



Unidade 4

Seção 2

# Sistemas Operacionais

## Webaula 2

### *Swapping: troca de processos*

Nesta webaula vamos apresentar os conceitos do *swapping*, mostrar como se dá a multiprogramação com partições variáveis e o gerenciamento de memória com mapa de bits, e ainda apresentar os algoritmos de troca de processos.



## Troca de processos

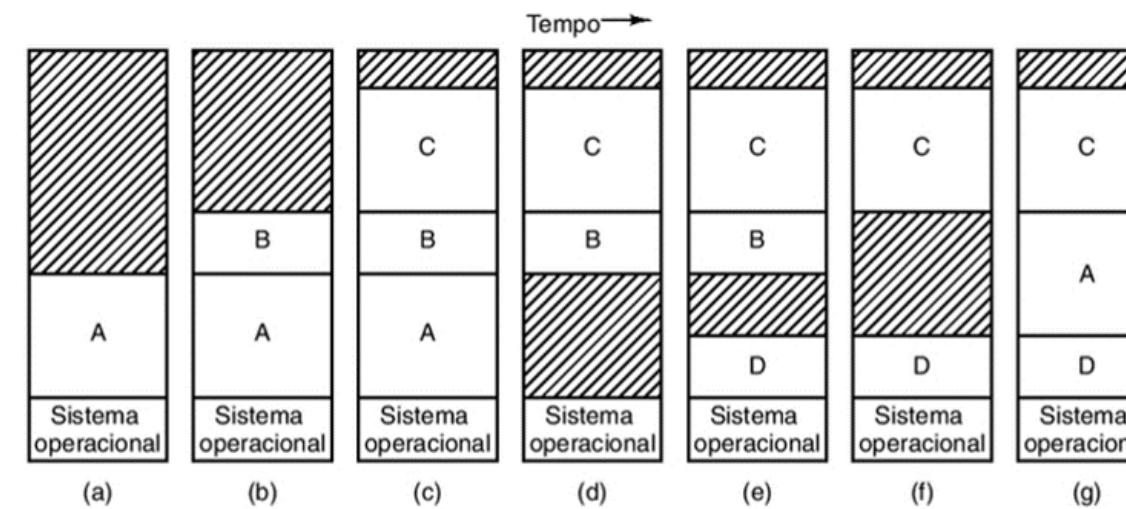
A troca de processos (*swapping*) é realizado quando não existe memória principal suficiente para executar todos os programas do computador ao mesmo tempo.

Na troca de processos (*swapping*) um programa é totalmente carregado em memória e executado por um tempo definido, enquanto os demais programas aguardam, em disco, sua vez de executar.





A troca de processos (*swapping*) traz totalmente cada processo para a memória, o executa por algum tempo e o retorna para o disco. A figura apresenta como se dá o funcionamento da troca de processos na memória principal.



Fonte: Tanenbaum (2003, p. 145).

