Sistemas Operacionais

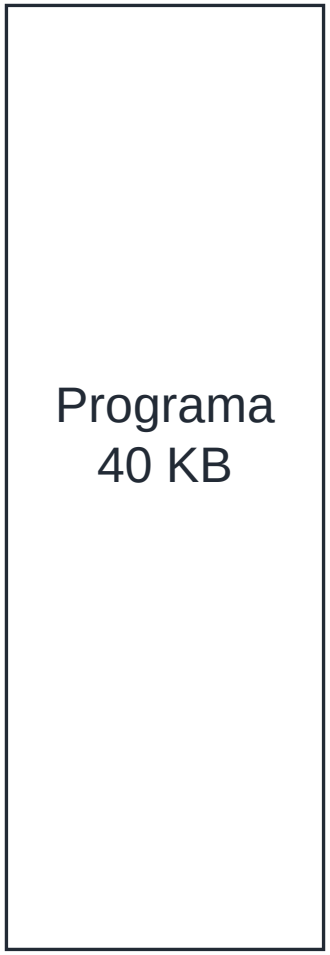
4º período

Professora: Michelle Hanne

PUC Minas

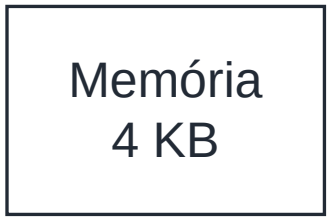
Gerência de Memória

Gerenciamento de Memória – Sem Abstração de Memória

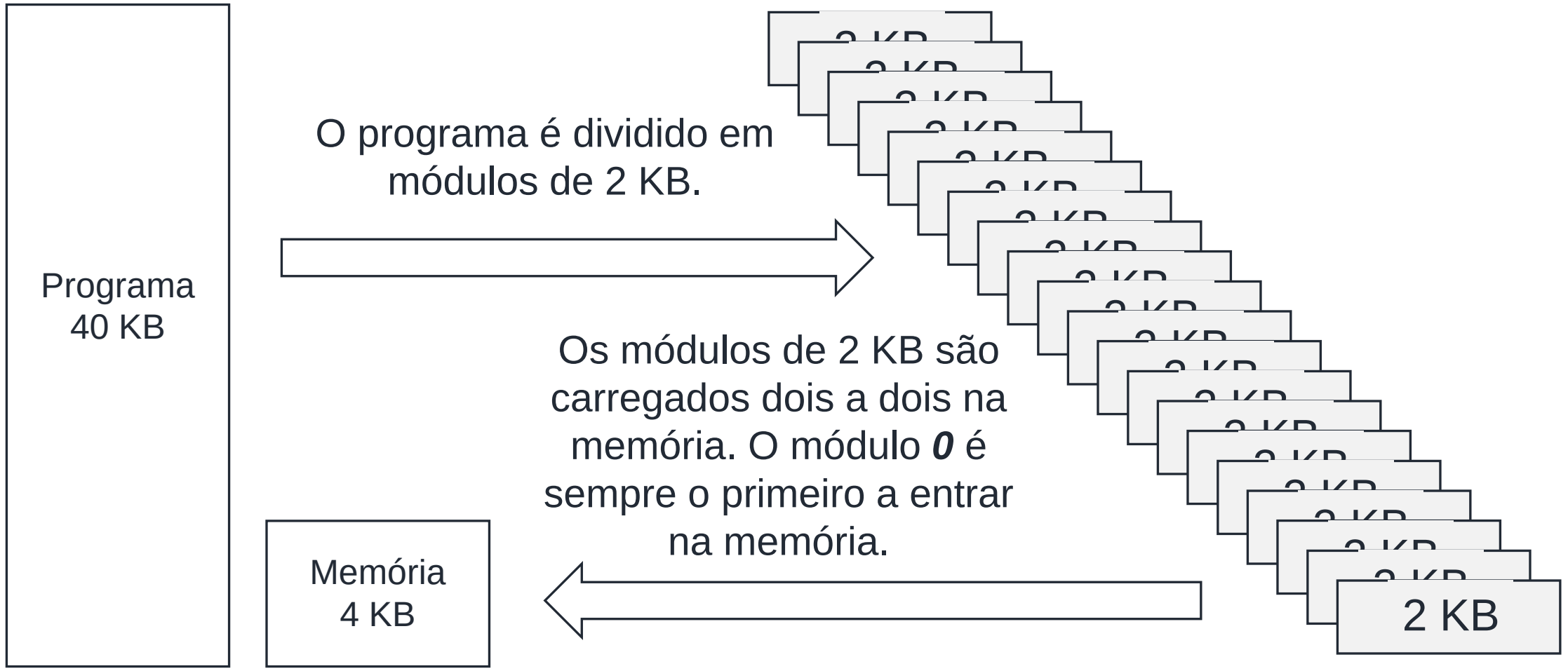


Desde o início da computação é comum a existência de programas maiores do que a memória disponível.

Uma solução utilizada com muita frequência era o ***overlay (módulos de sobreposição)***, que é a divisão do programa em subprogramas.



overlay (módulos de sobreposição)

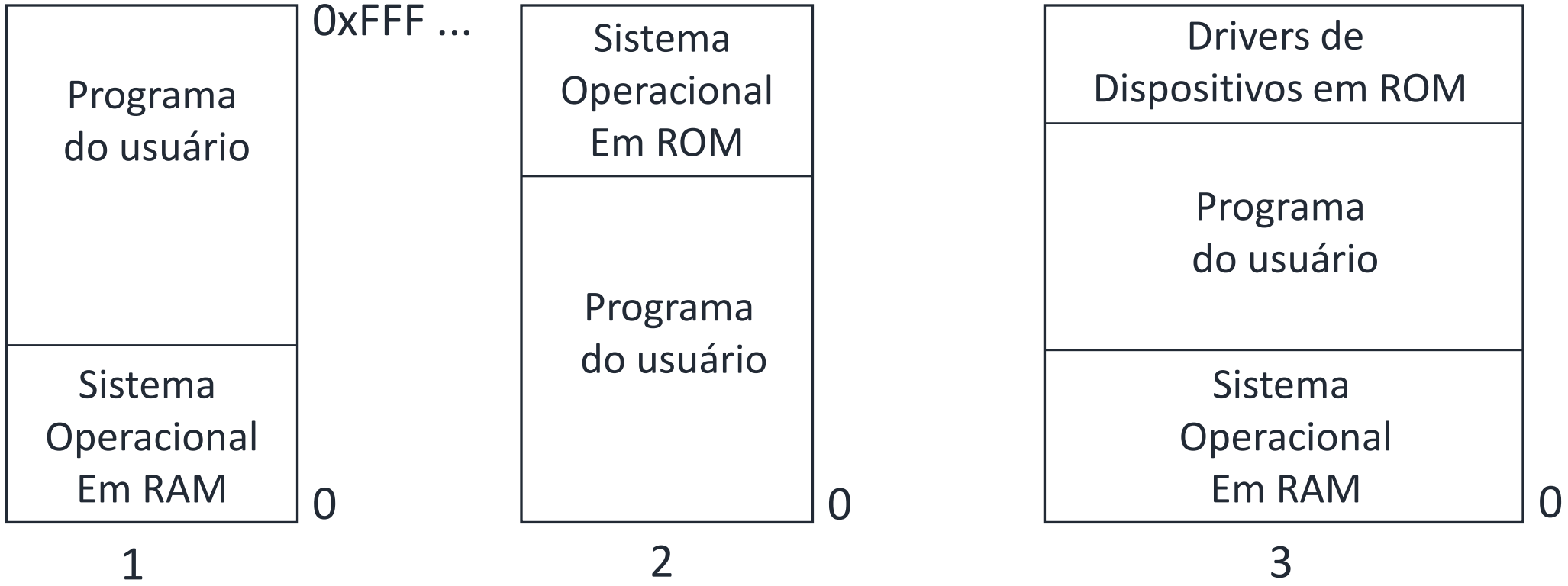


É de responsabilidade do programador fazer a divisão dos módulos e ordenar a sequência de execução dos módulos, o SO somente os carrega. Caso o programa seja transferido para uma máquina de arquitetura diferente (com menos memória), o programador deverá refazer a subdivisão do programa.

Gerenciamento de Memória – Sem Abstração de Memória

PUC Minas

Modelos simples de organizar a memória com um sistema operacional e um processo de usuários.

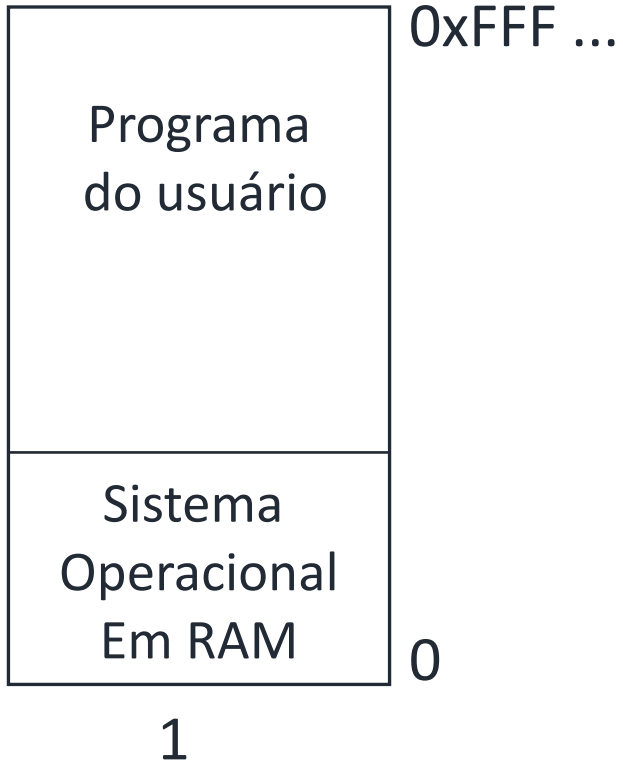


Obs: Existem outros modelos !!!!

Gerenciamento Básico de Memória

PUC Minas

Monoprogramação sem troca de processos ou paginação.

