Sistemas Operacionais

Gerência de Memória

Prof. Dr. Fábio Rodrigues de la Rocha

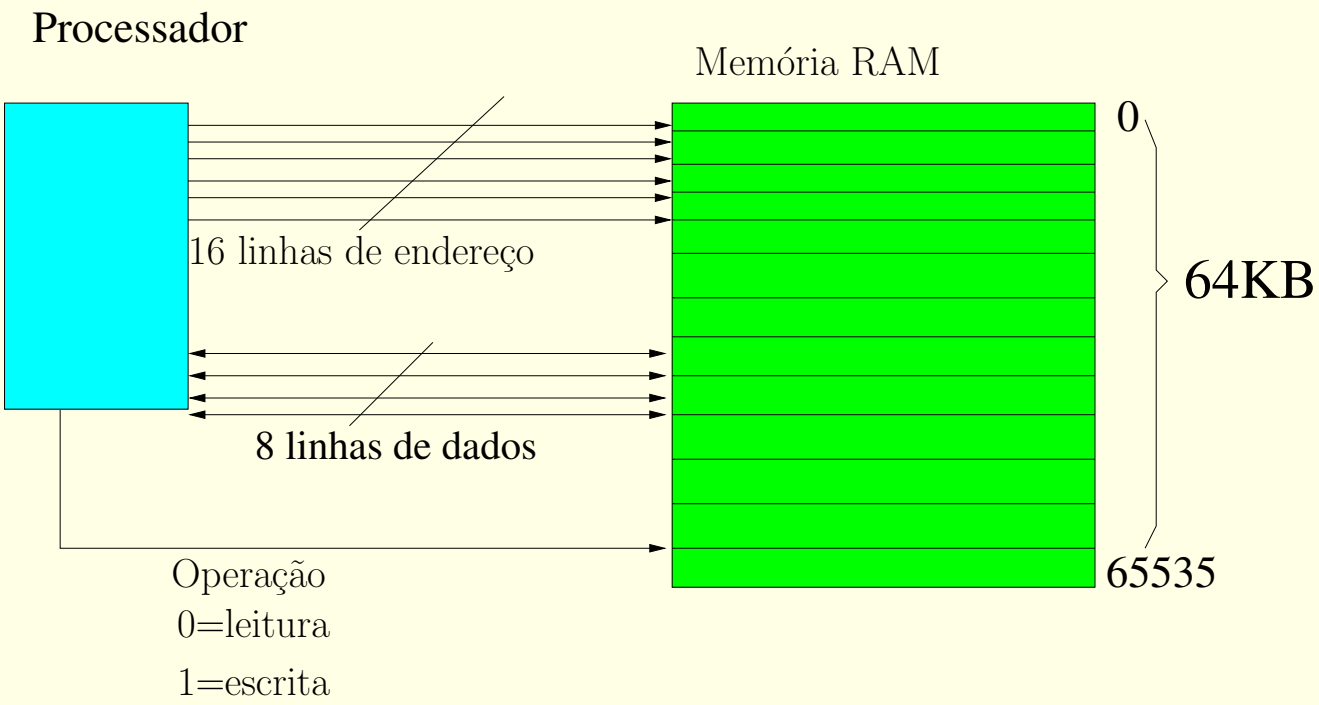
UTFPR

Gerência de memória

No estudo anterior, aprendemos que existem vários processos na memória do computador e que estes são escolhidos pelo **escalonador** para ter o direito de utilizar o processador. Como existem vários processos na memória do computador e a memória é apenas uma, algum mecanismo deve existir para permitir que os processos funcionem corretamente, ou seja, que um processo não use a memória que pertence a outro processo e consiga reservar a memória de que precisa.

