O mecanismo de alocação da CPU para execução de processos constitui a base dos sistemas operacionais multiprogramados.

- A multiprogramação tem como objetivo permitir que, a todo instante, haja algum processo sendo executado, para maximizar a utilização da CPU. Em um sistema monoprocessador, existe sempre apenas um processo em execução. Outros processos têm que esperar até que a CPU esteja livre e possa ser novamente alocada para sua execução.
- A multiprogramação se baseia em uma idéia relativamente simples. Um processo é executado até que tenha que esperar pelo término de alguma operação de E/S.
- Em um sistema computacional simples, a CPU ficaria ociosa durante esse tempo de espera. O tempo de espera seria todo desperdiçado, sem realização de nenhum trabalho útil. Com a multiprogramação procuramos usar esse tempo de maneira produtiva. A cada instante, códigos de execução de diversos processos ficam armazenados na memória. Quando um processo tem que esperar, o sistema operacional transfere o controle da CPU para outro processo. Isso ocorre continuamente toda vez que um processo precisa esperar, outro processo pode assumir o controle da CPU.

## Fases de uso da CPU e de E/S

A execução de processos se dá em um ciclo, no qual se alternam a execução pela CPU e a espera por E/S. Os estados de processos se alternam continuamente entre um e outro. A execução de processos geralmente começa com uma atividade que usa a CPU intensivamente. Em seguida, temos em geral muitas operações de E/S, que são seguidas por mais uma fase de uso intensivo da CPU, depois outra fase de E/S e assim por diante. Por fim, a ultima fase de uso da CPU termina com uma requisição do sistema para o término da execução.

- Um programa cujo tempo de processamento depende mais do tempo de processamento de operações de E/S do que do tempo de processamento da CPU, chamado de programa que depende mais de E/S, tem normalmente várias fases muito curtas de uso da CPU. Por outro lado, um programa cujo tempo de processamento depende mais do tempo de processamento da CPU, chamado de programa que depende mais da CPU, tem em geral algumas fases em que a CPU é usada por um período muito longo de tempo. Essa distribuição pode ser importante na seleção de um algoritmo adequado para alocação da CPU.

## Escalonador da CPU

Sempre que a CPU se torna ociosa, o sistema operacional deve selecionar um processo para execução, da fila de processos prontos. A seleção de processos é realizada por um programa chamado escalonador da CPU. Esse programa seleciona um processo, dentre aqueles que estão na memória prontos para serem executados, e aloca a CPU para sua execução.

- Quando consideramos os vários algoritmos que podem ser usados por um escalonador, uma fila de processos prontos pode ser implementada como uma fila em que o primeiro a entrar é o primeiro a sair, como uma fila na qual os elementos são ordenados de acordo com a prioridade, como uma árvore ou simplesmente como uma lista ligada, não-ordenada.

## Alocação Preemptiva

Decisões de alocação da CPU podem acontecer nos seguintes casos:

1. Quando um processo **muda do estado em execução para em espera** (por exemplo, por uma requisição de E/S ou espera pelo término da execução de um de seus processos filhos).

2. Quando um processo **muda do estado em execução para pronto** (por exemplo, quando ocorre uma interrupção).

3. Quando um processo **muda do estado em espera para pronto** (por exemplo, pelo término de uma operação de E/S).

4. Quando um processo **termina.**

- Nos casos 1 e 4, **não há escolha a ser feita em termos de alocação.** Um processo novo deve ser selecionado para execução (se existir na fila de processos prontos).
- Nos casos 2 e 3, no entanto, **é preciso que uma decisão seja tomada, no projeto de um mecanismo de alocação da CPU.**
- **Nos casos 1 e 4, dizemos que a alocação é não-preemptiva.**
- **Nos outros, a alocação da CPU a um outro processo é preemptiva.**

Em um mecanismo de alocação não-preemptiva, uma vez que a CPU tenha sido alocada a um processo, esse processo mantém o controle da CPU até que termine ou seu estado mude para um estado de espera. Esse método de escalonamento é usado no ambiente Windows.

- Existe um custo para o uso da alocação preemptiva. Considere o caso em que dois processos compartilham alguns dados. Um dos

processos pode ser interrompido durante uma modificação desses dados, e o segundo processo passa então a ser executado. Esse segundo processo pode tentar lês os dados, que estão naquele instante em um estado inconsistente.