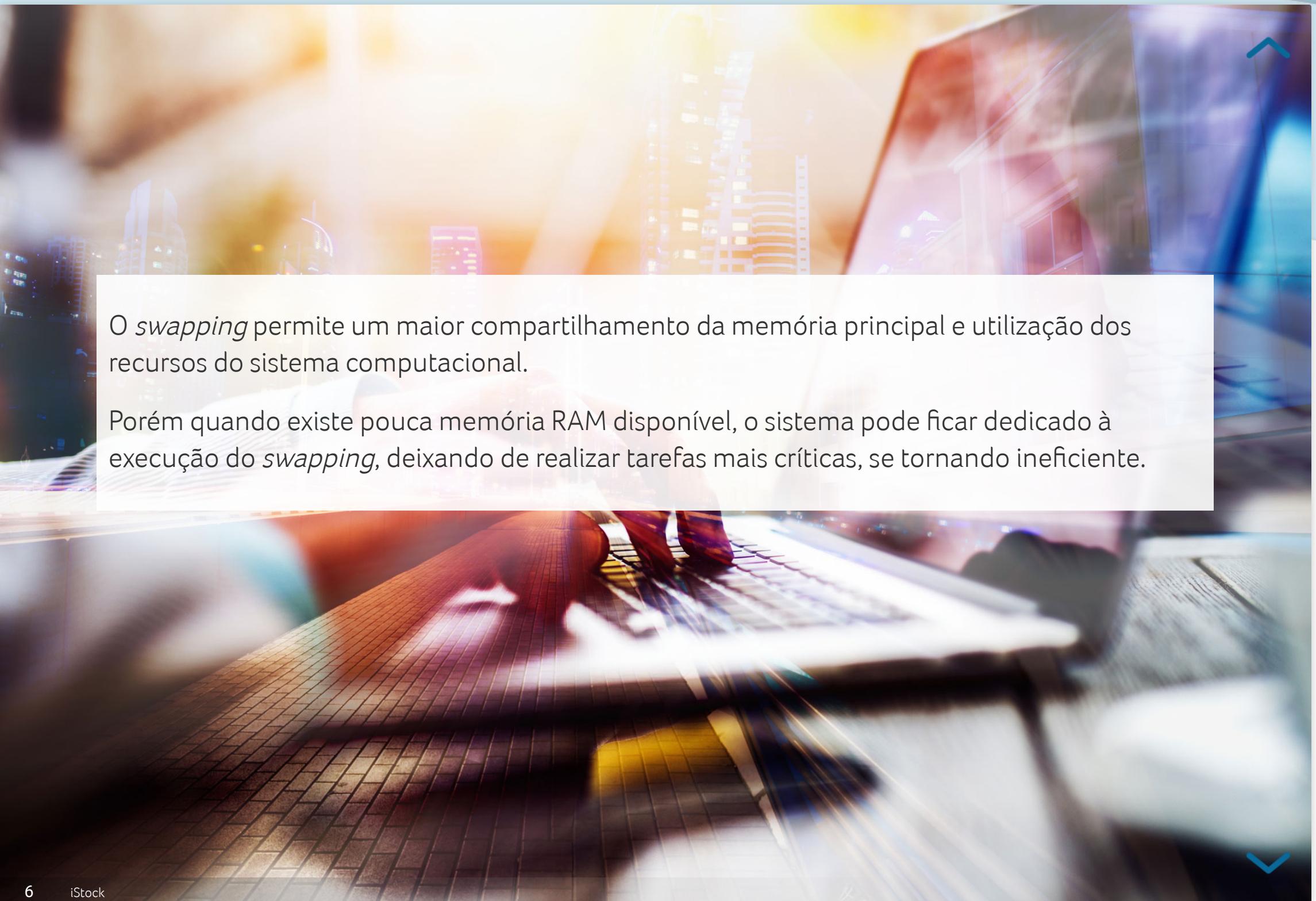




Vamos exemplificar o conceito de troca de processos (*swapping*). Explore a galeria.

Um computador possui uma memória de 512MB e tem 4 processos para serem executados com os tamanhos 481MB, 508MB, 380MB e 369MB, respectivamente.





O *swapping* permite um maior compartilhamento da memória principal e utilização dos recursos do sistema computacional.

Porém quando existe pouca memória RAM disponível, o sistema pode ficar dedicado à execução do *swapping*, deixando de realizar tarefas mais críticas, se tornando ineficiente.



## Multiprogramação com partições variáveis

A alocação particionada variável, consiste em ajustar dinamicamente o tamanho das partições de memória quando os processos chegam para serem executados. Ou seja, cada processo utiliza um espaço necessário para executar, não acontecendo a fragmentação interna.

A vantagem das partições variáveis é a flexibilidade por não estar preso a um número fixo de partições, melhorando a utilização da memória, porém impactando o gerenciamento das trocas de processos e na alocação e liberação da memória.





Quando processos precisam consumir mais memória durante o processamento, é necessário alocar memória dinamicamente. Existem dois métodos de gerenciamento de memória com alocação dinâmica:

Com Mapa de Bits.

Com Listas Encadeadas.

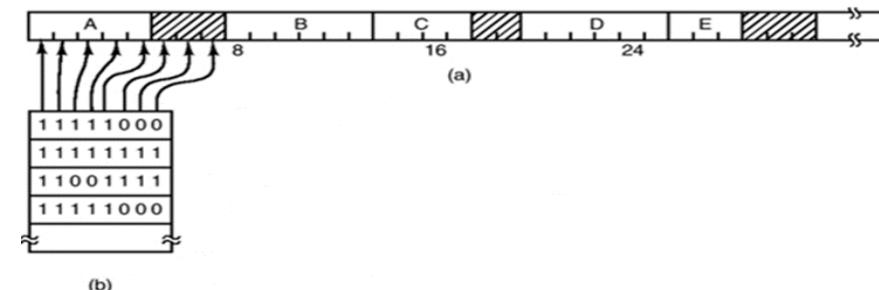




## Gerenciamento de memória com mapa de bits

Neste método a memória é dividida em unidades de alocação, a qual é associada a um bit no mapa de bits. Se o valor do bit for 1, indica que a unidade está ocupada e se o bit for 0 ela está livre. A figura a seguir apresenta o mapa de bits e a memória dividida em unidades de alocação. Clique na figura a seguir.

Mapa de bits e uma parte da memória



Fonte: Tanenbaum (2003, p. 146).

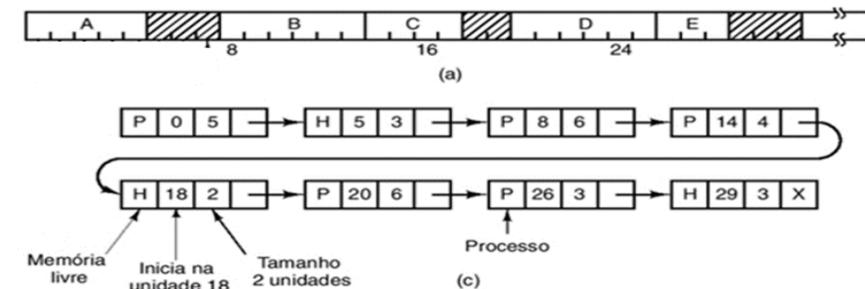


## Gerenciamento de memória com listas encadeadas

Este método consiste em manter uma lista encadeada de segmentos alocados e livres na memória.

A figura (a) mostra uma parte da memória com cinco segmentos alocados a processos (A, B, C, D e E) e três segmentos de memória livre (espaços vazios). A figura (c) apresenta a lista encadeada e a memória dividida em unidades de alocação. Clique na figura a seguir.

Lista encadeada e uma parte da memória



Fonte: Tanenbaum (2003, p. 146).



## Algoritmos de troca de processos

Para definir em qual área livre os processos serão executados por meio da lista encadeada são utilizados os algoritmos. Clique nas abas para conhecer os algoritmos de alocação de memória.

*First Fit*

*Next Fit*

*Best Fit*

*Worst Fit*

*Quick Fit*

*First Fit* (primeiro que couber): é o algoritmo mais simples e que consome menos recurso do sistema. O gerenciador de memória procura ao longo da lista por um segmento livre que seja suficientemente grande para esse processo.



É importante ressaltar que a estratégia a ser usada pelo gerenciador de memória para a escolha de qual algoritmo utilizar irá depender de fatores, como o tamanho dos processos executados no ambiente e das áreas livres disponíveis.







Bons estudos!