UTFPR

Gerência de memória



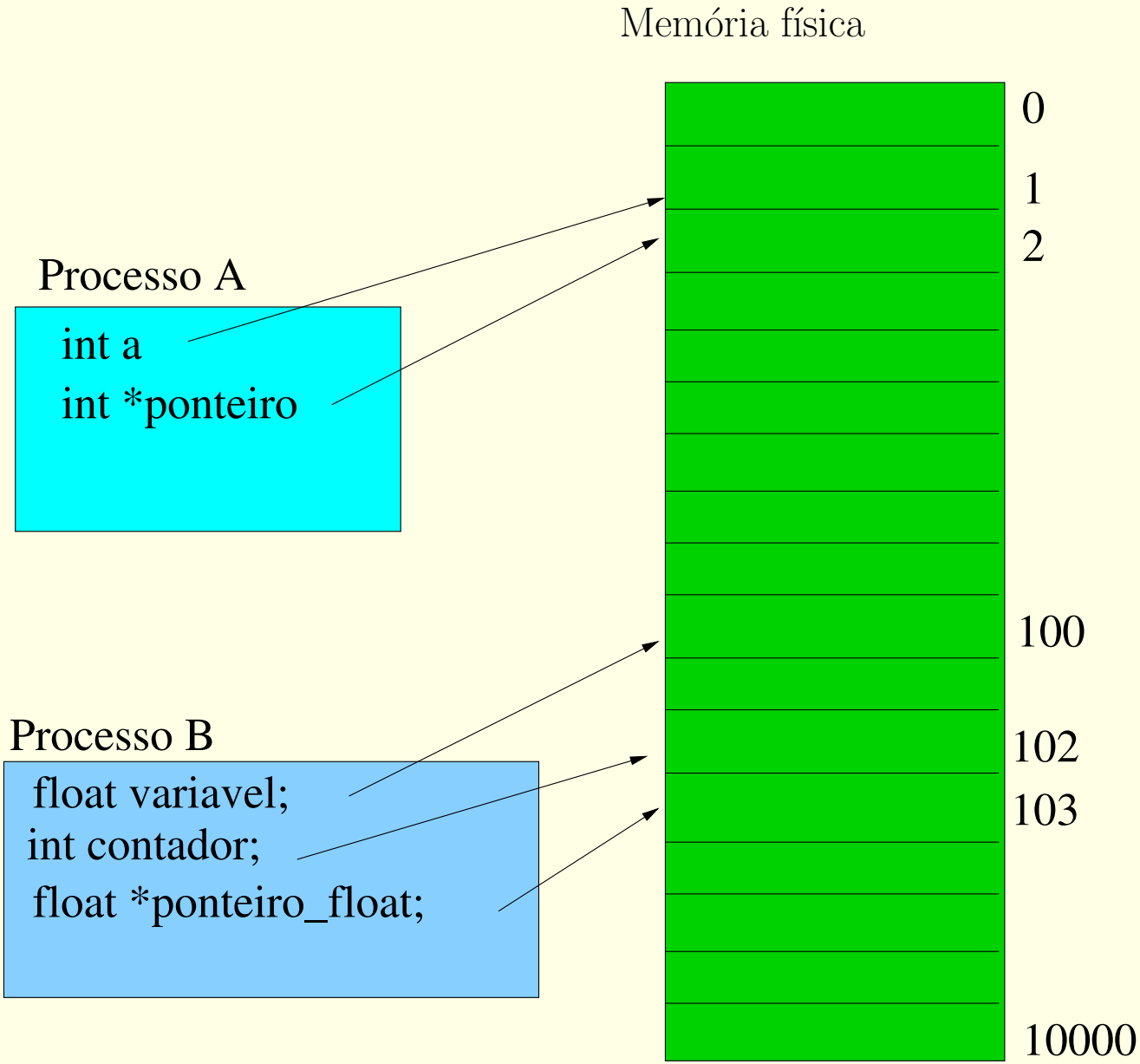
Memória lógica

Os endereços que um processo acessa são chamados endereços lógicos. Por exemplo: O código fonte C abaixo.

```c
// Processo A
#include <stdio.h>

int main ()
{
    int a;
    int *ponteiro;

    ponteiro=&a;
    Calcula_Soma();
    .
    .

    return 0;
}
```

```c
// Processo B
#include <stdio.h>

int main ()
{
    float variavel;
    int contador;
    float *ponteiro_float;

    ponteiro_float=&variavel;
    Calcula_Produto();
    .
    .
    return 0;
}
```

Gerência de memória



Gerência de memória

Memória física

É a memória implementada pelos circuitos eletrônicos do computador. O endereço físico é o endereço que é utilizado para endereçar a memória.