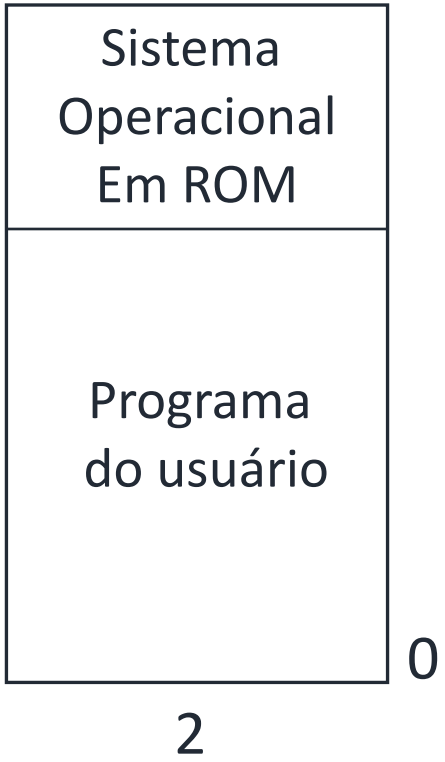


Monoprogramação - Quando os computadores processavam um único programa por vez.

O sistema operacional executava um único programa por vez, para trocar de programa, deveria salvar todo o esse contexto da memória, enviá-lo para um dispositivo de armazenamento secundário, para então carregar um outro programa.

Gerenciamento de Memória – Sem Abstração de Memória

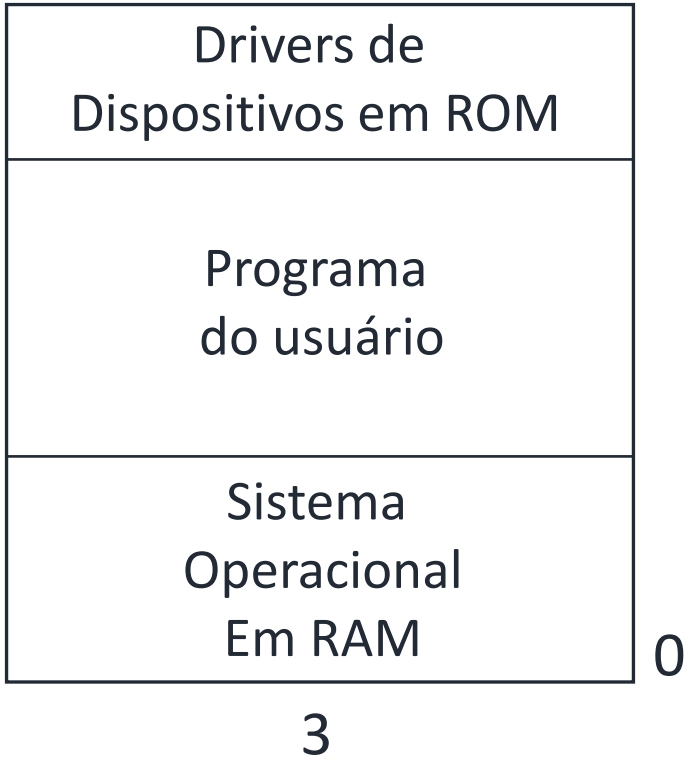
Monoprogramação sem troca de processos ou paginação.



O sistema operacional fica armazenado em uma memória ROM , o programa do usuário fica armazenado em na memória RAM.
Exemplo: Palms, agendas eletrônicas, dentre

(Tanenbaum, 2016)

Gerenciamento de Memória – Sem Abstração de Memória



Computadores antigos utilizando o MS-DOS.

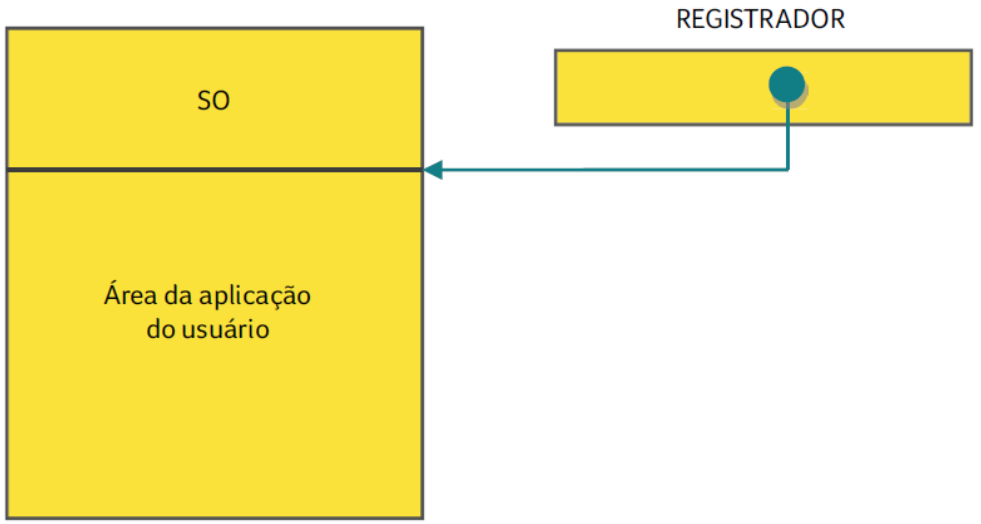
Não era possível utilizar os computadores como utilizamos hoje.

(Tanenbaum, 2016)

Gerenciamento de Memória – Sem Abstração de Memória

PUC Minas

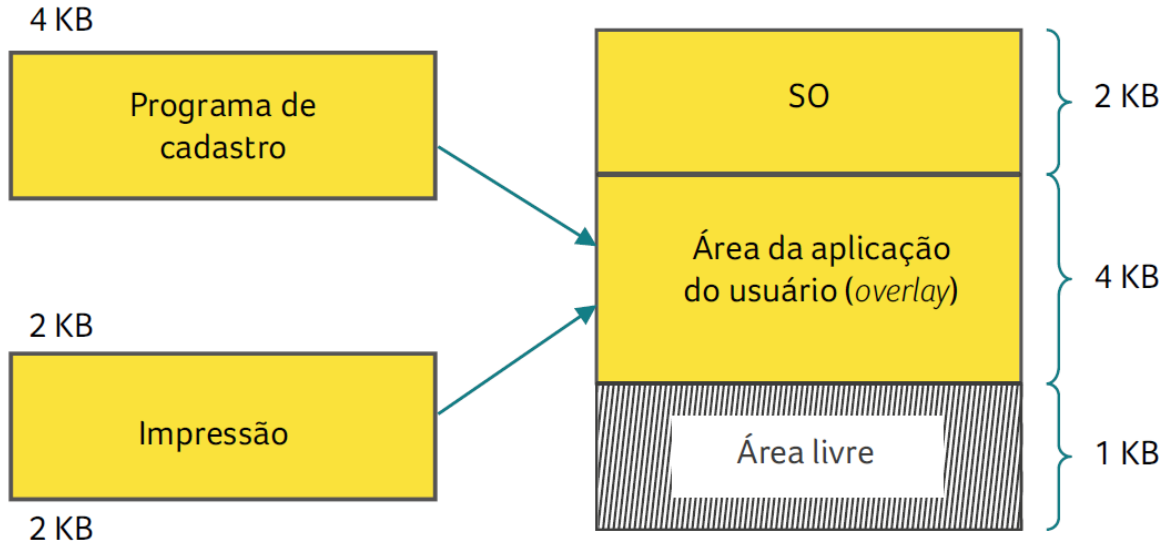
- No sistema **monoprogramável**, a responsabilidade de não ultrapassar **os limites de memória era do programador**.
- O **programador tinha total controle sobre toda a memória principal** e podia acessar qualquer posição de memória.
- Assim, qualquer **usuário poderia atacar o sistema operacional em virtude do acesso total a qualquer posição de memória.** Um usuário malicioso poderia destruir ou alterar qualquer item em memória, inclusive o próprio SO (sistema operacional).
- Na proteção desse possível ataque, **registradores foram implementados**. Esses registradores tinham como finalidade delimitar as áreas de memórias. Quando determinado programa do usuário fazia referência a um endereço na memória, o sistema verificava a disponibilidade dos endereços dentro dos seus limites.



Gerenciamento de Memória – Sem Abstração de Memória

PUC Minas

- **Técnica de Overlay:** Dividir os programas em módulos menores; dessa forma, parte do programa poderia ser executada independentemente de outra, uma forma mais eficiente de usar o espaço disponível em memória, conhecida como técnica de sobreposição (overlay).
- A área de *overlay* é compartilhada pelos programas de cadastro e impressão. O programador tem a função de definir essa área. Esse tamanho é estabelecido a partir do maior módulo implementado pelo programador



Multiprogramação

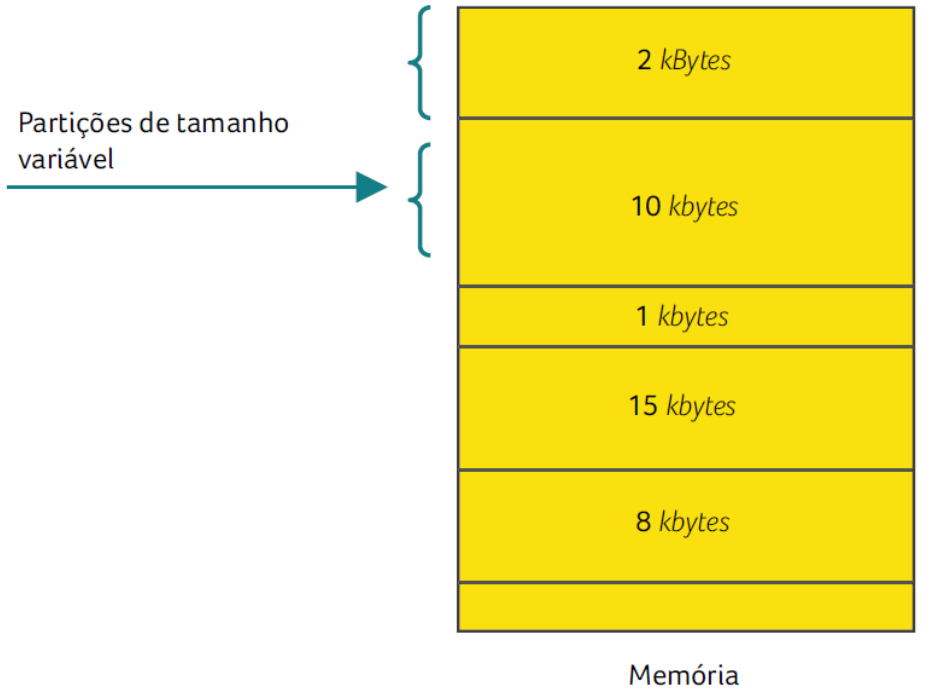
PUC Minas

No sistema multitarefa, a presença da interface gráfica determina uma facilidade de uso dos recursos do computador. Sistemas multitarefas são multiusuários e multiprogramáveis**. Diversas aplicações podem ser executadas ao mesmo tempo, levando a uma condição bem mais complexa do que os sistemas monoprogramáveis.**

Hoje, a maioria dos sistemas permite que múltiplos processos sejam executados de forma simultânea na CPU, o que resulta **numa condição de espera por processos que dependem que uma operação de entrada e saída seja finalizada.**

Esse procedimento demanda uma complexidade de utilização da CPU. Os aplicativos atuais são geralmente desenvolvidos em plataforma *web* e acessados por meio de navegadores.

