

PCS 3446 - Sistemas Operacionais

Prof. João José Neto

AULA 03

Administração de memória (2)

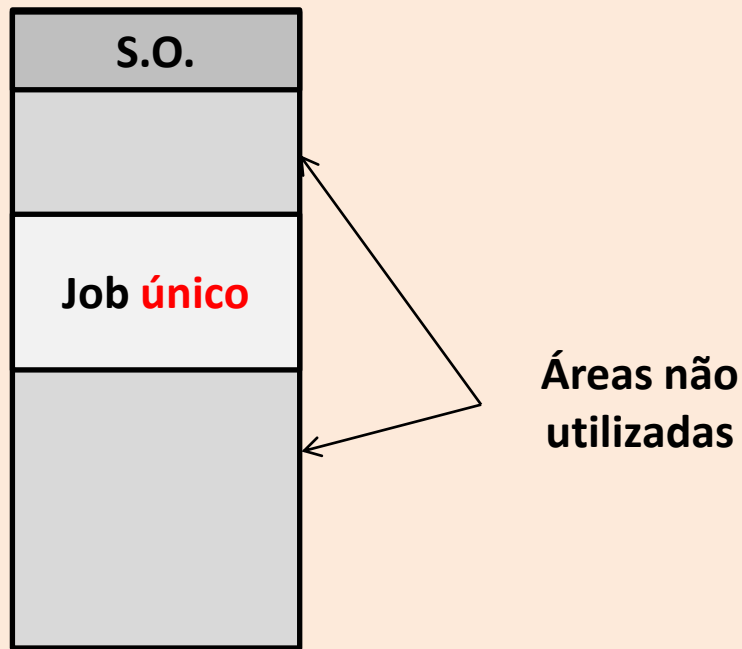
**Memória física – Multiprogramação –
Alocação Particionada: Simples e Relocável**

Introdução

- Nesta aula vamos estudar o esquema de administração de memória conhecido como “**Particionamento**”.
- É técnica diretamente inspirada na alocação contígua simples, tendo porém como objetivo manter na memória **vários programas/dados ao mesmo tempo**.
- Isso permite que programas chegados durante o processamento de um outro possam ir sendo **alocados previamente** na memória enquanto esperam sua vez para serem processados.
- Propicia também um esquema de **disponibilizar overlays**, utilizando memória particionada, no qual os *overlays* podem ser carregados em partições vazias.
- Viabiliza ainda adotar a **técnica da multiprogramação**, que estudaremos mais adiante.