Gerência de memória - proteção de memória

Proteção de memória

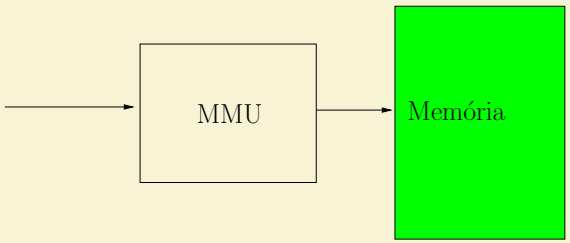
Imagine como seria o mundo se num sistema multitarefa um programa acesse a memória que pertence a outro programa e modifique-a ? Naturalmente, o programa que teve sua memória adulterada não funcionará como esperado. Essa situação mostra que um processo não pode acessar a memória de outro processo. O sistema operacional deve implementar algum mecanismo que permita bloquear o acesso de um processo a memória de outro.

Gerência de memória - proteção de memória

MMU

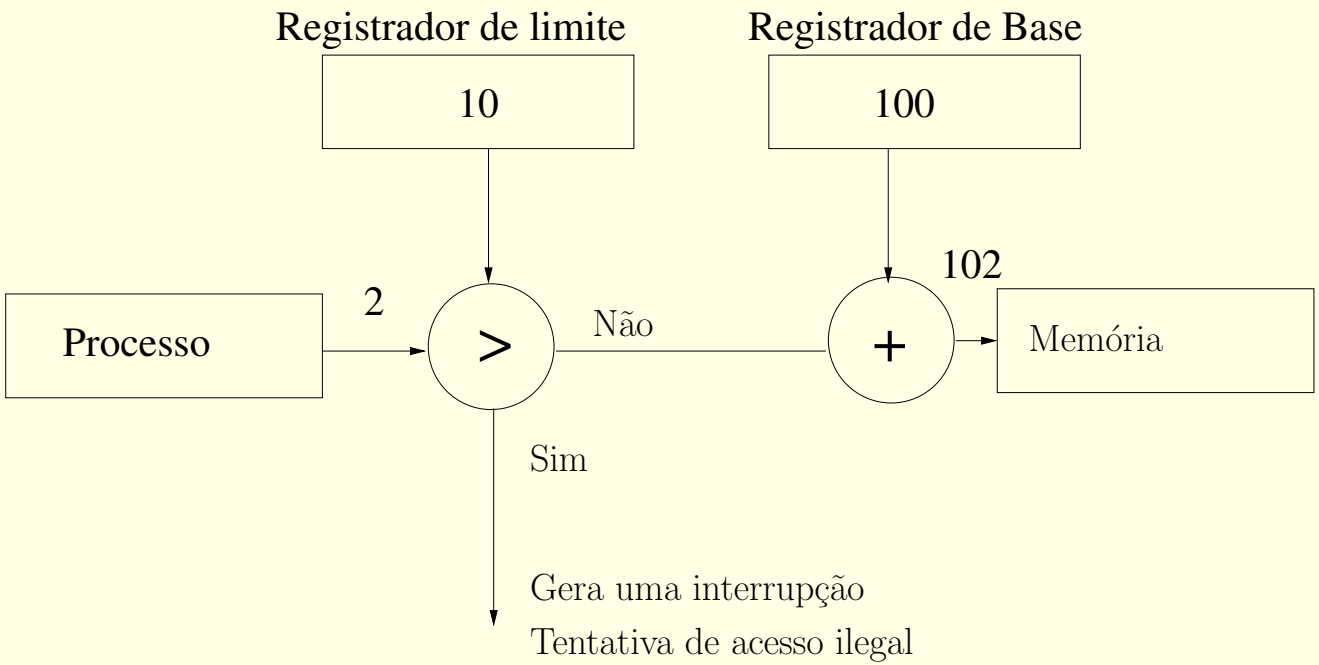
MMU é um circuito que existe dentro de um processador e que é responsável por mapear os endereços lógicos em endereços físicos que são enviados para a memória. A figura abaixo mostra que a MMU faz o meio de campo entre o processador e a memória.

```
// Processo B
#include <stdio.h>

int main ()
{
   float variavel;
   int contador;
   float *ponteiro_float;

   ponteiro_float=&variavel;
   Calcula_Produto();
   .
   .
   return 0;
}
```



