Gerência de Memória

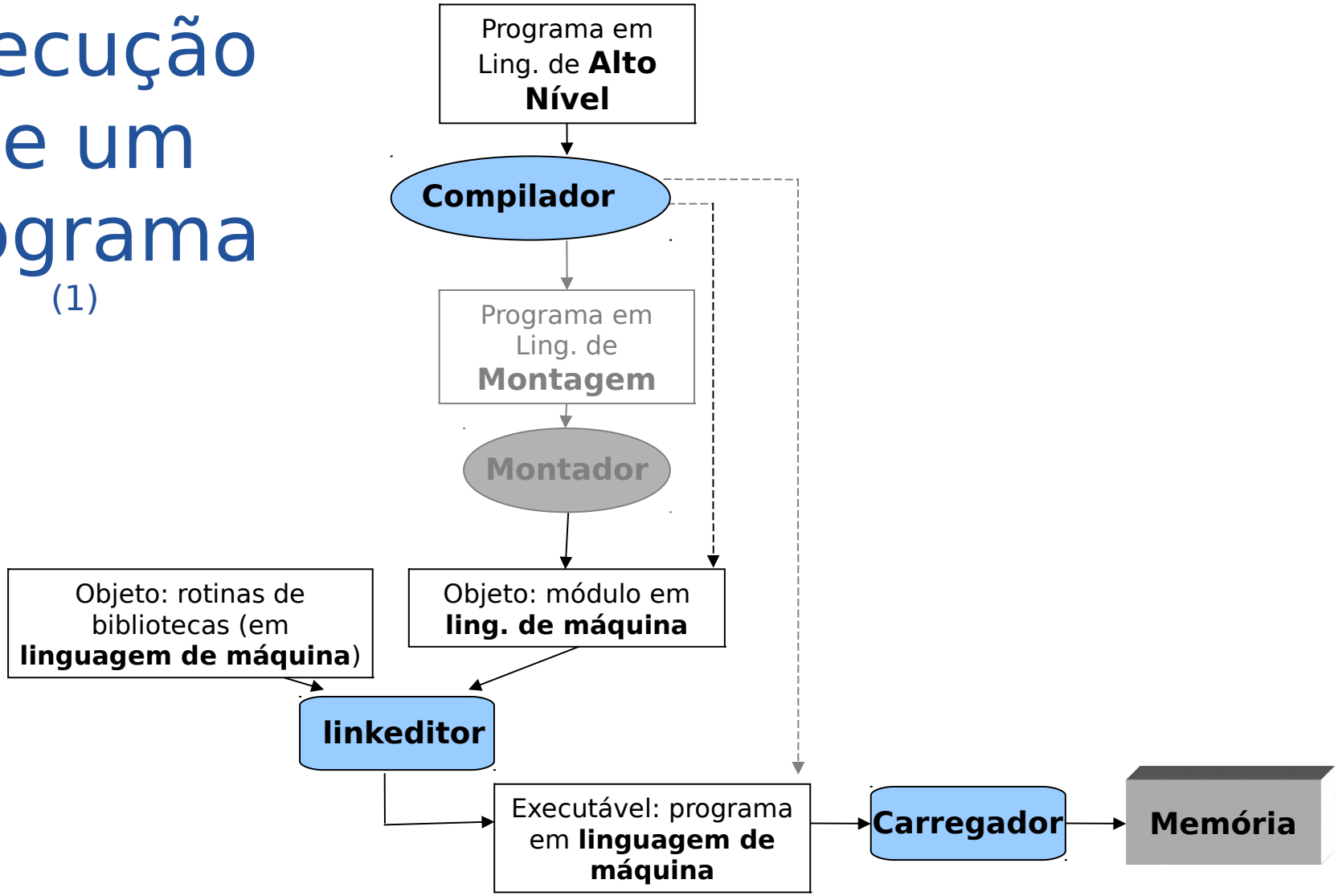
Introdução

- Considerações:
  - Recurso caro e escasso;
  - Programas só executam se estiverem na memória principal;
  - Quanto mais processos residentes na memória principal, melhor será o compartilhamento do processador;
  - Necessidade de uso otimizado;
  - O S.O. não deve ocupar muita memória;
  - “É um dos fatores mais importantes em um projeto de S.O.”.