- Preempção tem também efeito sobre o projeto do núcleo do sistema operacional. Durante o processamento de uma chamada ao sistema, o núcleo pode estar realizando uma operação requerida por um processo. Tais operações podem envolver mudanças importantes em dados do núcleo (por exemplo, filas de E/S). O que acontece se esse processo for interrompido de forma preemptiva, no meio dessas mudanças, e o núcleo (ou a rotina de controle de um dispositivo) precisar ler ou modificar essa mesma estrutura? O resultado será um caos. Para tratar esse problema, alguns sistemas operacionais, como a maioria das versões do UNIX, só realizam mudanças de contexto depois que a execução de chamadas ao sistema ou operações de E/S terminam. Esse esquema permite que a estrutura do núcleo fique simples, uma vez que a execução de rotinas do núcleo não será interrompida, evitando que as estruturas de dados do núcleo fiquem em um estado inconsistente quando outros processos forem executados.

## Despachante

- Componente envolvido na alocação da CPU.
- Módulo que fornece o controle da CPU ao processo selecionado pelo escalonador. Essa função envolve:
  - Mudança de contexto.
  - Mudança para modo usuário.
  - Desvio para o endereço adequado no programa do usuário, para reiniciar o programa.
- O despachante deve ser o mais rápido possível, uma vez que ele é chamado durante cada transferência do controle da CPU. O tempo gasto pelo despachante para interromper a execução de um processo e iniciar a execução do outro processo é **chamado de latência de despacho.**

## Critérios de alocação

Os diferentes algoritmos de alocação da CPU apresentam características diferentes e podem favorecer uma ou outra classe de processos. Para escolher qual algoritmo usar em uma determinada situação, devemos considerar as características dos diversos algoritmos. Os critérios usados incluem os seguintes:

- Utilização da CPU
- Produtividade
- Tempo de processamento
- Tempo de espera

- Tempo de resposta

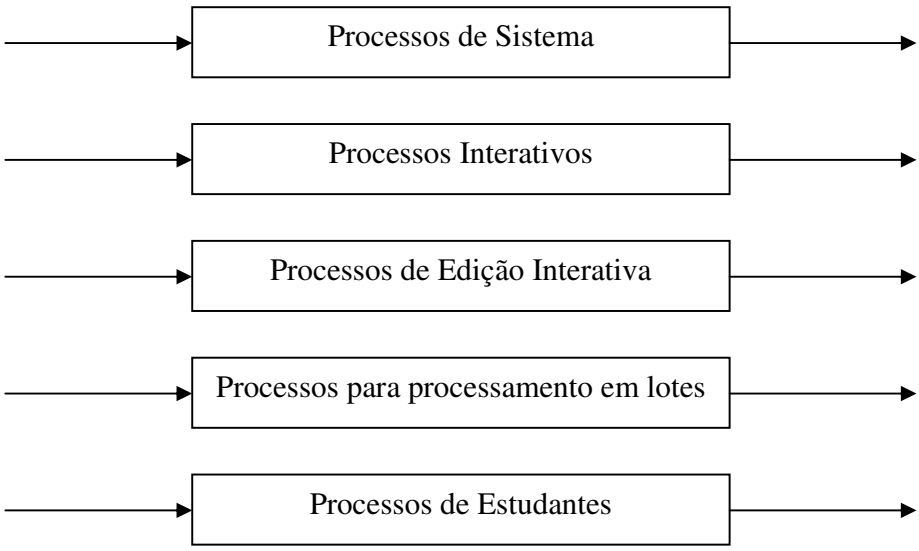
É desejável maximizar a utilização da CPU e a produtividade e minimizar os tempos de processamento, de espera e de resposta.

Na maioria dos casos, escolhemos algoritmos que maximizam um valor médio de utilização da CPU e de produtividade e minimizam um valor médio para os tempos de processamento, espera e resposta.

## Algoritmos de Alocação

- Primeiro a chegar, primeiro a ser servido.
- Menor primeiro.
- Alocação por prioridade.
- Alocação circular.
- Alocação com várias filas.
  - Divide a fila de processos prontos em vários níveis, constituindo filas separadas. Os processos entram em uma determinada fila – geralmente de acordo com alguma prioridade, como tamanho do espaço em memória necessária para execução do processo, prioridade do processo ou tipo do processo, e permanecem nessa fila até serem selecionados para execução. Cada fila usa um determinado algoritmo para seleção de processos. Por exemplo, podemos ter filas separadas para processos interativos e não-interativos.
  - Consideremos um exemplo de um algoritmo de alocação da CPU com cinco níveis (filas).