O mecanismo mais comum de compartilhamento da memória em sistema de multiprogramação é dividir a memória em diversas áreas (partições), que podem ser fixas ou variáveis.



Multiprogramação com Partições Fixas de Memória

PUC Minas

Esse tipo de compartilhamento de área de memória foi idealizado nos primeiros sistemas monoprogramáveis.

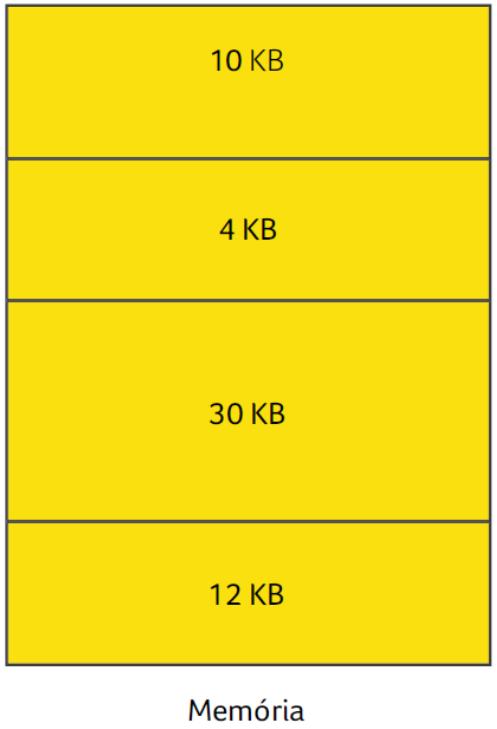
- Consiste em dividir a memória em área de tamanho fixo.

Essas áreas eram estabelecidas na inicialização do sistema operacional.

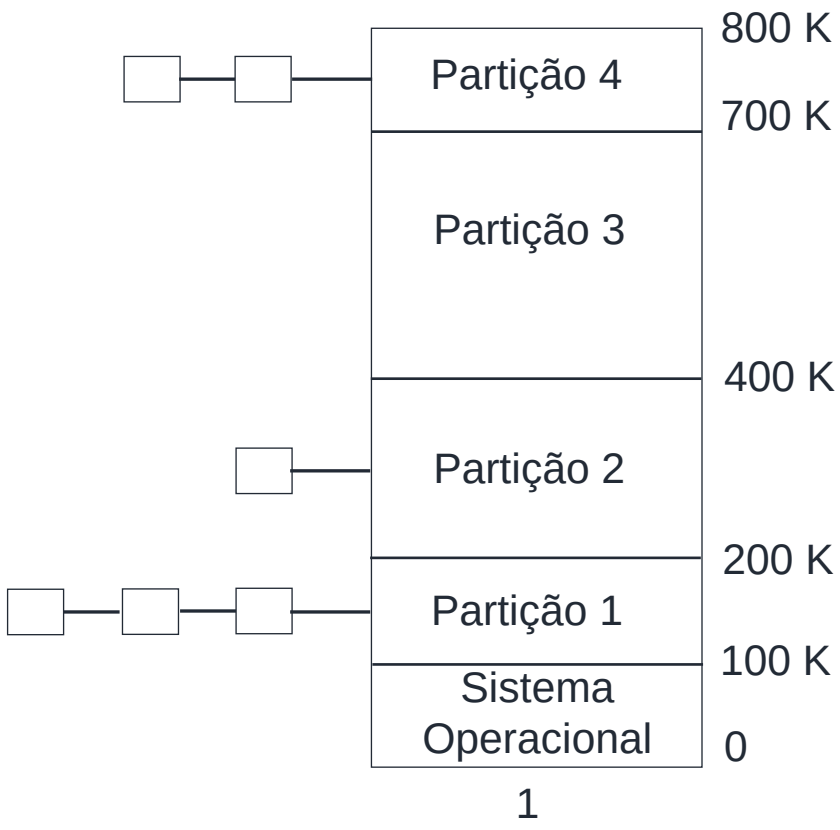
- Por exemplo, no **Microsoft Disk Operating System**, todo o programa seria utilizado em determinada partição fixa, de tamanho fixo. Esse sistema operacional ocupava 640 KB da área de memória RAM; então, esse teria que ser o tamanho mínimo da partição que deveria armazenar o MS-DOS.

O problema consistia em: sempre que a partição fosse alterada, o sistema deveria ser paralisado e reiniciado novamente.

- No início, os programas só podiam ser executados somente em uma partição, mesmo com outras partições disponíveis. Posteriormente, as partições passaram a ser realocáveis.



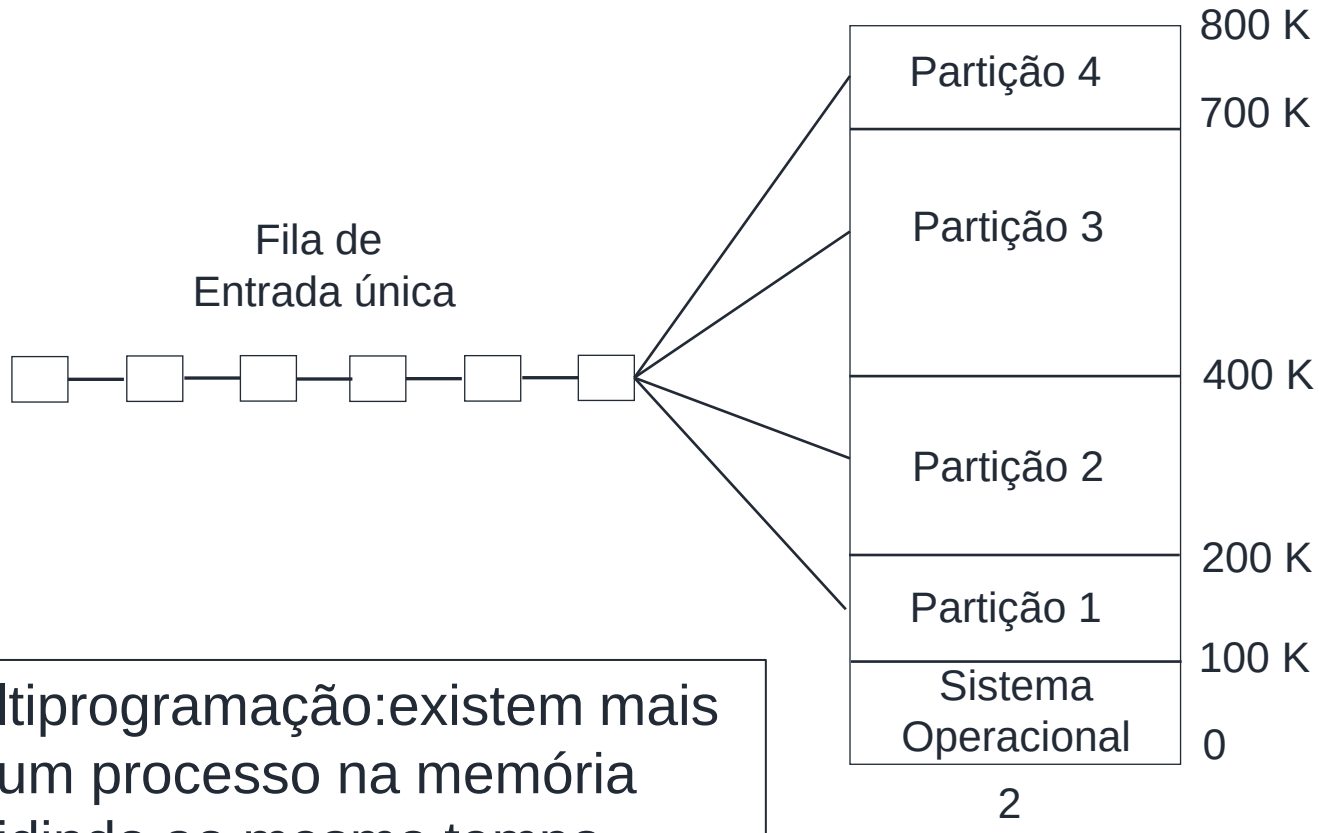
Multiprogramação com Partições Fixas de Memória



1

Filas de entrada separadas para cada partição.
Se um processo trava, prejudica os demais processos.

Multiprogramação:existem mais de um processo na memória residindo ao mesmo tempo.