End. lógico	End. físico
0	100
2	102
3	103

Gerência de memória - proteção de memória



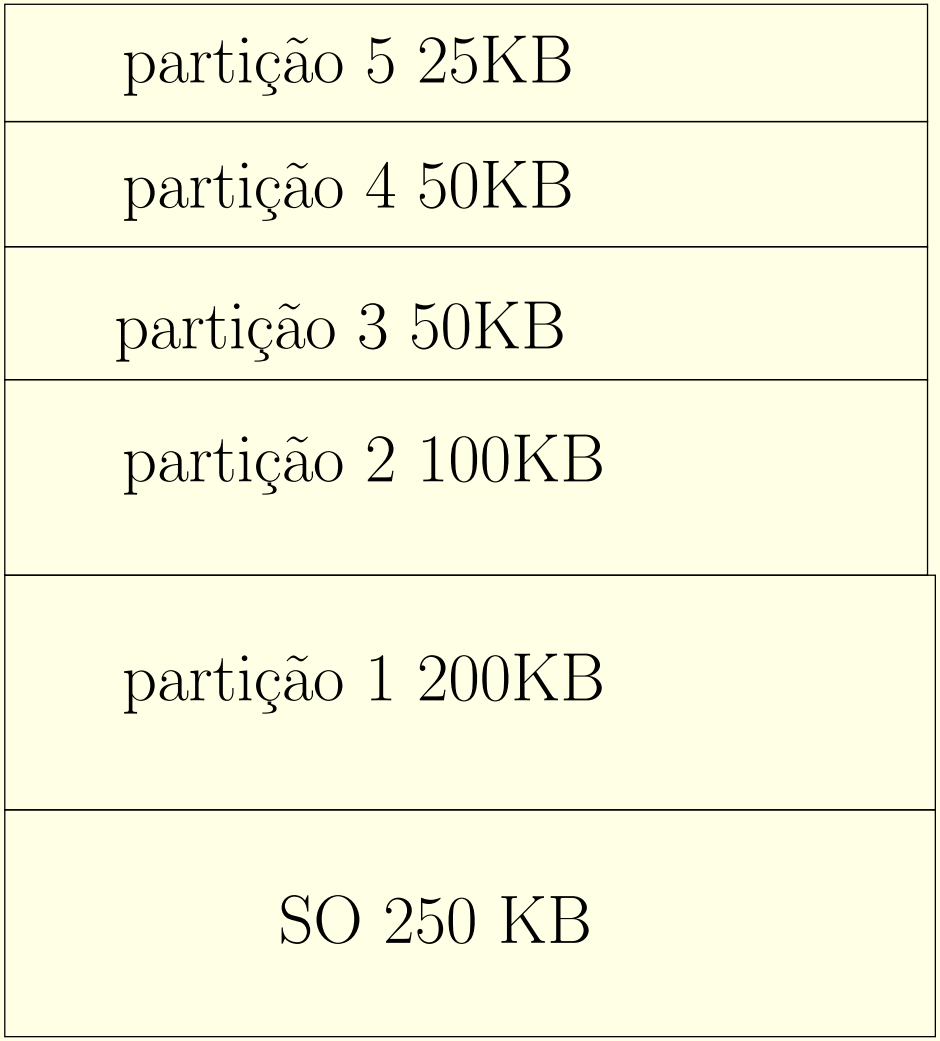
Gerência de memória - proteção de memória

```c
// Programa que tenta acessar enderecos
// que nao estao na sua faixa de enderecos validos
#include <stdio.h>

int main ()
{
    int *x;  //x eh um ponteiro para inteiro
    x=(int *)10000; // x aponta para o endereco 10000
    // esse endereco nao esta na faixa de enderecos
    // validos para este processo
    printf("Aparece\n");
    *x=10; //Escreve o valor 10 na posicao 10000

     //A instrucao acima causa uma interrupcao
     //O programa sera encerrado
     //A linha abaixo nao sera executada
     printf("Nao aparece\n");

    return 0;
}
```