Prioridade mais alta




Prioridade mais baixa

**Alocação com várias filas**

- Cada fila tem prioridade absoluta sobre as filas de menor prioridade. Por exemplo, nenhum processo na fila de processos para processamento em lotes poderia ser executado a menos que as filas de processos de sistemas, de processos interativa e de processos de edição interativa estivessem todas vazias. Se um processo de edição interativa entrasse na fila de processos prontos enquanto um outro processo estivesse sendo processado em um sistema de lotes, a execução desse último seria interrompida.
- Outra possibilidade é repartir o tempo entre as filas.

- Alocação com várias filas e transferências entre filas.

  - Normalmente, em um algoritmo de alocação da CPU que usa várias filas, os processos entram em uma determinada fila e ai permanecem. Não ocorre nenhuma transferência de processos entre as filas.
  - A alocação com várias filas e transferências entre as filas permite que os processos sejam transferidos entre as diversas filas. A idéia é separar os processos com características diferentes de uso da CPU. Se um processo usa a CPU durante um tempo longo demais, ele é transferido para uma fila de prioridade mais baixa. Esse esquema mantém nas filas de prioridades mais altas os processos interativos e os processos que dependem mais de E/S. De maneira similar, um processo que fique muito tempo em uma fila de prioridade mais baixa pode ser transferido para uma fila de prioridade mais alta. Essa transferência evita que os processos sejam abandonados, sem entrar em execução.
  - Um escalonador com várias filas e transferências entre as filas é definido pelos seguintes parâmetros
    - Número de filas.
    - Algoritmo de alocação da CPU usado para cada fila.
    - Método usado para determinar quando transferir um processo para uma fila de prioridade mais alta.
    - Método usado para determinar quando transferir um processo para uma fila de prioridade mais baixa.
    - Método usado para determinar em qual fila um processo deve ser colocado, quando precisar usar a CPU.
  - O mecanismo de alocação baseado em várias filas, com possibilidade de transferências de processos entre as filas, é o mais geral dentre os mecanismos de alocação da CPU.

## Resumo

A tarefa de alocação da CPU consiste em selecionar um processo da fila de processos prontos e alocar a CPU a esse processo. O programa que realiza essa tarefa é chamado de escalonador.

- O algoritmo de alocação PCPS (Primeiro a chegar, primeiro a ser servido) é o mais simples de alocação da CPU, mas um processo que poderia ser executado rapidamente pode, com esse algoritmo, ter que esperar por processos que gastam muito tempo para ser processados. O algoritmo de alocação MP (menor primeiro) é comprovadamente ótimo, para o qual o tempo médio de espera é mínimo. Sua implementação apresenta a dificuldade de necessitar de uma previsão da duração da próxima fase de uso da CPU. O algoritmo MP é um caso especial do algoritmo geral de alocação por prioridade, que simplesmente aloca a CPU ao processo de maior prioridade. A alocação por prioridade e MP podem ambas provocar o abandono de processos, que pode ser evitado pelo uso da técnica de envelhecimento de processos.
- Alocação Circular (AC) é mais apropriada para um sistema de tempo compartilhado (interativo). Nossa abordagem; a CPU é alocada, por 7 unidades de tempo, ao primeiro processo na fila de processos prontos, onde q é um quantum de tempo. Depois das q unidades de tempo, se o processo não tiver liberado a CPU, ele será interrompido e colocado no final da fila de processos prontos. O maior problema é a seleção do valor para o quantum de tempo. Se ele for muito grande, AC degenera para PCPS; se for muito pequeno, o tempo gasto em mudanças de contexto torna-se relativamente grande e assim o tempo de uso da CPU pelo sistema operacional torna-se excessiva.
- O algoritmo PCPS é não-preemptivo; o algoritmo AC é preemptivo. O algoritmo MP e os algoritmos de alocação por prioridade podem ser tanto preemptivos quanto não-preemptivos.
- Algoritmos de alocação com várias filas permitem que algoritmos diferentes sejam usados para várias classes de processos. O mais comum é uma fila de processos interativos, para o qual o algoritmo AC é usado, e uma fila de processos não-interativos, para a qual o algoritmo PCPS é usado. Uma extensão desses algoritmos permite transferências de processos de uma fila para outra.
- A grande variedade de algoritmos de alocação disponíveis demanda que tenhamos métodos para selecionar um deles. Métodos analíticos usam análises matemáticas para determinar o desempenho de um algoritmo. Métodos de simulação determinam o desempenho imitando o algoritmo de alocação em uma amostra representativa de processos e calculando o desempenho resultante.