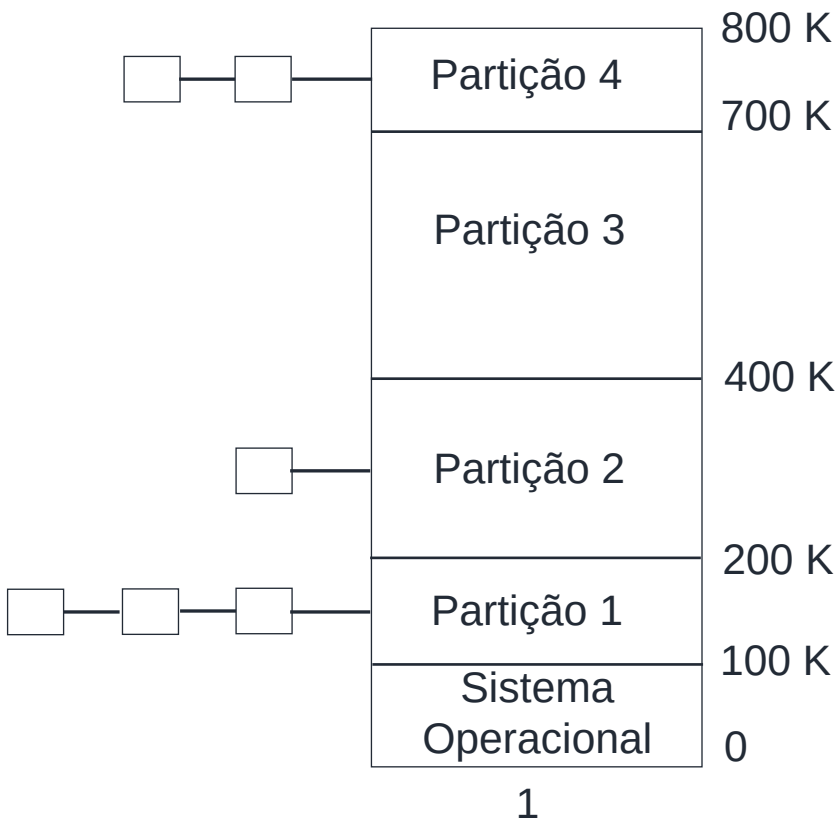


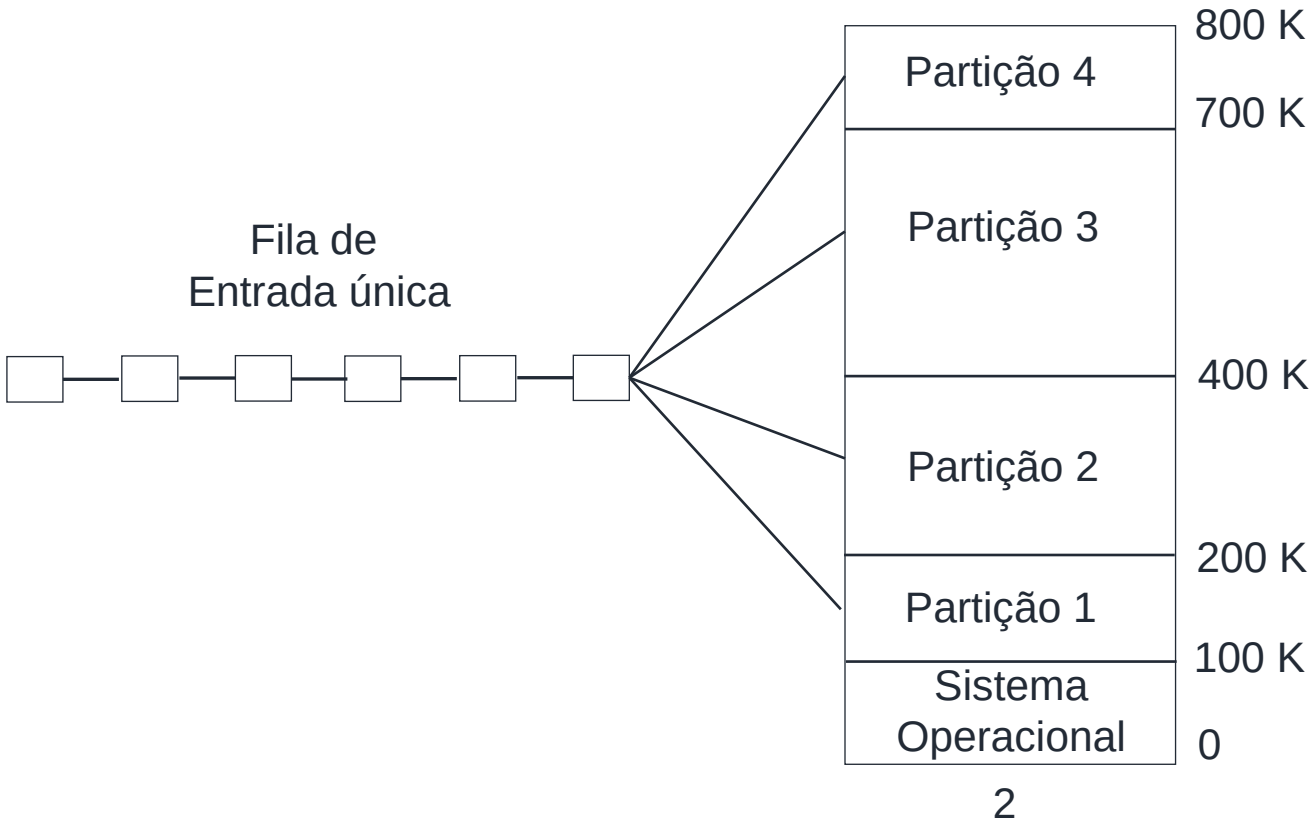
2

Filas única de entrada.

(Tanenbaum, 2016)

Multiprogramação - Problemas

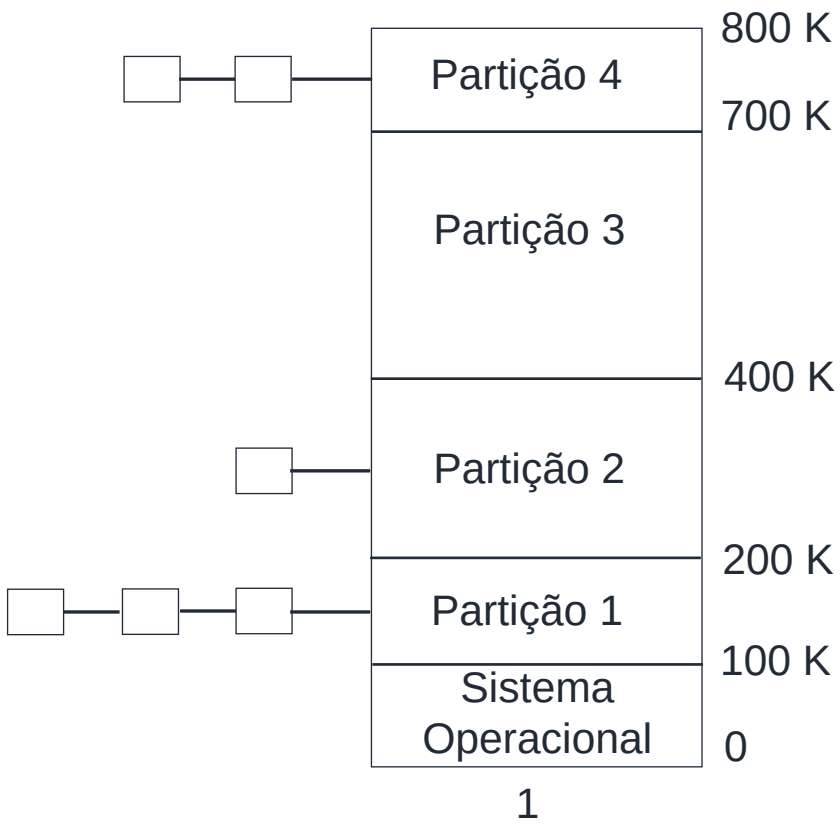


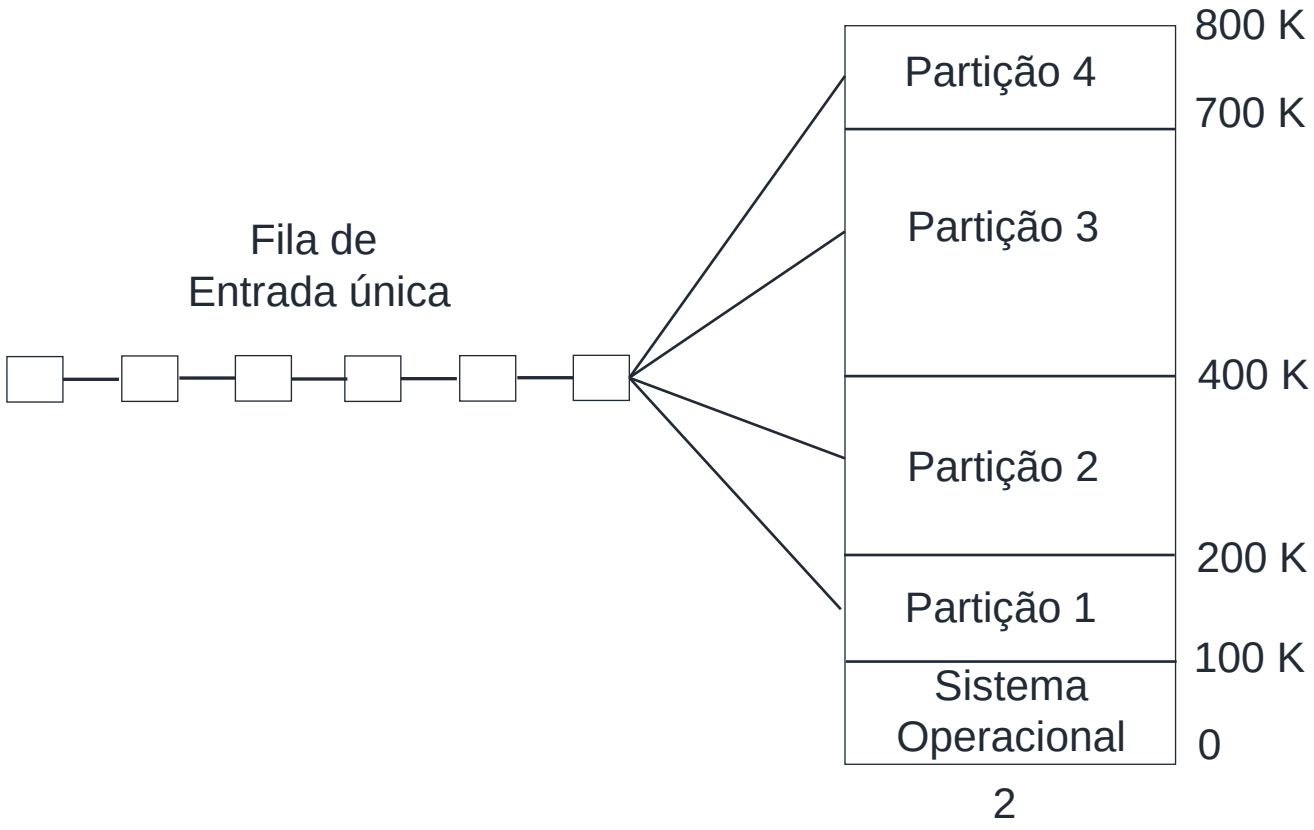


Não se sabe com certeza onde o programa será carregado na memória

- Localização de endereços de variáveis e de código de rotinas não podem ser absolutos.
- Os endereços das variáveis são referentes à onde os programas são carregados na memória. **Ex. Utilizando a partição 2, variável X na posição 2, início do programa alocado em 202K, a variável será alocada em 204K.**