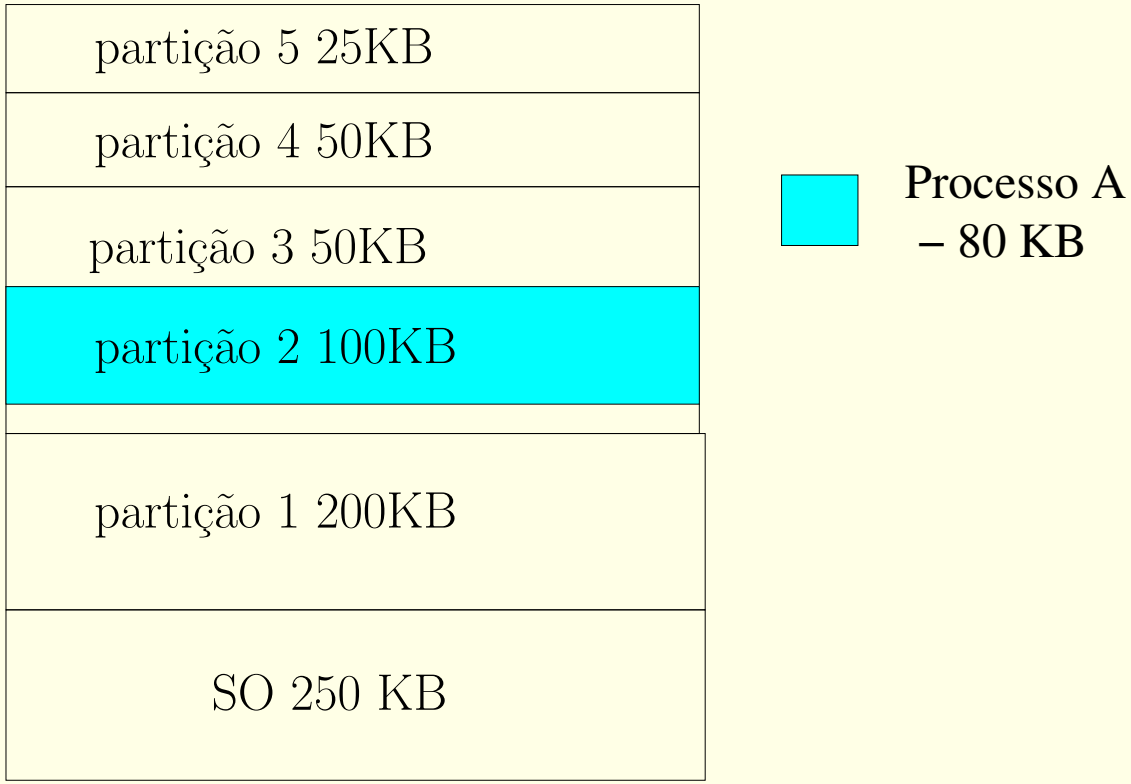
Gerência de memória - Alocação de memória

- Partições Fixas;
- Partições variáveis;

Gerência de memória - Alocação de memória



Gerência de memória - Alocação de memória



Gerência de memória - Alocação de memória

Fragmentação interna

Fragmentação interna é a memória perdida dentro de uma partição. No slide anterior o processo A precisou de 80KB memória, então foi alocada uma partição de tamanho 100KB. Os 20KB não utilizados foram perdidos.

