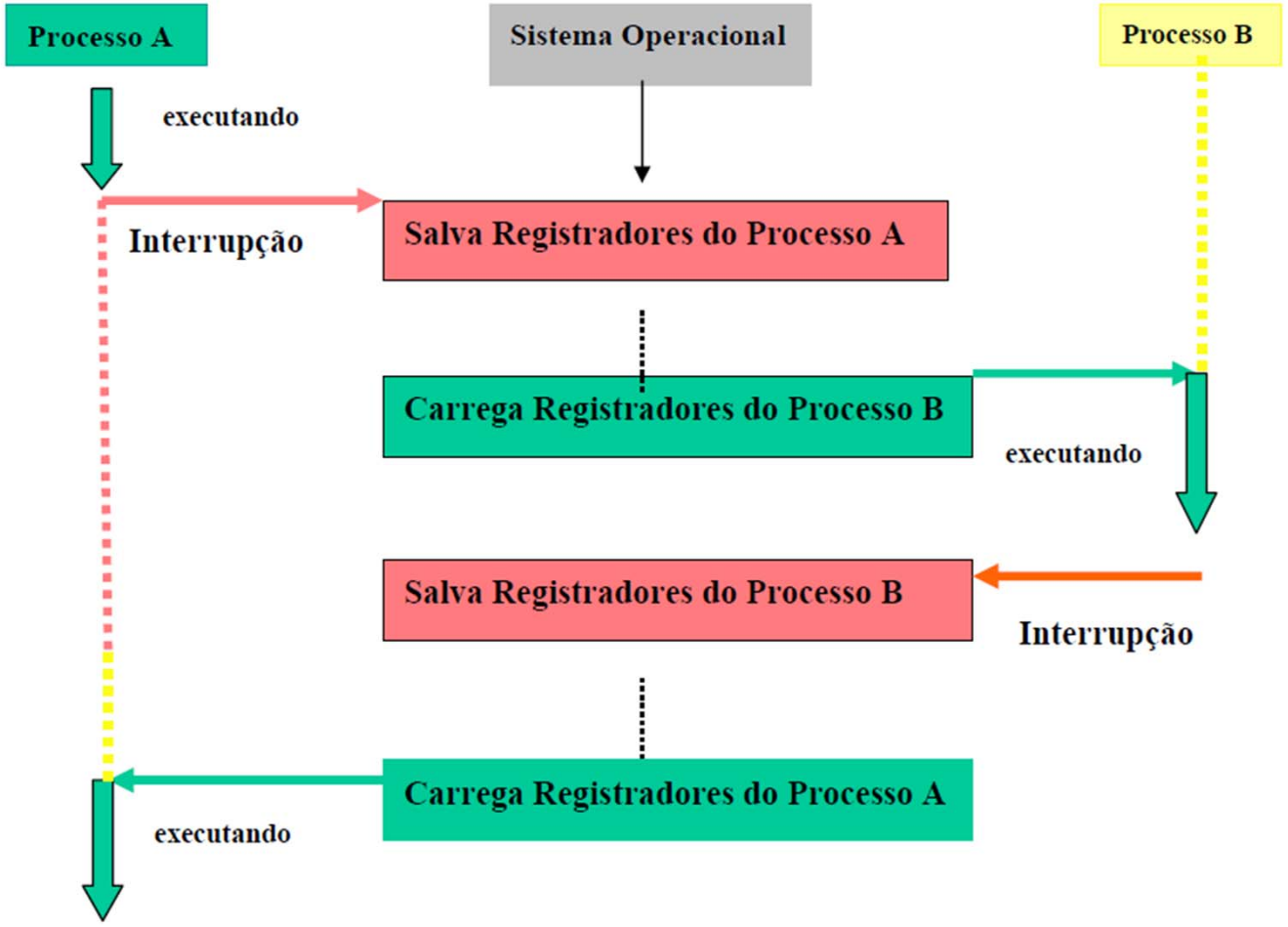


Estados de Processos (1)

A troca de um processo por outro, na CPU, realizada pelo sistema operacional, é denominada mudança de contexto , que consiste em salvar o conteúdo dos registradores da CPU e carrega-los com os valores referentes ao do processo que esteja ganhando a utilização do processador.