Multiprogramação - Problemas



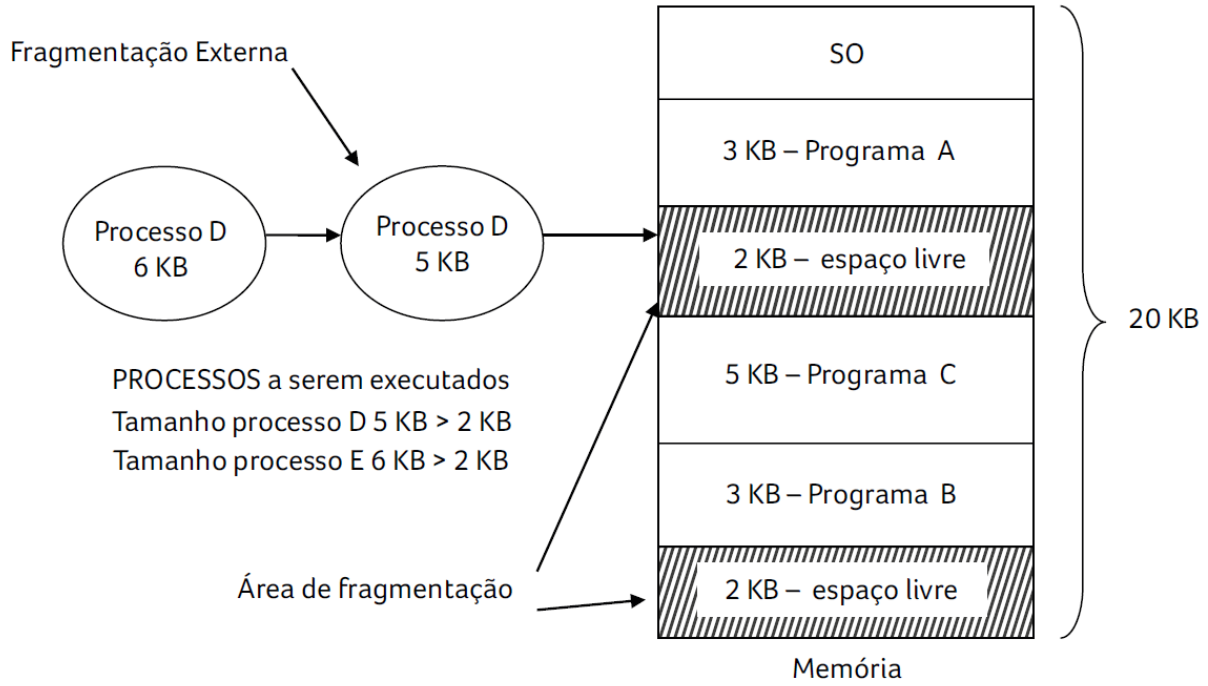


Não se sabe com certeza onde o programa será carregado na memória

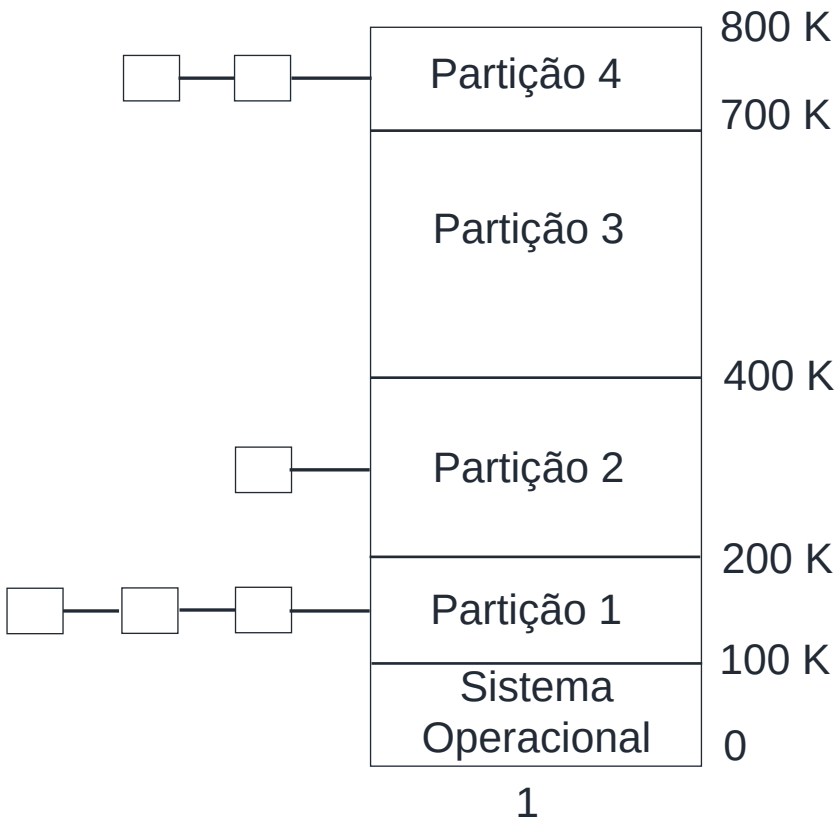
- Localização de endereços de variáveis e de código de rotinas não podem ser absolutos.
- Os endereços das variáveis são referentes à onde os programas são carregados na memória. **Ex. Utilizando a partição 2, variável X na posição 2, início do programa alocado em 202K, a variável será alocada em 204K.**

Multiprogramação - Problemas

- Imagine um sistema que possua **três (A, B, C) programas executados na CPU e outros dois programas a serem executados (D, E)**, além de uma memória dividida em seis partições: uma para o SO e as outras cinco para os programas. As partições destinadas para os programas são de tamanho 2 KB, 2 KB, 3 KB, 3 KB e 5 KB, e a destinada para o sistema operacional possui 5 KB. Então, o tamanho total de memória é de 20 KB.
- **Nesse cenário, os três programas utilizam as partições de 3 KB, 3 KB e 5 KB respectivamente, sobrando então uma área total livre de 4 KB**, porém dividida em duas partições de 2 KB. No entanto, os dois processos que precisam ser alocados na memória **possuem tamanho 6 KB e 5 KB**. Nota-se que o espaço disponível em memória é menor que os tamanhos de ambos os processos. Sendo assim, a memória fica fragmentada, pois há espaços, porém, não utilizados por processos. Essa fragmentação é chamada de fragmentação externa (MACHADO; MAIA, 2013).



Multiprogramação - Problemas



- Quando estou programando, utilizo variáveis e ponteiros. Os endereços das variáveis e dos ponteiros são relativos a que?
- Existe uma realocação de processos
- Um processo na partição dois cresceu demais, não cabe mais na partição 2 e será enviado para a partição 3. O endereço inicial que antes era 200K passou a ser 400K.

- A realocação faz com que tudo seja relativo ao seu ponto de carregamento.

- **Uma solução para realocação e proteção: Uso de valores base e limite**
- - Localização de endereços são somados ao valor base antes de serem mapeados na memoria física.
- - Localização de endereços maior do que o valor limite indicam erro.

Multiprogramação com Partições Variáveis de Memória

PUC Minas

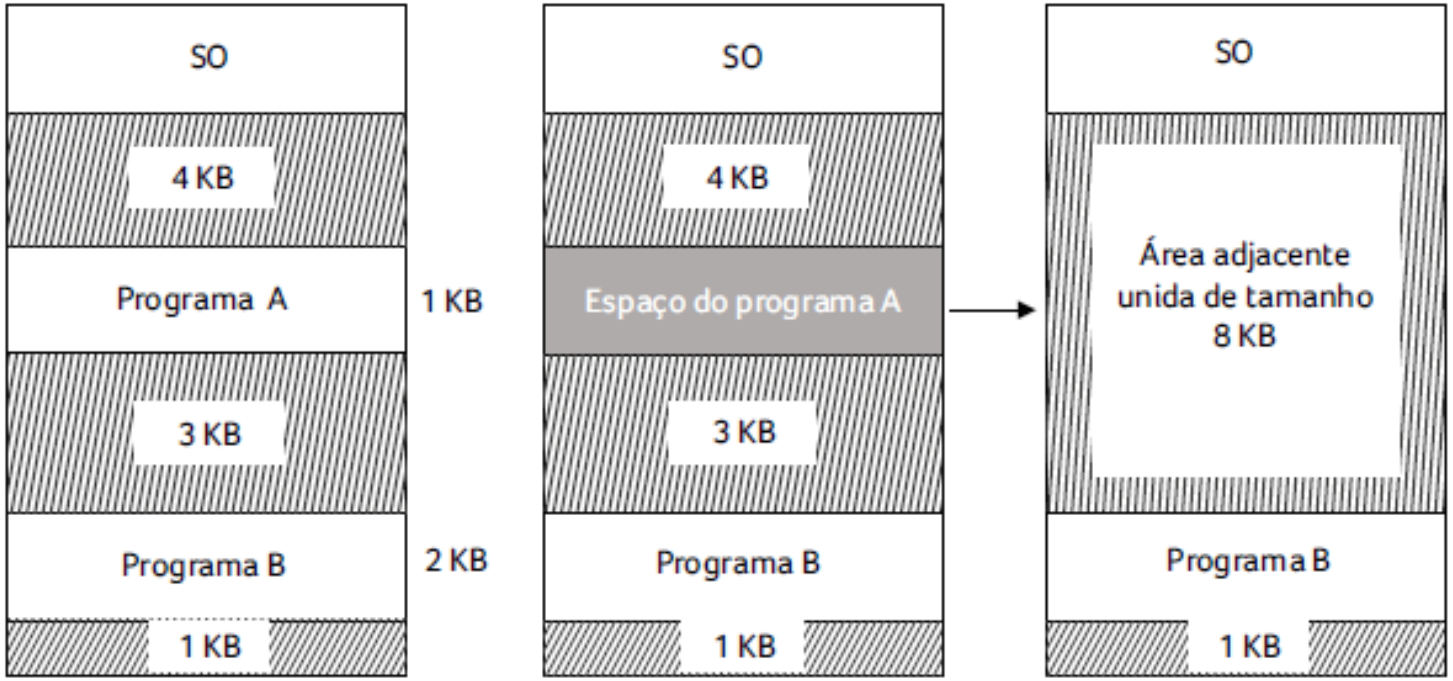
Necessidade de aumentar o grau de compartilhamento da memória. **Com partições variáveis, o conceito de partições fixas foi eliminado.** Na multiprogramação com partições variáveis, **cada programa usa o espaço necessário para ser executado na memória**. Porém, esse método pode criar espaços de memória que não podem ser alocados por outros processos.

Essa área de memória que não é utilizada por outro processo pelo fato de ser menor que a tarefa a ser armazenada em memória é chamada de fragmentação.