Fragmentação externa

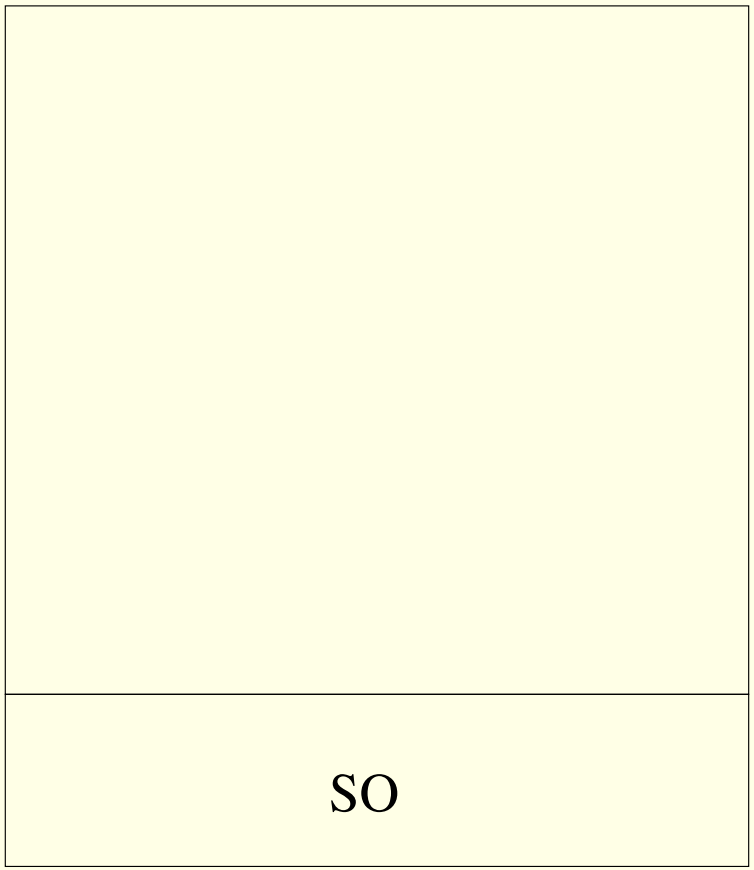
Voltando o Slide da alocação de memória, percebe-se que existe muita memória livre (345 KB). Porém, se desejarmos alocar memória para um processo B que necessita de 150 KB, não será possível. A razão disso é que não existe tanta memória **contígua**.