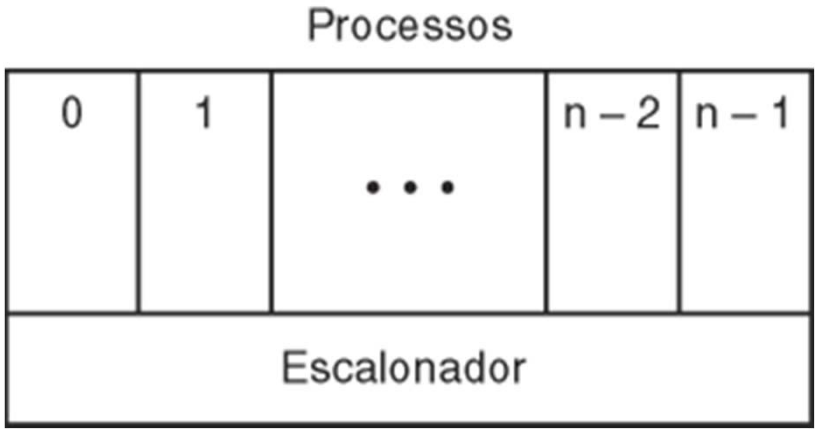


Estados de Processos (1)



BCP na troca de processos em execução pela CPU.

Estados de Processos (2)



- Camada mais inferior de um SO estruturado por processos
  - trata interrupções, escalonamento
- Acima daquela camada estão os processos sequenciais

Escalonamento de Processos

Os sistemas multiprogramáveis foram implementados pela possibilidade da CPU ser compartilhada entre diversos processos, eles possuem um critério para determinar qual a ordem na escolha dos processos para execução, entre os vários que concorrem pela utilização do processador. O procedimento de seleção dos processos para que use o processador (CPU) é uma das principais funções realizadas por um sistema operacional, sendo conhecido como escalonamento (é a troca da CPU entre os vários programas na memória prontos para executar), o responsável pelo escalonamento é chamado de escalonador.

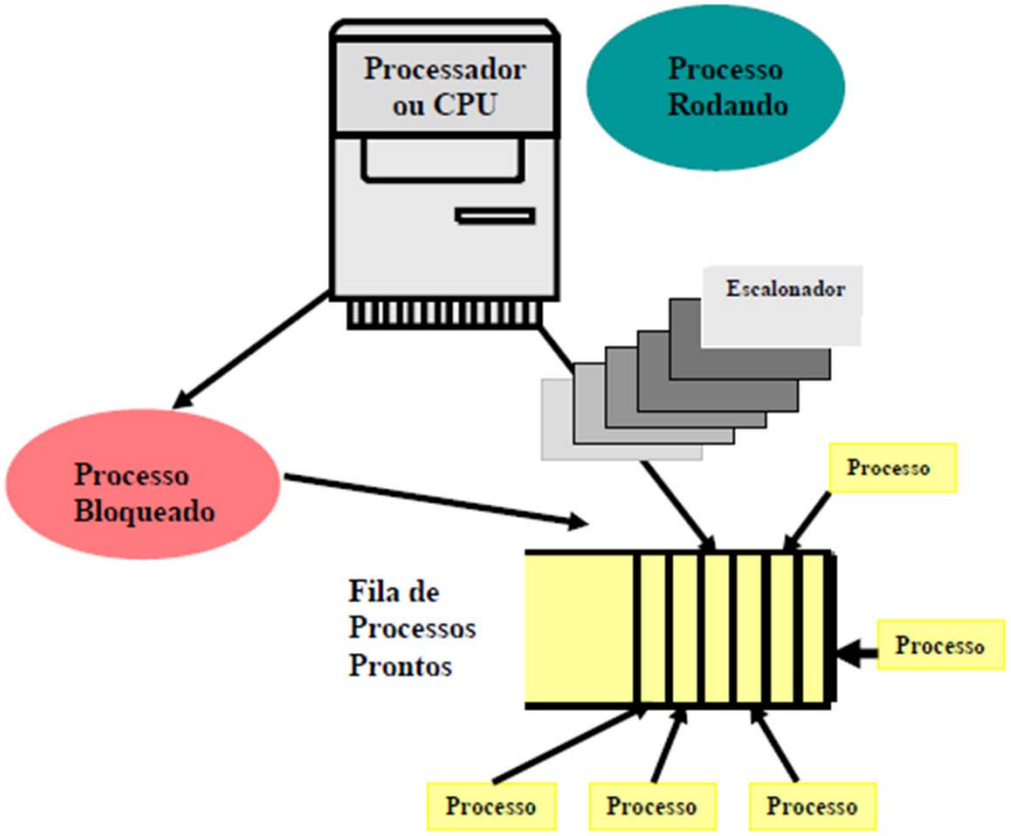
Os principais objetivos do escalonamento são, basicamente: manter a CPU ocupada a maior parte do tempo; balancear a utilização do processador entre diversos processos; maximizar o número de processos ativos ao mesmo tempo; e oferecer tempos de resposta razoáveis para os usuários interativos.