Multiprogramação - Fragmentação

PUC Minas

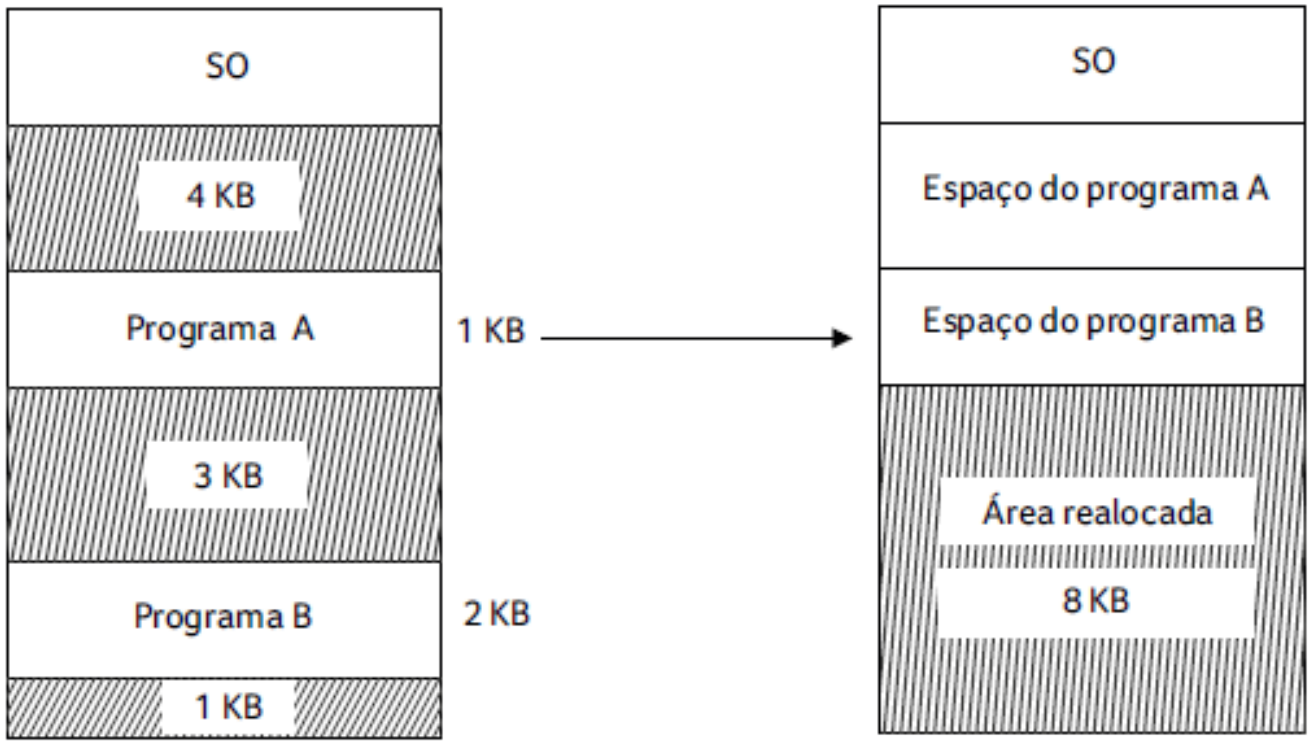
Fragmentação de Espaços Adjacentes



Multiprogramação - Fragmentação

PUC Minas

Realocação de Partição



Swapping

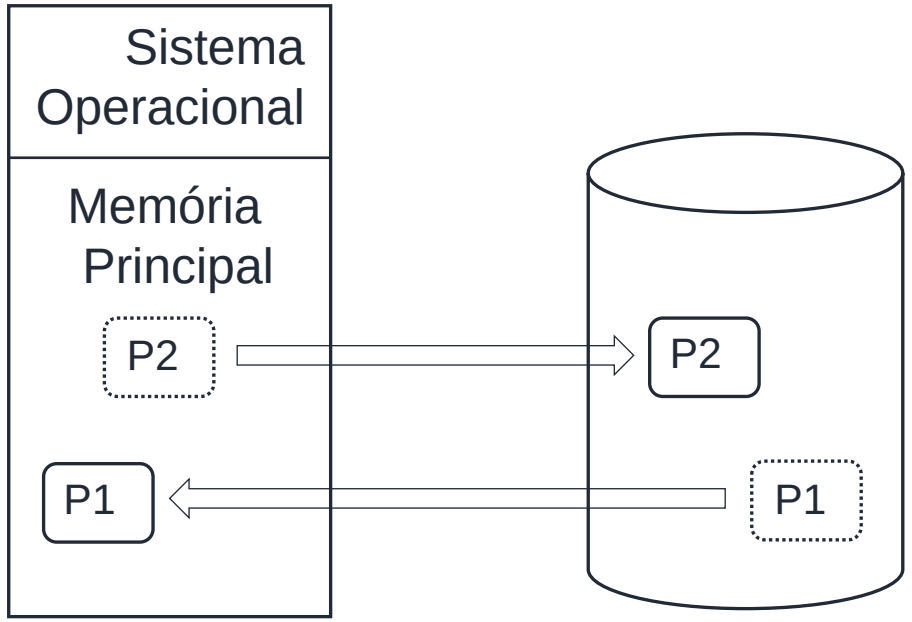
- Memória de *swapping* (troca) **é uma memória secundária que tem a finalidade de prover, ajudar a memória principal com recursos extras para armazenamento de *cache* (temporário).**
- Ela pode ser utilizada **como memória de troca de arquivos**, quando necessário, pelo **sistema operacional**, geralmente em uma situação onde há pouca memória RAM. Excesso de troca, nesse caso, pode levar à lentidão do sistema operacional.
- **Processos ou parte deles também fazem uso da área de *swap*.** O *swap* é utilizado para alocar espaço aos processos, mesmo quando a RAM já está cheia.

Técnicas de Troca de Processos

PUC Minas

1) Troca de Processos (***swapping***):

- Consiste em trazer, em sua totalidade, cada processo para memória, executá-lo durante um certo tempo, e então, devolvê-lo ao disco.



Técnicas de Troca de Processos

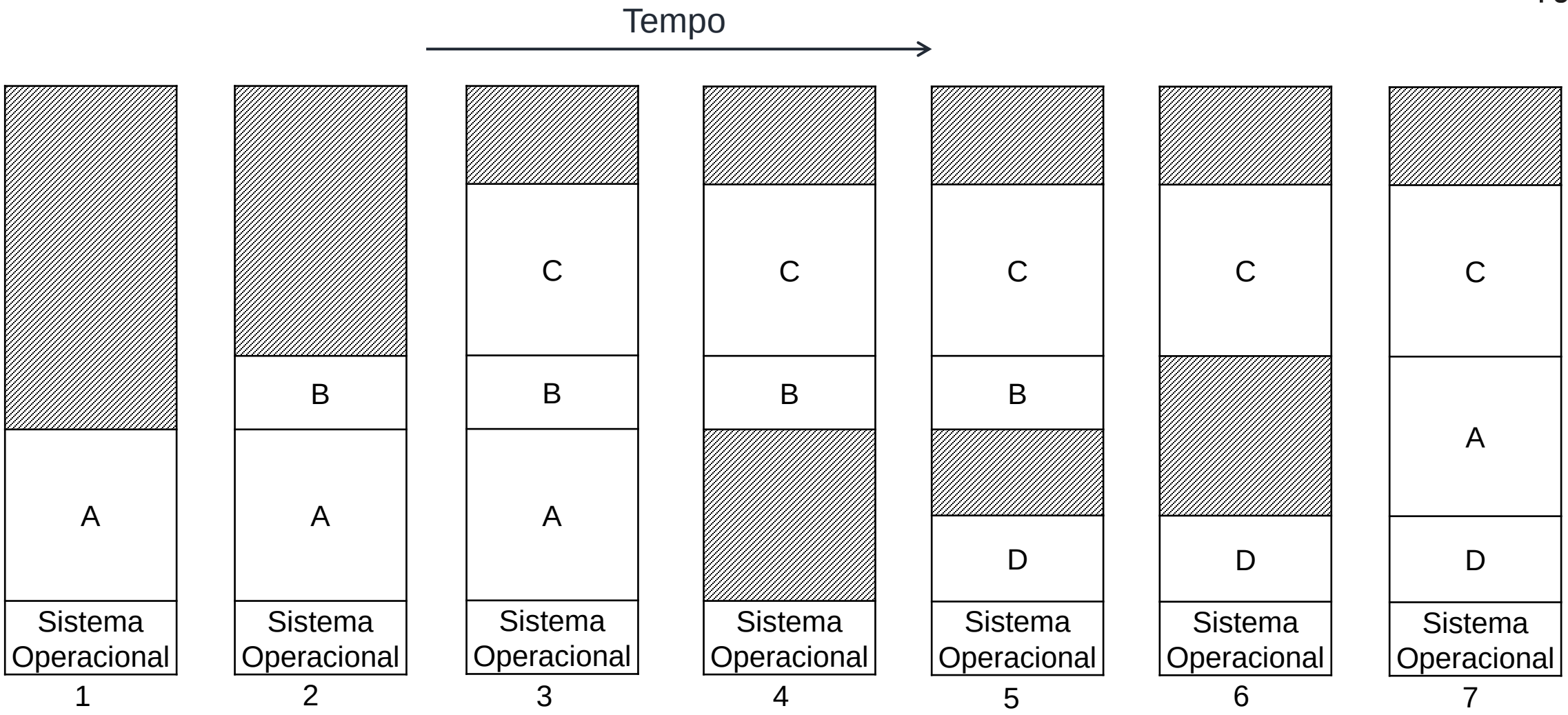
PUC Minas

2) Memória Virtual: Permite que programas possam ser processados mesmo que estejam parcialmente na memória.