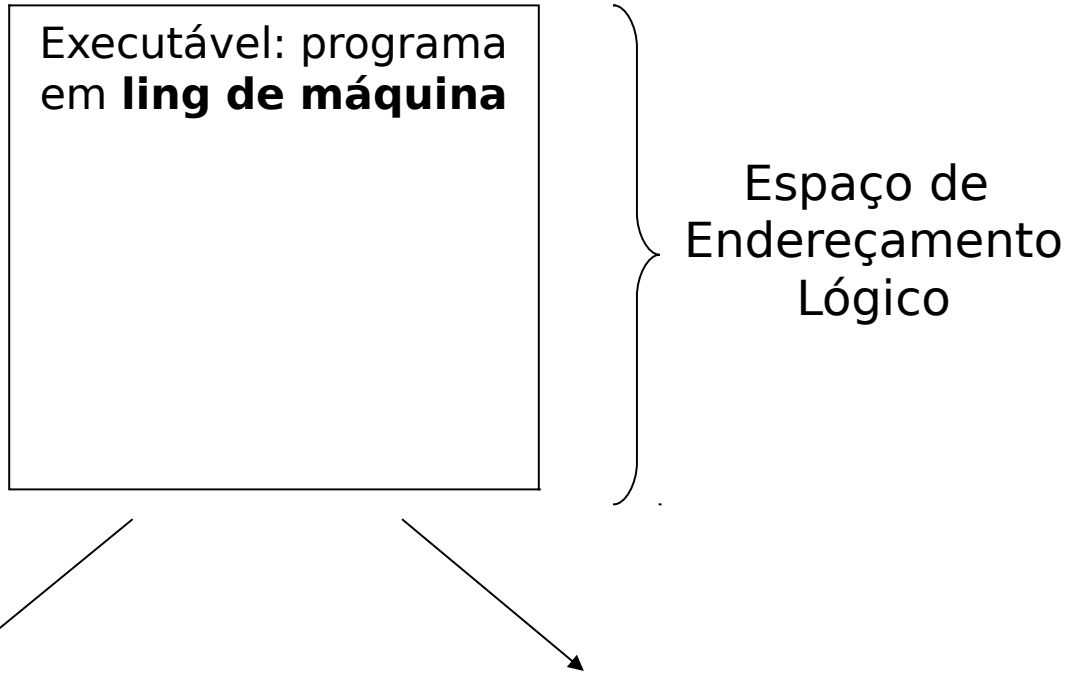
Introdução

- Sistema operacional deve
  - controlar quais regiões de memória são utilizadas e por qual processo
  - decidir qual processo deve ser carregado para a memória, quando houver espaço disponível
  - alocar e desalocar espaço de memória
- Algumas funções do **Gerenciador de memória**:
  - Controlar quais as unidades de memória estão ou não estão em uso, para que sejam alocadas quando necessário;
  - Liberar as unidades de memória que foram desocupadas por um processo que finalizou;
  - Tratar do Swapping entre memória principal e memória secundária.
    - Transferência temporária de processos residentes na memória principal para memória secundária.

Execução de um Programa (1)



Execução de um Programa (2)



Código absoluto:

- Endereços relativos ao início da memória (endereços reais)
- Programas exclusivos para partições específicas na memória

Código relocável

- O programa pode ser carregado em qualquer posição da memória.
- Deve haver uma **tradução** de endereços (ou relocação de endereços)

Execução de um Programa (3)

- Relocação de Endereços
  - Estática
    - O Loader (em tempo de carga) reloca os endereços das instruções relocávies (ex: JMP endx)
  - Dinâmica
    - Em tempo de execução
    - O processo pode ser movimentado dentro da memória física
    - Um hardware especial deve estar disponível para que funcione (MMU)

Execução de um Programa (4)

- Relocação de Endereços (cont.)