Algoritmos de Escalonamento

Um algoritmo de escalonamento tem como função decidir qual dos processos prontos para execução deve ser alocado à CPU. Cada sistema operacional necessita de um algoritmo de escalonamento adequado a seu tipo de processamento.

Algoritmos Não-Preemptivos de Escalonamento

Nos primeiros sistemas operacionais multiprogramáveis, onde predominava tipicamente o processamento batch, o escalonamento implementado era do tipo não-preemptivo. Nesse tipo de escalonamento, quando um processo ganha o direito de utilizar a CPU, nenhum outro processo pode lhe tirar esse recurso.



Algoritmos Não-Preemptivos de Escalonamento

Escalonamento com Algoritmo FCFS (First-Come-First-Served)

Primeiro processo que requisita a CPU é o primeiro que a recebe, isto é, o processo que chegar primeiro (first-come) é o primeiro a ser selecionado para execução (first-served), esta situação pode ser representado como apenas uma fila. Isto é

- **1. Processo pronto para executar entra no final da lista ready (fila de prontos)**
- **2. Quando a CPU é liberada é alocada para o primeiro da lista (primeiro da fila)**

Algoritmos Não-Preemptivos de Escalonamento

Exemplo : Dados 3 processos, com seus tempos de execução respectivamente.

Caso 1:

Processo	Tempo execução
P1	24
P2	3
P3	3

Tempo gasto pelos processos : **P1 = 24 P2 = 27 P3 = 30**

Tempo médio gasto pelos processos: Média = (24 + 27 + 30) / 3 = 27

Algoritmos Não-Preemptivos de Escalonamento

Caso 2:

Processo	Tempo execução
P2	3
P3	3
P1	24

Tempo gasto pelos processos : **P1 = 30** **P2 = 3** **P3 = 6**

Tempo médio gasto pelos processos: **Média = (30 + 3 + 6) / 3 = 13**

O desempenho do sistema pode ser comprometido, dependendo da ordem de chegada dos processos, o tempo de processamento, bem como o tempo de espera na fila..

Algoritmos Não-Preemptivos de Escalonamento

Escalonamento Com Algoritmo SJF(Shortest Job First)

O algoritmo SJF é apropriado para sistemas que rodam jobs em batch, nos quais o tempo de processamento de cada job é conhecido com antecedência. No escalonamento com algoritmo SJF, é selecionado para execução em primeiro lugar o processo que precisar de menos tempo de CPU para terminar o seu processamento.

Algoritmos Não-Preemptivos de Escalonamento

Exemplo : Dados 3 processos, com seus tempos de execução respectivamente.

Processo	Tempo execução
P1	6
P2	3
P3	8
P4	7

Tempo gasto pelos processos : **P2 = 3** **P1 = 9** **P4 = 16** **P3 = 24**

Tempo médio gasto pelos processos: **Média = (9 + 3 + 24 + 16) / 4 = 13**

Características:
média pequena dos tempos gastos para processar os processos
diminui tempo de espera de processos pequenos

usado pelo escalonador de longo prazo em sistemas batch
para fatias de execução iguais, utiliza-se FCFS.

Algoritmos Não-Preemptivos de Escalonamento

Escalonamento cooperativo

Esse tipo de escalonamento permite a implementação de sistemas multiprogramáveis com uma melhor distribuição do uso do processador entre processos. Sua principal característica está no fato de a liberação do processador ser uma tarefa realizada exclusivamente pelo processo em execução, que libera a CPU para um outro processo.

Um exemplo deste tipo de escalonamento pode ser encontrado nos sistemas Microsoft Windows 3.1 e Windows 3.11, sendo conhecidos como multitarefas cooperativas.