Gerência de memória - Alocação de memória

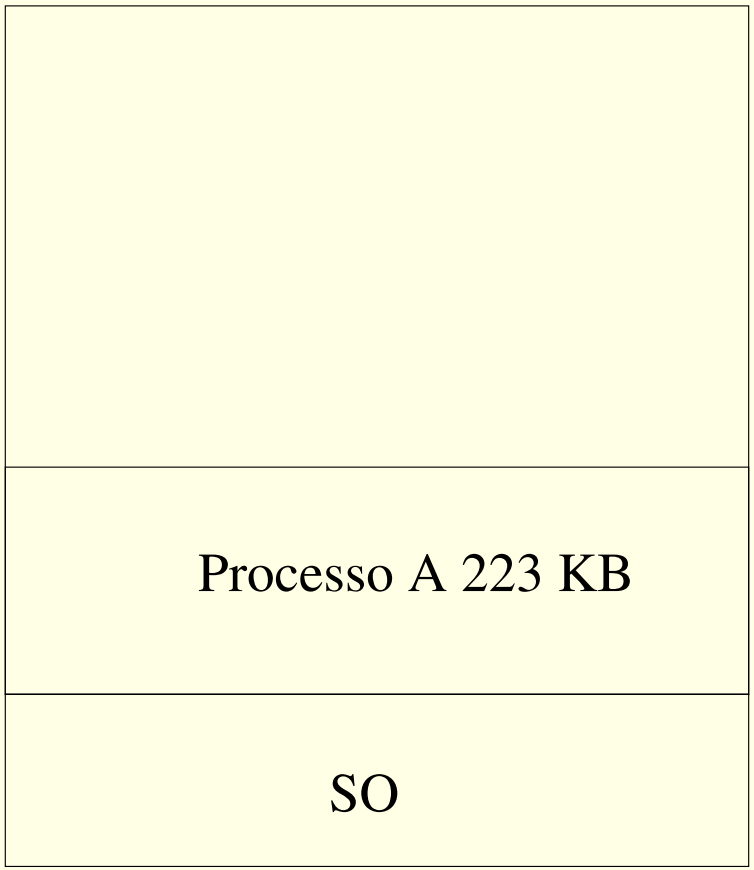
Partições variáveis

No método de partições variáveis, o tamanho das partições é ajustado dinamicamente às necessidades exatas dos processos. O SO quando realiza a alocação de memória de um processo deve varrer a memória e decidir a faixa de endereços em que um processo será escrito. Chamamos de **lacunas** às áreas de memória que estão livres (ou seja, não estão reservadas para um processo).

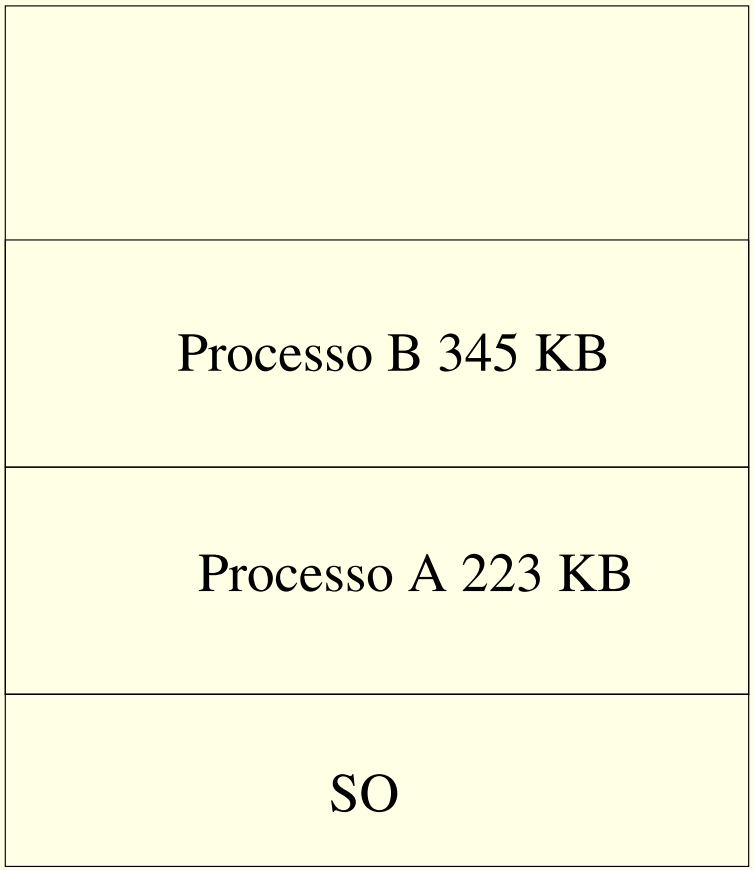
Gerência de memória - Alocação de memória