- Criado em 1961 o método Fortheringham, ficou conhecido como memória virtual. **A ideia básica por trás da memória virtual é que cada programa tem seu próprio espaço de endereçamento, que é divido em blocos chamados páginas.**
- Essas páginas são mapeadas na memória física, mas nem todas precisam estar na memória física para executar o programa.
- **Quando o programa referencia uma parte de seu espaço que está na memória física, o hardware executa a instrução.**
- **Quando o programa referencia uma parte de seu espaço que não está na memória física, o sistema operacional é alertado para obter a parte que falta e reexecutar a instrução que falhou.**

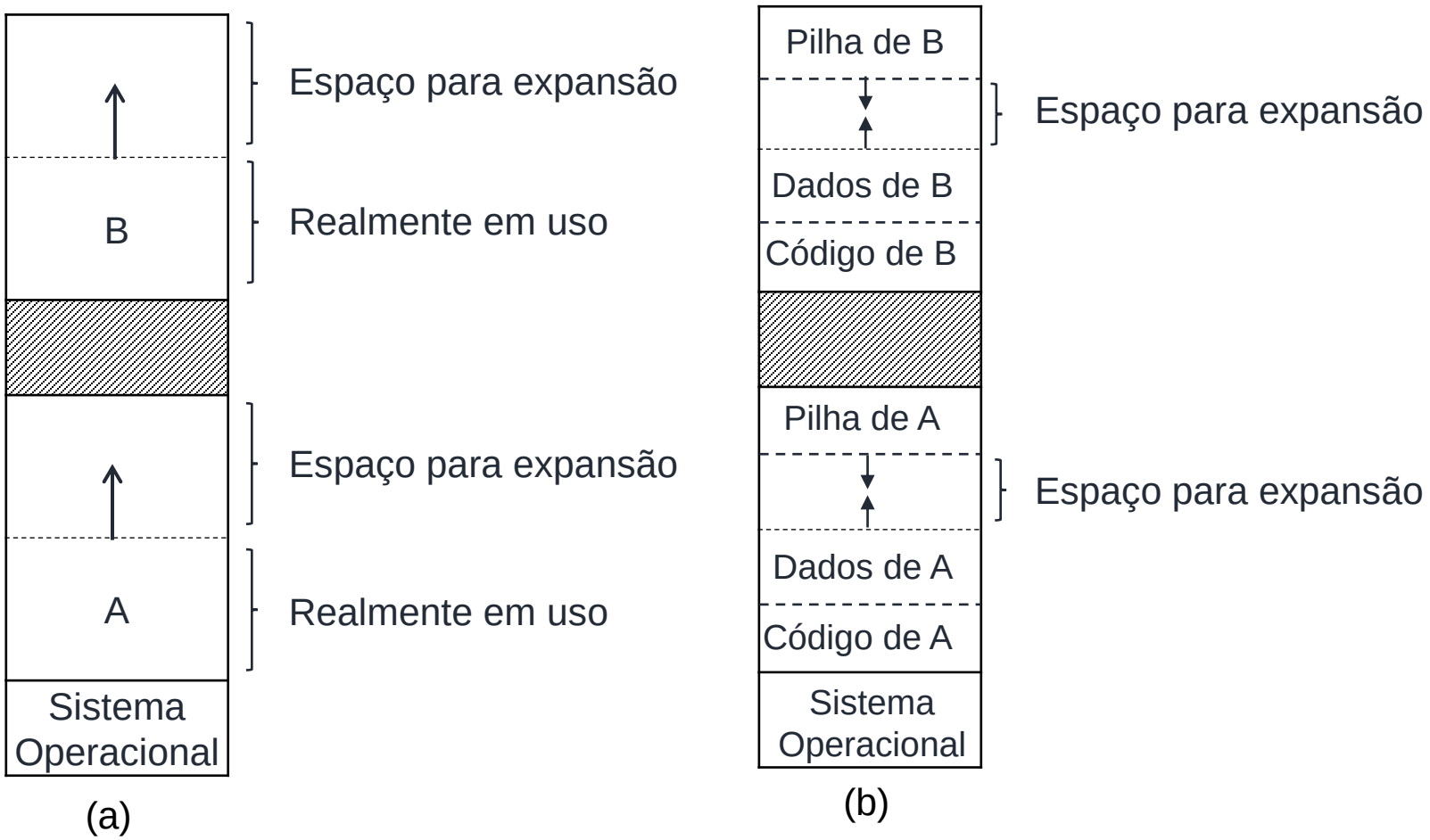
(Tanenbaum, 2016)

Alteração na alocação de memória à medida que o processos entram e saem dela.



As regiões sombreadas correspondem às regiões de memória livre ou não utilizadas naquele instante.

Técnicas de Troca de Processos



Pilha de Execução: Realiza o controle da execução de subrotinas de um programa.

Um programa quando é carregado para a memória, é dividido em três regiões: código, dados e pilhas de métodos em execução.

Entre dados e pilhas existe um espaço vago para a expansão das chamadas dos métodos e para a expansão dos dados.

A pilha cresce de cima para baixo à medida que métodos vão sendo chamados métodos, e os dados crescem de baixo para cima à medida que são criadas variáveis..

Gerenciamento de Memória Livre

PUC Minas

Mapas de Bits

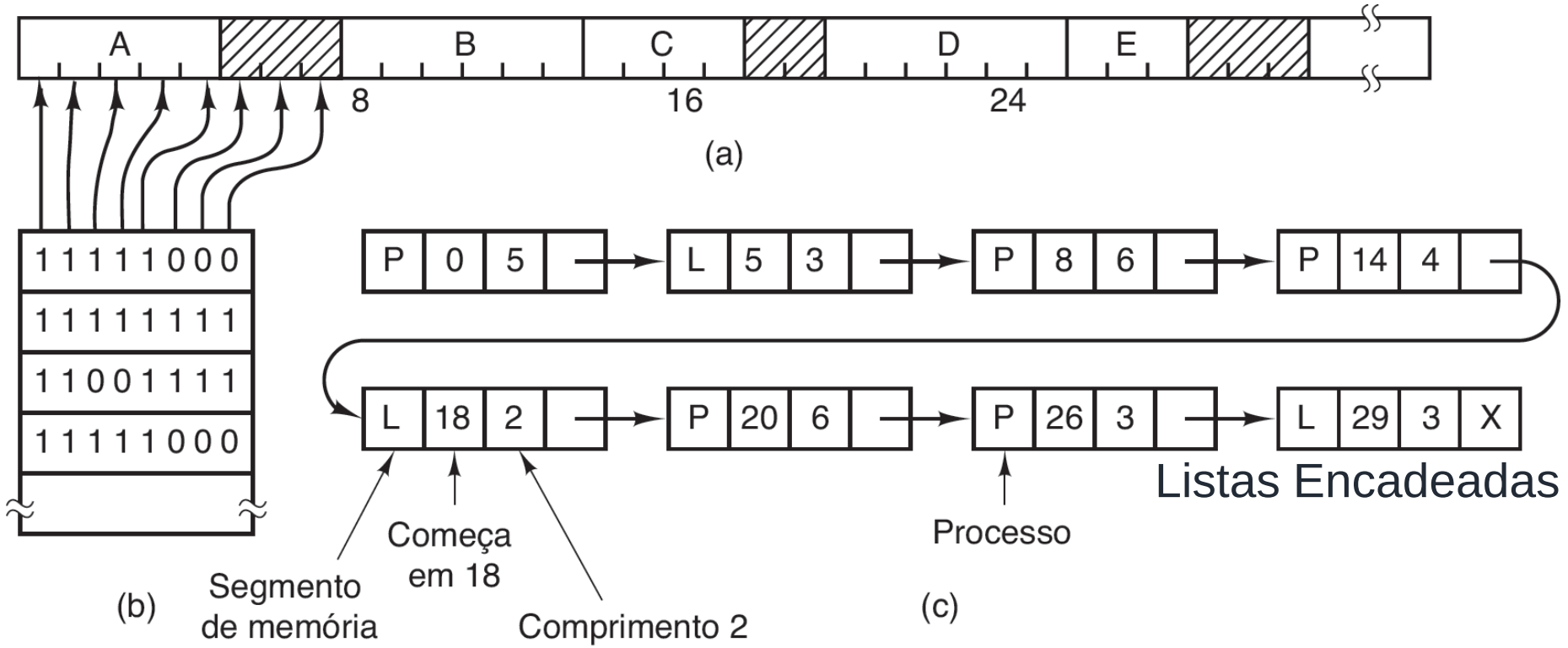
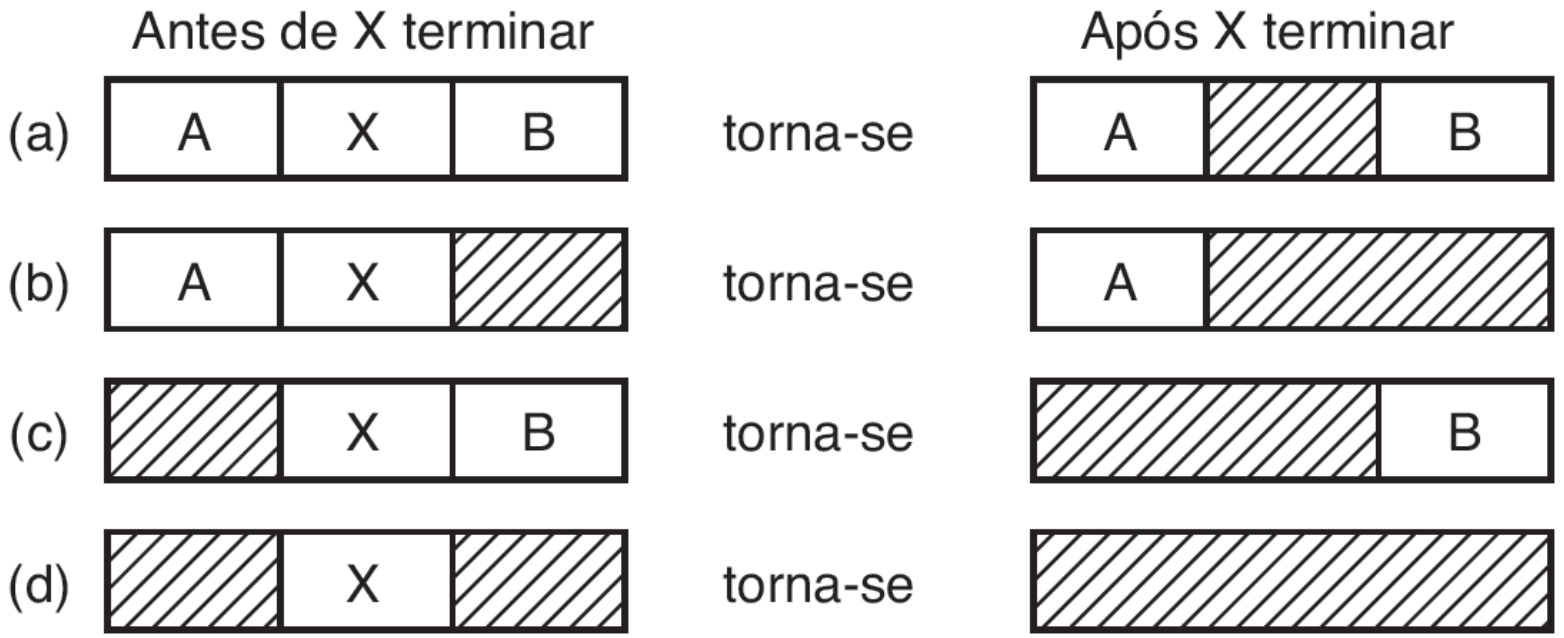


Figura 3.6 (a) Parte da memória com cinco processos e três segmentos de memória. As marcas mostram as unidades de alocação de memória. As regiões sombreadas (0 no mapa de bits) estão livres. (b) O mapa de bits correspondente. (c) A mesma informação como lista.