Algoritmos Preemptivos de Escalonamento

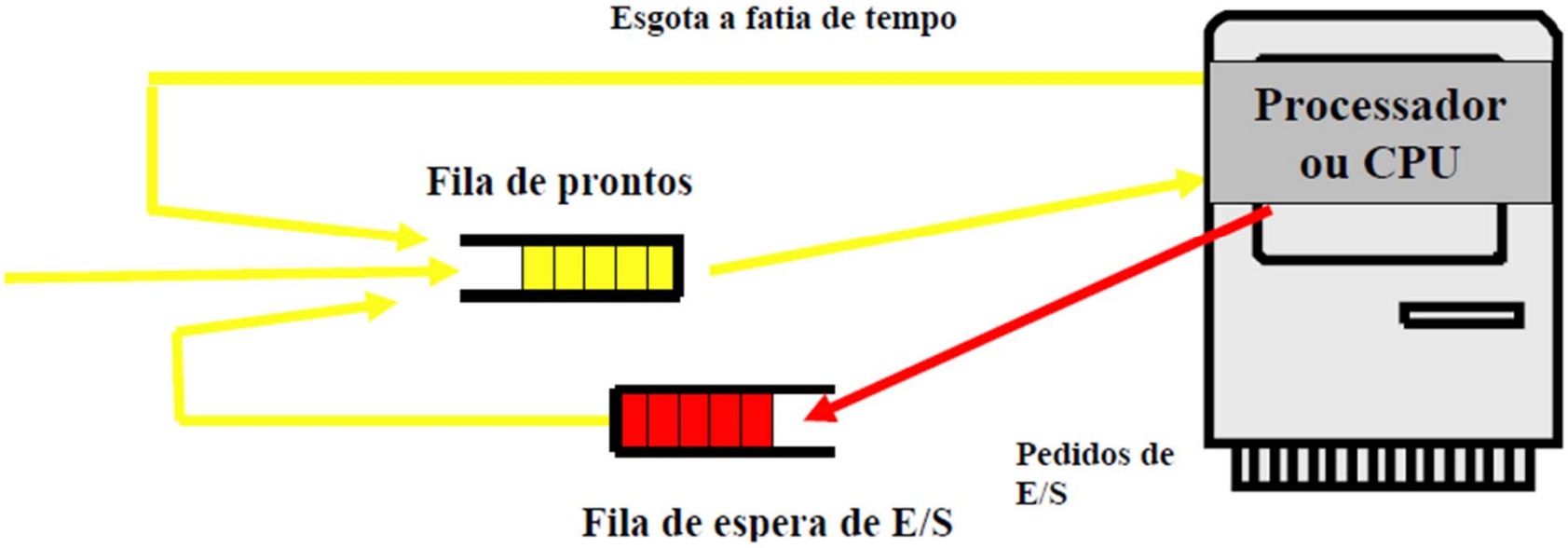
Um algoritmo de escalonamento é dito **preemptivo** quando o sistema pode interromper um processo em execução para que outro processo utilize o processador. Em sistemas que não implementam preempção, um processo pode utilizar o processador enquanto for necessário.

O escalonamento premptivo permite que o sistema dê atenção imediata a processos mais prioritários, como no caso de sistemas de tempo real, além de proporcionar melhores tempos de resposta em sistema de tempo compartilhado. Outro benefício decorrente deste tipo de escalonamento é o compartilhamento do processador de uma maneira uniforme entre os processos.

Algoritmos Preemptivos de Escalonamento

Escalonamento circular ou Round Robin

O escalonamento circular é implementados através de um algoritmo projetado especialmente para sistemas de tempo compartilhado (time sharing). Nesse algoritmo, quando um processo passa para o estado de execução, ou seja, ganha a CPU, existe um tempo-limite para sua utilização de forma contínua. Quando esse tempo, denominado *time-slice ou quantum,* expira, sem que antes a CPU seja liberada pelo processo, este volta ao estado de pronto, dando a vez para outro processo. Esse mecanismo é definido com *preempção por tempo*.



Algoritmos Preemptivos de Escalonamento

Escalonamento por prioridade

O escalonamento circular consegue a distribuição melhor do tempo de CPU em relação aos escalonamento não-preemptivos, porém não consegue implementar um compartilhamento eqüitativo entre os diferentes tipos de processos. Isso acontece em razão de o escalonamento circular tratar todos os processos de uma maneira igual.

Como alguns processos devem ser tratados de maneira diferente dos outros, é preciso associar a cada um deles uma prioridade de execução. Nesse esquema, processo de maior prioridade são escalonados preferencialmente. Toda vez que um processo for para a fila de pronto com prioridade superior ao do processo em execução, o sistema deve interromper o processo corrente, colocá-lo no estado de pronto e selecionar o de maior prioridade para ser executado. Esse mecanismo é definido como preempção por prioridade.

As prioridades podem ser atribuídas a processos de forma estática ou dinâmica. Em forma estática, por exemplo, em uma universidade, jobs de professores terão maior prioridade (100), os jobs de secretárias (50), os monitores (25) e jobs de alunos terão menor prioridae (10).

Algoritmos Preemptivos de Escalonamento

Prioridade dinâmica, jobs'que solicitam um número maior de vezes de dispositivos de E/S terão maior prioridade e jobs que solicitam um número menor de vezes de dispositivos de E/S terão menor prioridade. Algumas vezes, pode ser conveniente agrupar processos em classes de prioridades e usar escalonamento com prioridade entre classes e o round robin dentro de cada classe. A figura 2.9 mostra um sistema com quatro classes de prioridades. Enquanto houver processos prontos na classe de prioridade 4, tais processos serão escolhidos segundo o algoritmo round robin, não podendo ser escolhido nenhum processos pertencentes às classes de prioridade mais baixa.