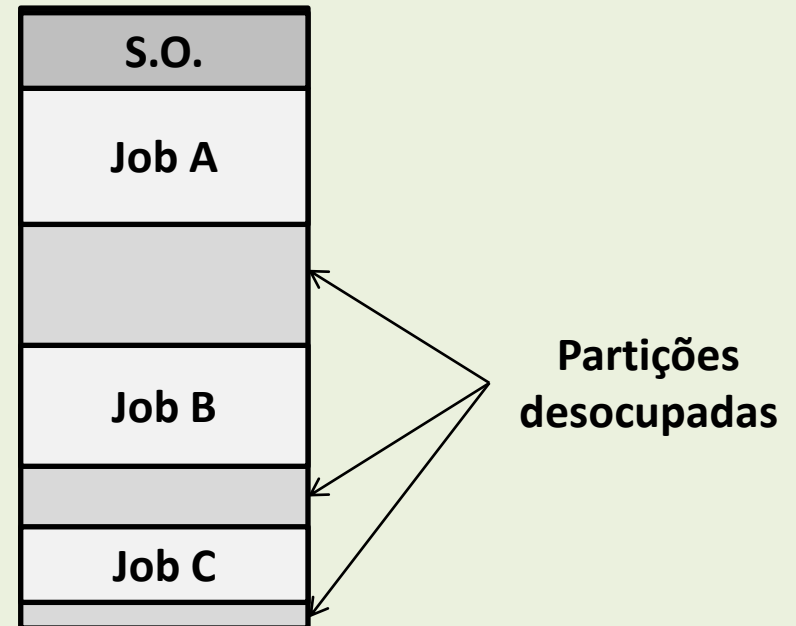
Particionamento de memória

Memória física
em alocação contígua **simples**



Sem divisões

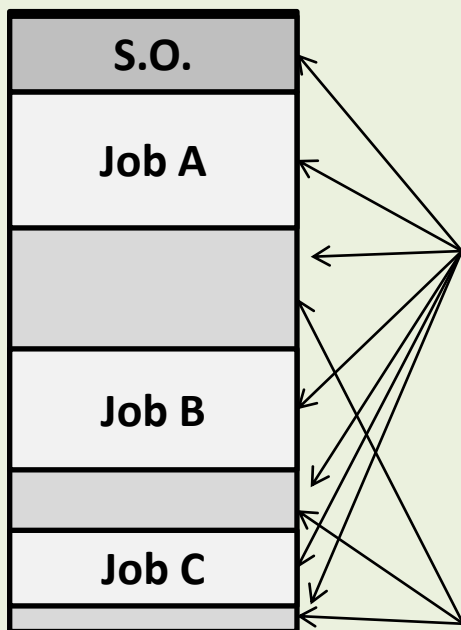
Memória física
em alocação **particionada**



Uma **partição**
para cada job

Particionamento estático e dinâmico

Particionamento **estático**

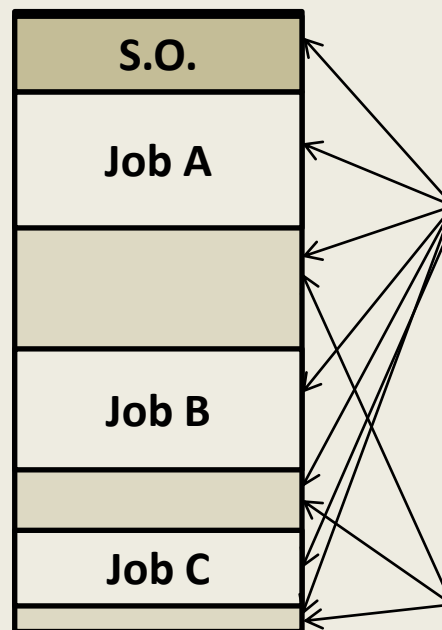


Número (identificação), tamanho e endereços físicos das partições são **invariáveis** durante sua permanência no sistema.

Partições desocupadas

Uma partição para cada job

Particionamento **dinâmico**



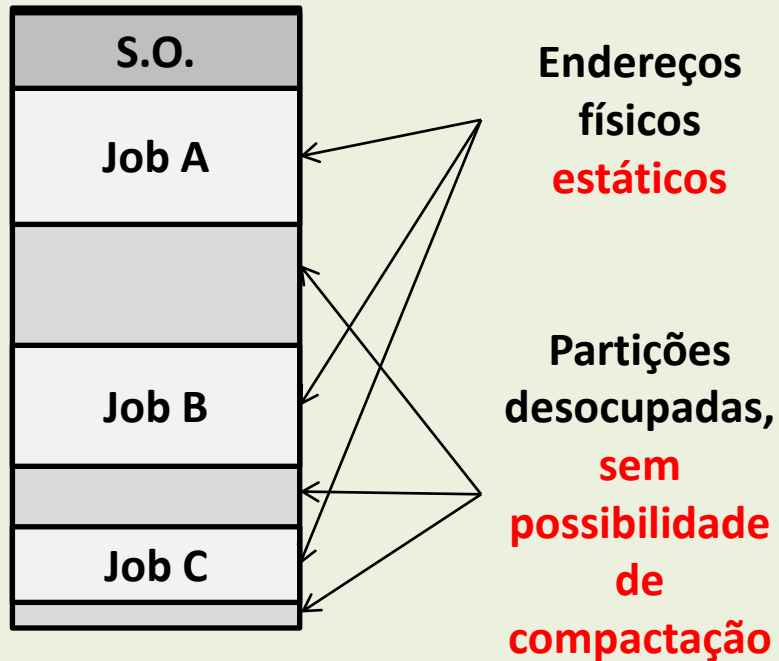
Número (identificação), tamanho e endereços físicos das partições podem ser **alterados a qualquer momento.**