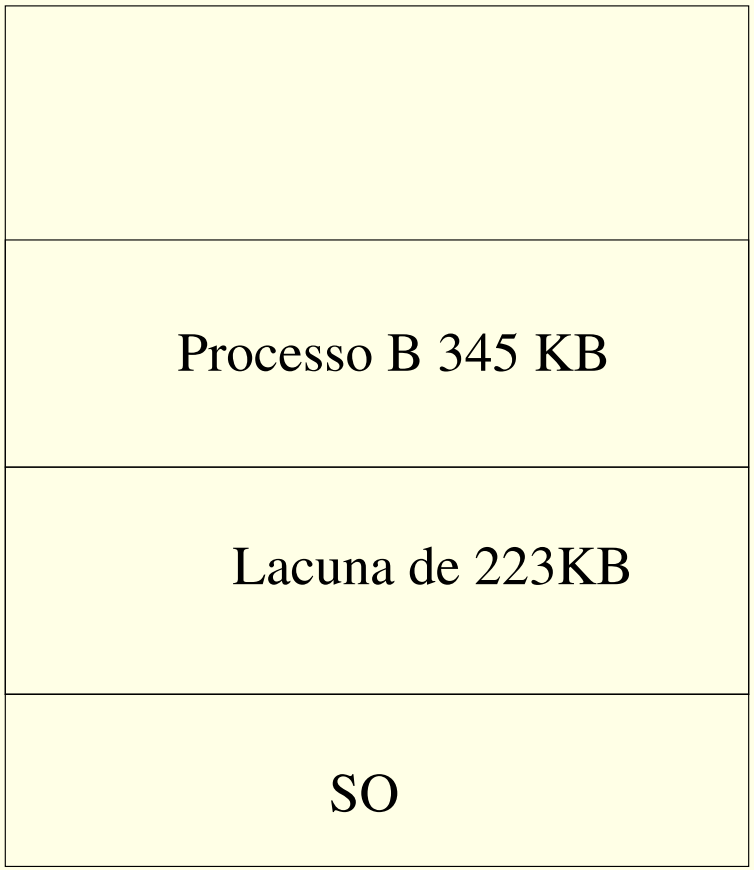
Gerência de memória - Alocação de memória



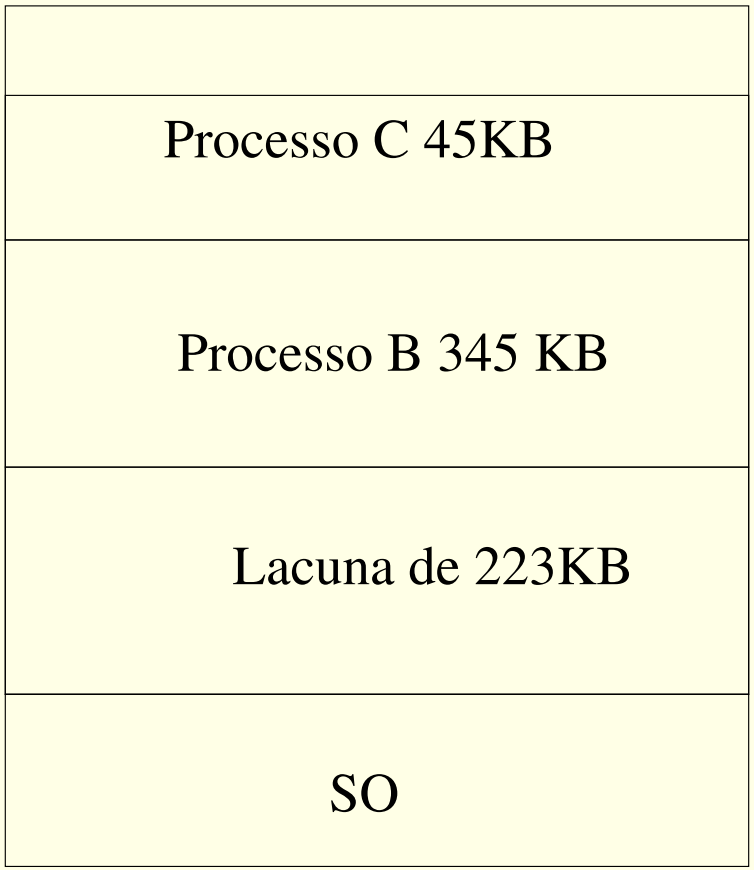
Gerência de memória - Alocação de memória



Gerência de memória - Alocação de memória



Gerência de memória - Alocação de memória



Gerência de memória - Alocação de memória

Algoritmos para alocação de memória:

- First-Fit;
- Best-Fit;
- Worst-Firt;