(Tanenbaum, 2009)



Quatro combinações de vizinhos para o processo que termina, *X*.

Memória virtual - Paginação

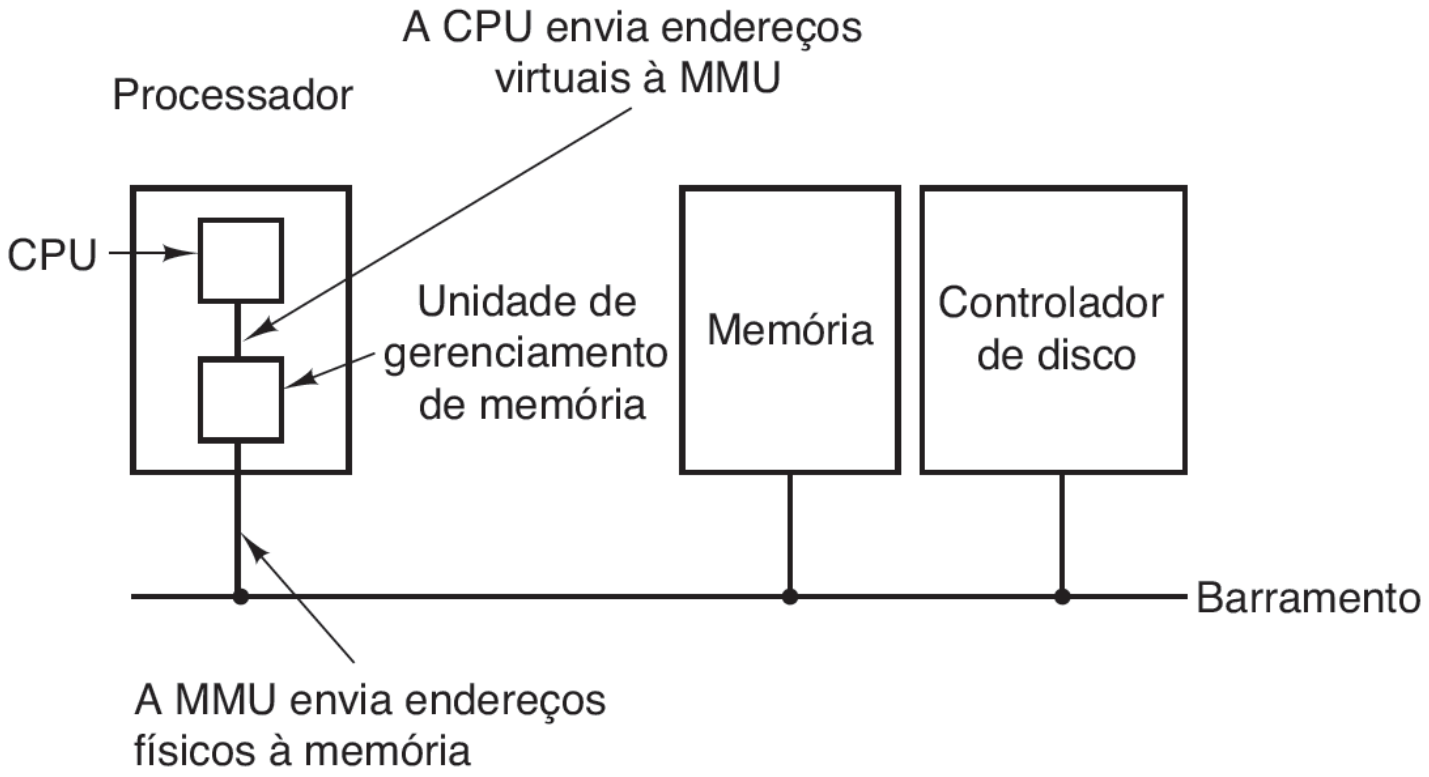


Figura 3.8 A posição e a função da MMU. Aqui a MMU é mostrada como parte do chip da CPU (processador) porque isso é comum atualmente. Contudo, em termos lógicos, poderia ser um chip separado, como ocorria no passado.

MMU

Unidade de Gerenciamento de Memória, é um dispositivo de hardware que traduz endereços virtuais em endereços físicos.

MMU

Gerenciamento de memória virtual

Memória virtual - Paginação

PUC Minas

MMU

- ✓ Faz a tradução de endereços de páginas virtuais para moldura de página reais.

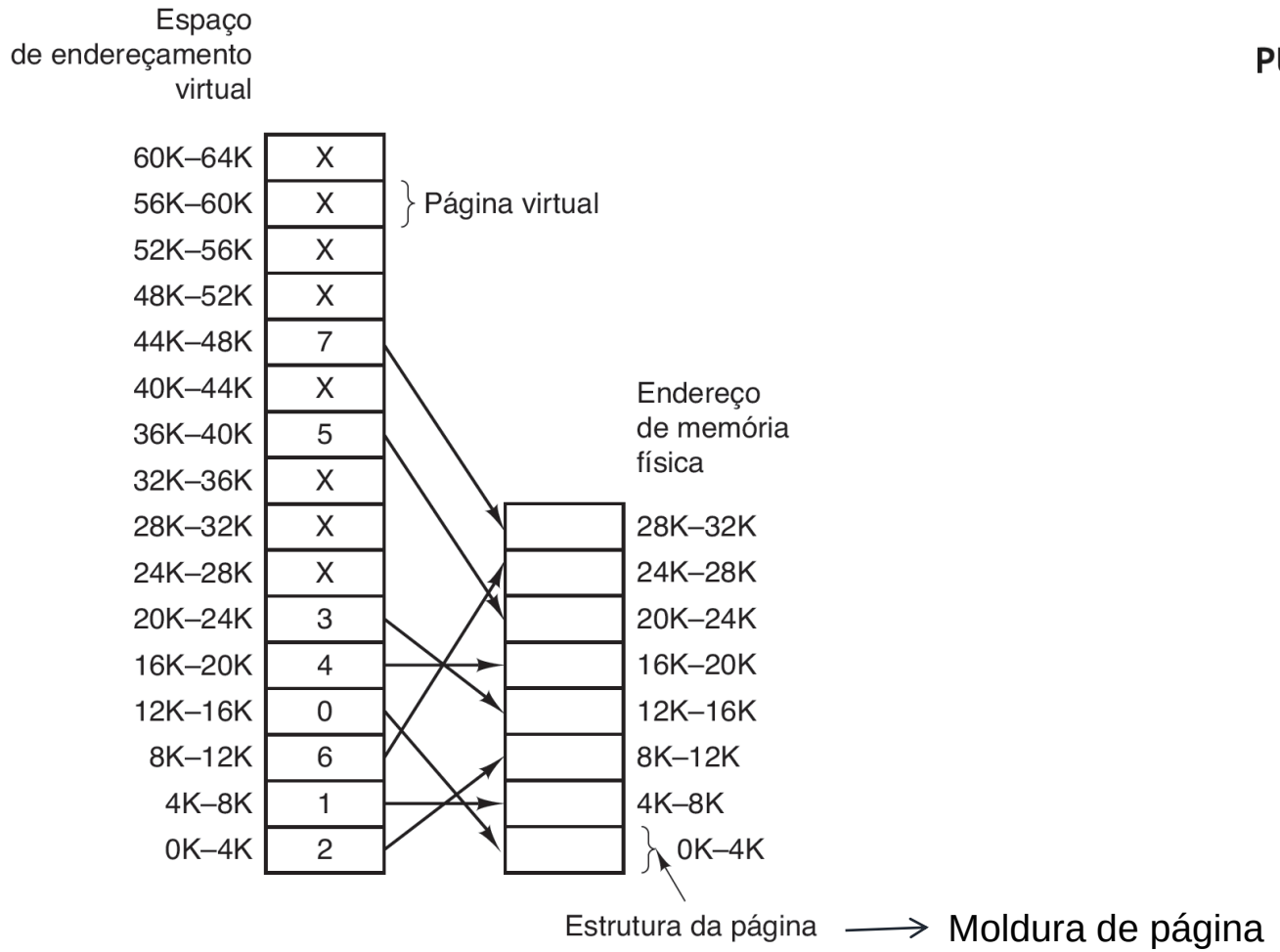


Figura 3.9 A relação entre endereços virtuais e endereços de memória física é dada pela tabela de páginas. Cada página começa com um múltiplo de 4096 e termina 4095 endereços acima; assim, 4K–8K na verdade significa 4096–8191 e 8K–12K significa 8192–12287.

Memória virtual - Paginação

PUC Minas

- A falta de página força uma escolha:
  - Qual página deve ser removida.
  - Alocação de espaço para a página que vai ser trazida para a memória.

- A página modificada deve primeiro ser salva
  - Se não tiver sido modificada é apenas sobreposta.

- Melhor não escolher uma página que está sendo muito utilizada
  - Provavelmente esta página precisará ser trazida de volta em um breve espaço de tempo.

Algoritmos

PUC Minas

Diversos algoritmos são utilizados para substituição de páginas, como:

- Algoritmo ótimo de substituição de página.
- Algoritmo de substituição de página não usado recentemente.
- Algoritmo de substituição de página primeiro a entrar, primeiro a sair.
- Algoritmo de substituição de página segunda chance.
- Algoritmo de substituição de página de relógio.
- Algoritmo de substituição de página usado menos recentemente.
- Algoritmo de substituição de página de conjunto de trabalho.
- Algoritmo de substituição de página WSClock.

Referências

PUC Minas

- STUART, Brian L.. Princípios de sistema operacionais: projetos e aplicações. São Paulo: Cenage Learning, 2011. 637 p. ISBN: 9788522107339.
- TANENBAUM, Andrew S. Sistemas Operacionais Modernos. 4a ed. São Paulo: Pearson, 2016. 864 p. ISBN: 9788543005676.
- MACHADO, Francis Berenger; MAIA, Luiz Paulo. Arquitetura de sistemas operacionais. 5a ed. Rio de Janeiro: LTC, 2013. 263 p. ISBN:9788521622109.
- SILBERSCHATZ, Abraham. Fundamentos de Sistemas Operacionais. 9ª ed. LTC, 2015. 524 p. ISBN: 9788521629399. ISBN digital 9788521630012.