Partições desocupadas

Uma partição para cada job

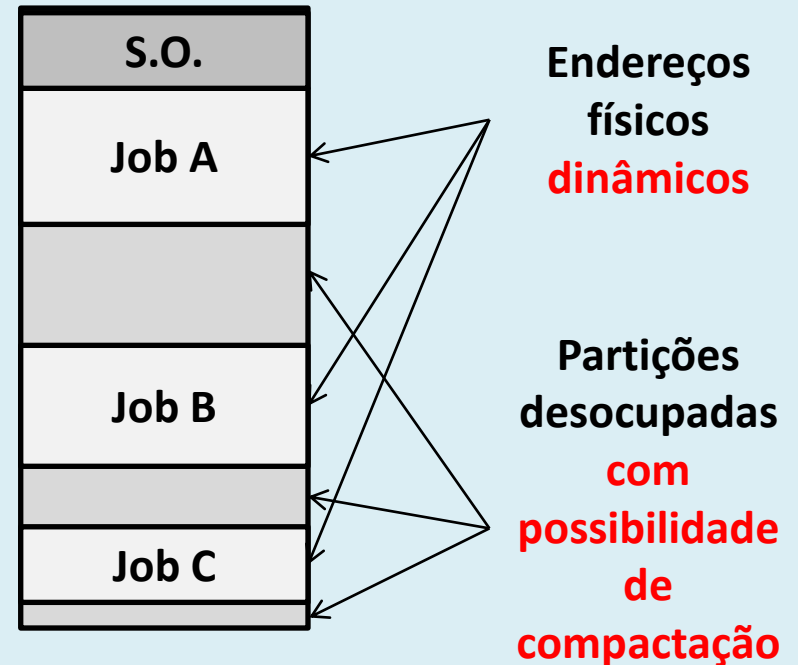
Particionamento simples e relocável

Memória física
em alocação **particionada simples**



Uma partição
para cada job

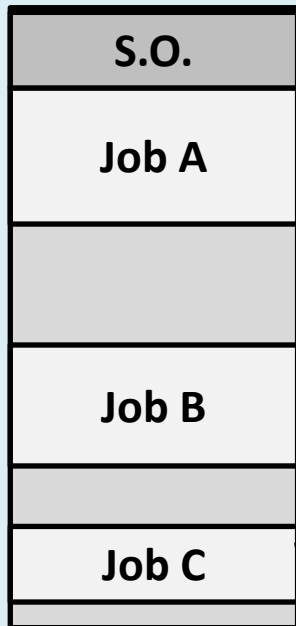
Memória física
em alocação **particionada relocável**



Uma partição
para cada job

Relocação estática e dinâmica

Memória física em alocação
particionada, com **relocação estática**

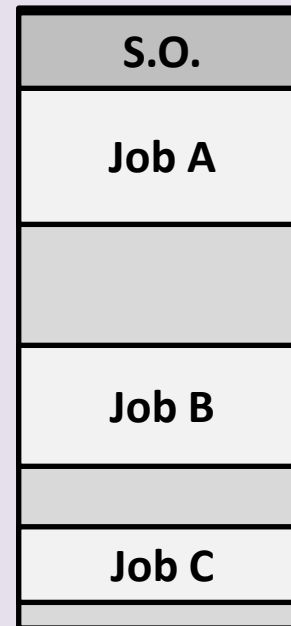


Endereços
físicos **não se
alteram** com
o job em
execução

Partições
desocupadas
com
possibilidade
de
compactação
dinâmica
(garbage
collection)

Uma partição
para cada job

Memória física em alocação
particionada, com **relocação dinâmica**



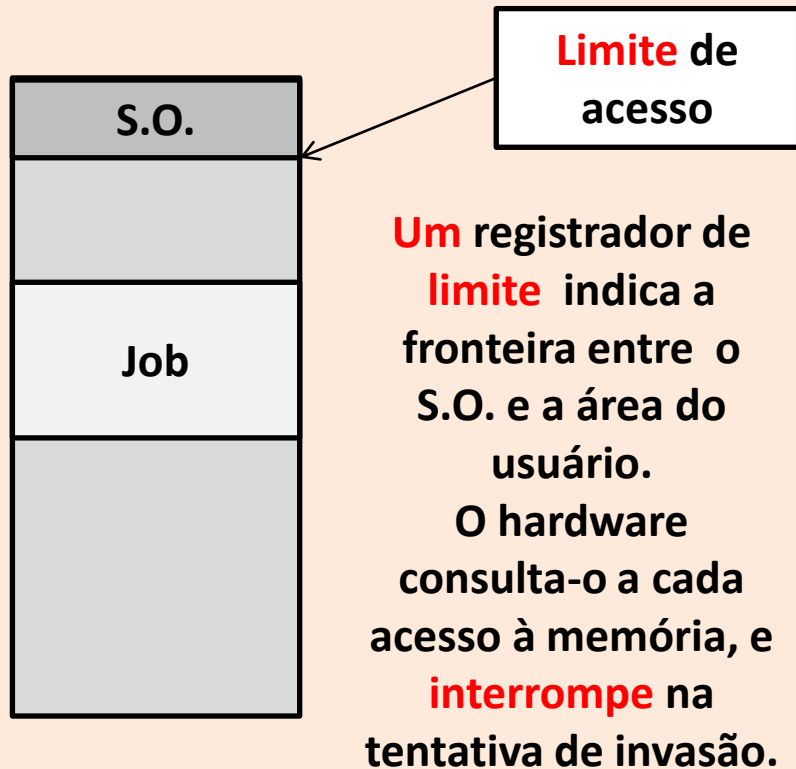
Endereços
físicos **podem
alterar-se**
com o job em
execução

Partições
desocupadas
com
possibilidade
de
compactação
dinâmica
(garbage
collection)

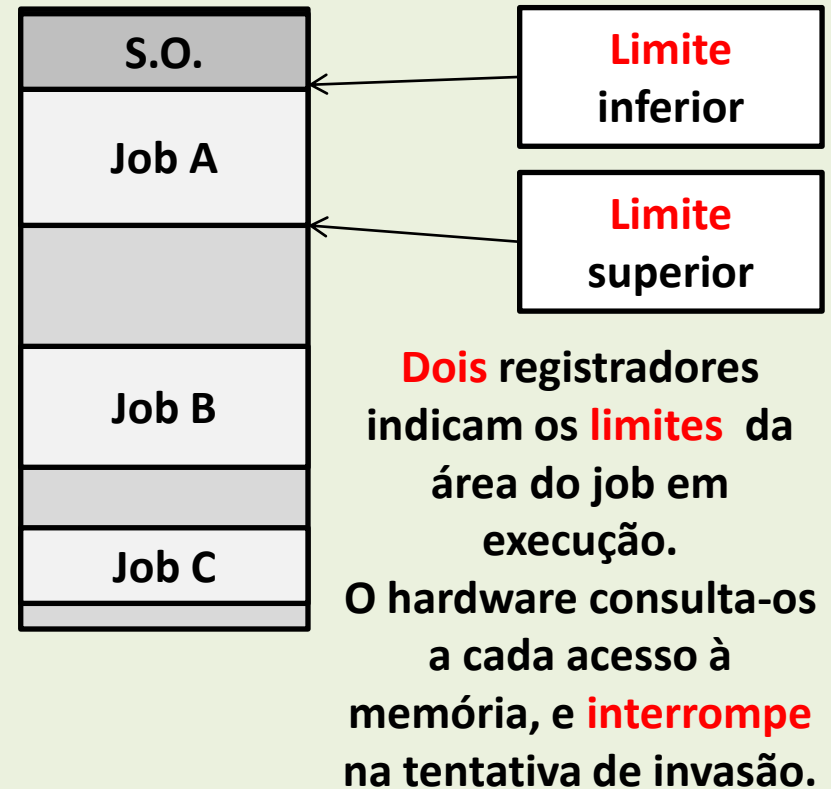
Uma partição
para cada job

Proteção de memória

a) Entre sistema operacional e usuário

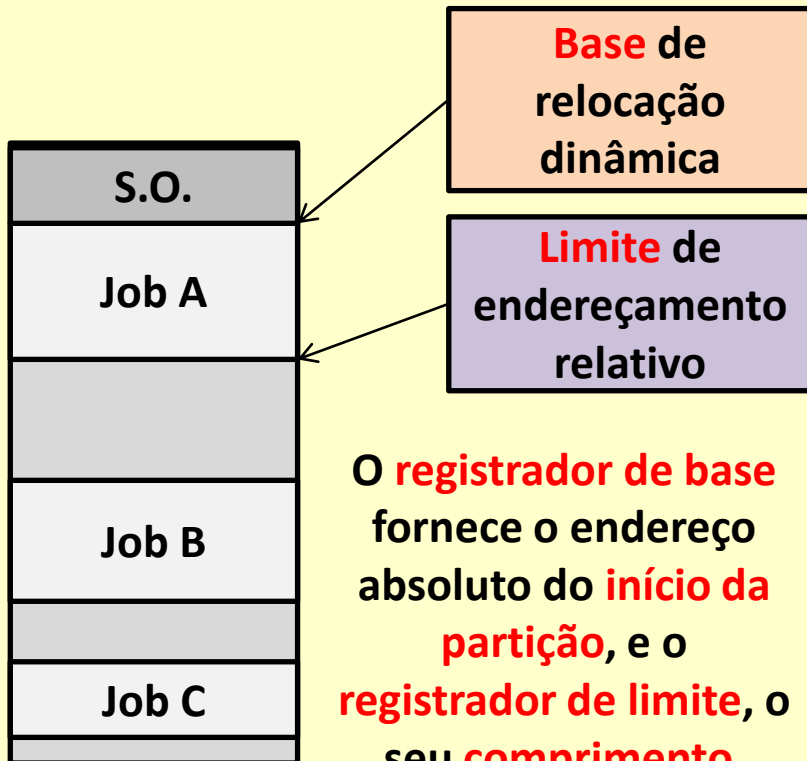


b) Entre jobs e/ou sistema operacional



Proteção de memória com relocação

c) Acrescentando **relocação dinâmica**:



O **registrador de base** fornece o endereço absoluto do **início da partição**, e o **registrador de limite**, o seu **comprimento**, permitindo **proteção e relocação** simultâneos.

Este esquema favorece a implementação da **multiprogramação**, pois simplifica o chaveamento de jobs no uso do processador (além de salvar/restaurar o status do programa, para passar a executar outro job, basta substituir o par base-limite). Todos os **endereços** referenciados no programa devem ser **relativos à base**, e portanto o primeiro endereço relativo de cada partição deve ser zero. O **hardware**, naturalmente, deve compensar as referências relativas adicionando a cada endereço relativo de memória o conteúdo do registrador de base de relocação dinâmica. Isto permite que qualquer job seja **movido para qualquer endereço**, sem alterações.

MULTIPROGRAMAÇÃO NA ADMINISTRAÇÃO DE MEMÓRIA

Conceito de Multiprogramação

- **Multiprogramação** é uma técnica de **virtualização de processamento**, que permite a vários programas operarem simultaneamente, mesmo sendo mais numerosos que os processadores disponíveis.
- Sua concepção é bastante antiga, e foi criada para uso nos computadores pioneiros, que dispunham de um único processador e uma quantidade modesta de memória física.
- Alocando simultaneamente partições diferentes para diferentes programas, e executando um **rodízio dos programas** para uso do processador por intervalos de tempo conhecidos, foi possível criar a ilusão de que todos esses programas dispunham de recursos de processamento para sua execução. Voltaremos a esse assunto posteriormente.
- O **particionamento** é uma forma de administração de memória que viabiliza a técnica da **multiprogramação** a um **custo muito baixo**.

Funcionamento da Multiprogramação

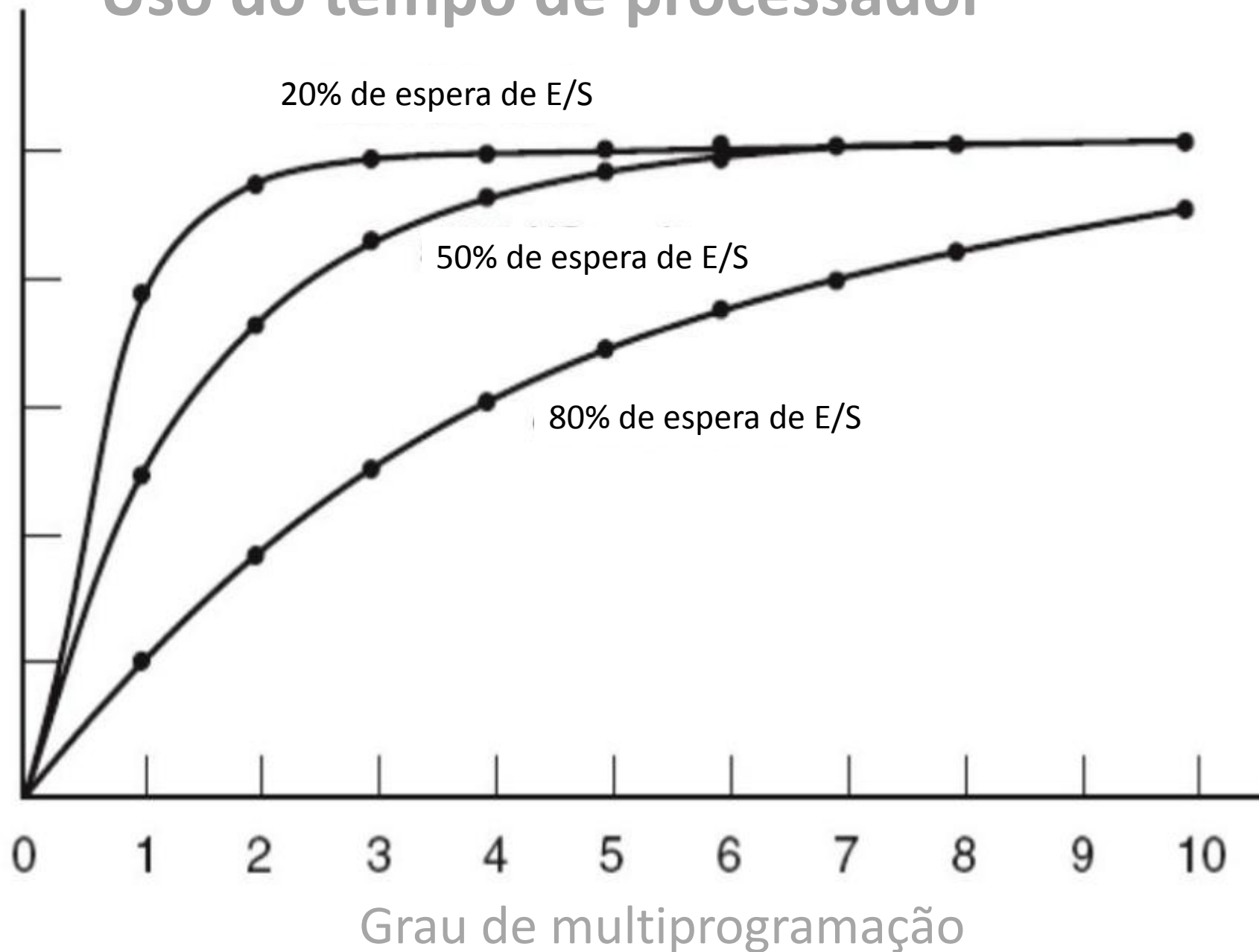
- É necessário que o processador disponha de um **relógio** que possa **interromper** o processamento a intervalos regulares.
- Alocadas partições de memória entre os N programas ativos em certo instante, e estando um deles a ser executado, os demais permanecem em **fila de espera**, aguardando sua vez.
- Após ser executado pelo intervalo de um pequeno ***quantum*** de tempo Δt , medido pelo relógio, este **interrompe** o processamento e o **controle é passado para o sistema** operacional.
- O sistema então **aloca o processador ao primeiro** programa da fila de espera, e **põe aquele que foi interrompido no final da fila**, onde deverá aguardar a sua vez para continuar a sua execução.
- Dessa forma, havendo N programas em operação, a cada ***quantum*** todos avançarão (no máximo Δt) em seu uso do processador, e aos poucos, todos irão progredindo, até que venham a terminar seu trabalho, saindo então do sistema. Isso cria a **ilusão de haver N processadores virtuais** lentos, em vez de um único real, rápido.

Multiprogramação e desempenho

- Nos próximos três slides antecipamos alguns dados sobre a influência da multiprogramação no desempenho do computador:
 - Curvas mostrando a variação do aproveitamento processador em função do grau de multiprogramação
 - Fórmula da taxa de espera de entrada/saída no sistema em função do grau de multiprogramação
 - Uma tabela comparativa de taxas de espera de entrada/saída, para alguns valores do grau de multiprogramação

Uso do tempo de processador

Porcentagem de uso do processador



Taxa de espera de E/S

Sem multiprogramação:

$$w = \frac{\textit{tempo total de espera de E/S}}{\textit{tempo total de processamento} + \textit{tempo total de espera de E/S}}$$

Com grau n de multiprogramação:

$$w' = \frac{\left(\frac{w}{1-w}\right)^n}{n! * \sum_{i=0}^n \frac{\left(\frac{w}{1-w}\right)^i}{i!}}$$

Taxa de espera de um job em multiprogramação

Valores de w' para $w=65\%$:

$n=2, w'=37,6\%;$

$n=3, w'=18,9\%;$

$n=4, w'=8,1\%;$

$n=5, w'=2,9\%;$

$n=6, w'=0,9\%;$

$n=7, w'=0,2\%;$

$n=8, w'=0,1\%.$

POLÍTICAS NA ADMINISTRAÇÃO DE MEMÓRIA PARTICIONADA

Políticas de alocação de memória

- Em esquemas multiprogramados, é aleatório o momento em que os jobs ocupam ou liberam partições de memória, de modo que se pode considerar esse processo como um fenômeno estocástico.
- Assim, não há como prever nem como determinar a forma ótima como se devem efetuar as alocações, relocações e compactações de memória, por isso são geralmente adotadas heurísticas, procurando obter boas decisões em uma parcela aceitável de casos.
- Destacam-se neste sentido os algoritmos clássicos de alocação ***First-fit, Best-fit, Worst-fit*** (há outros).

First-Fit

- Consiste em percorrer linearmente as partições vazias a partir dos endereços mais baixos da memória, em busca da primeira partição em que caiba o programa que está para ser alocado.
- O algoritmo *first-fit* é de todos o mais simples, e costuma dar resultados bastante satisfatórios.
- A tendência estatística deste algoritmo é de se acumularem no final da memória partições vazias grandes, o que evita o espalhamento de fragmentos (partições pequenas) pela memória.

Best-Fit

- Consiste em varrer todas as partições vazias a partir do início da memória, em busca de uma na qual o programa que está para ser alocado caiba da forma mais perfeita, resultando portanto o menor resíduo (fragmento) possível.
- O algoritmo *best-fit* também é simples, mas pelo fato de buscar resultados ótimos, frequentemente despreza boas opções, menos onerosas.
- A tendência estatística desta política é a de espalhar pela memória partições vazias pequenas demais, o que tende a fragmentar a memória e dificultar o aparecimento de áreas vazias úteis (grandes).

Worst-Fit

- Consiste em buscar, entre as partições vazias, a partir do início da memória, aquela na qual a alocação do programa deixe como resíduo o fragmento mais útil (maior) possível.
- O algoritmo *worst-fit* procura uma partição cuja alocação fragmente a memória o menos possível.
- Como isso tende a consumir partições grandes, sua tendência estatística é a de que, aos poucos, na memória se espalhem partições vazias cada vez menores, levando à fragmentação e dificultando o aparecimento de áreas vazias grandes.

Fragmentação

- Característico da alocação particionada relocável, o fenômeno da fragmentação se instala sempre que, ainda que haja suficiente área livre de memória, mesmo assim não seja possível alocar tal área a um job solicitante, posto que esse espaço não forma uma região contígua da memória física.
- Para o sistema operacional, é obviamente indesejável que ocorra a fragmentação, pois isso acarreta desperdício de memória.

Garbage Collection

- Quando ocorre o fenômeno da fragmentação, aplicam-se algoritmos que procuram remover da memória os fragmentos, disponibilizando a área útil para ser alocada a um job solicitante.
- Os algoritmos de ***Garbage Collection*** (coleta de lixo, ou de fragmentos), também conhecidos como algoritmos de **defragmentação**, ou de **compactação da memória**, consistem na fusão, em uma única grande partição física vazia, de todas as partições vazias (especialmente as pequenas) que se encontrem espalhadas pela memória.
- Familiarize-se com eles em bons livros de algoritmos.

FIM