Executável: programa em **linguagem de máquina**

```
10000000:       3c051000        lui     a1,0x1000
10000004:       24a50a98        addiu   a1,a1,2712
10000008:       3c041000        lui     a0,0x1000
1000000c:       24840a98        addiu   a0,a0,2712
10000010:       3c1d1000        lui     sp,0x1000
10000014:       27bd3ffc        addiu   sp,sp,16380
10000018:       aca00000        sw      zero,0(a1)
1000001c:       00a4182a        slt     v1,a1,a0
10000020:       24a50004        addiu   a1,a1,4
10000024:       1460fffc        bnez    v1,10000018 <_entry+0x18>
10000028:       00000000        nop
1000002c:       3404e100        li      a0,0xe100
10000030:       0c0000a4        jal     10000290 <_etext>
10000034:       00000000        nop
10000038:       0c00023d        jal     100008f4 <app_main>
1000003c:       00000000        nop
```

Espaço de Endereçamento Lógico

Tradução

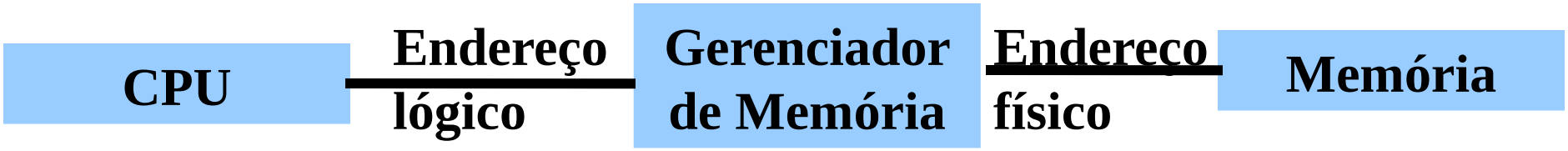
Espaço de Endereçamento Físico

- Conjunto de endereços reais

Gerência de Memória

Memória Lógica - é aquela que o processo enxerga, o processo é capaz de acessar.

Memória Física - é aquela implementada pelos circuitos integrados de memória, pela eletrônica do computador (memória real)

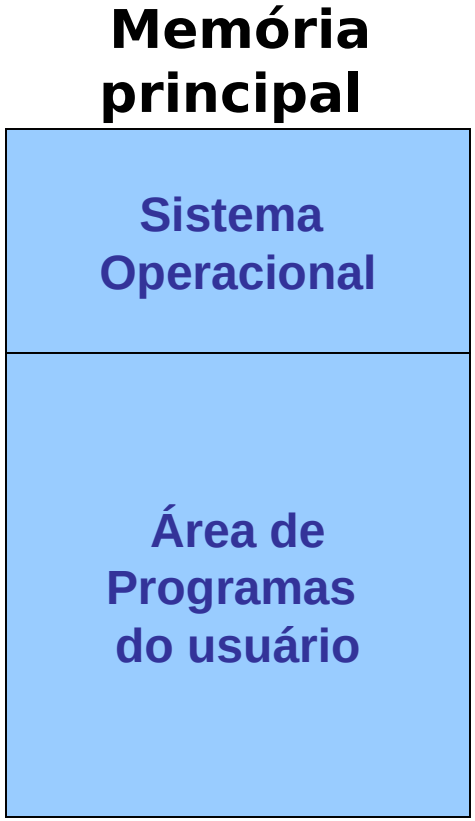


Técnicas de Gerência de Memória Real

- Alocação Contígua Simples
- Alocação Particionada
  - Partições Fixas
    - Alocação Particionada Estática;
  - Partições Variáveis
    - Alocação Particionada Dinâmica.

Alocação Contígua Simples (1)

- Alocação implementada nos primeiros sistemas e ainda usada nos monoprogramáveis;
- A Memória é dividida em duas áreas:
  - Área do Sistema Operacional
  - Área do Usuário
- Um usuário não pode usar uma área maior do que a disponível;
- Sem proteção:
  - Um usuário pode acessar a área do Sistema Operacional.

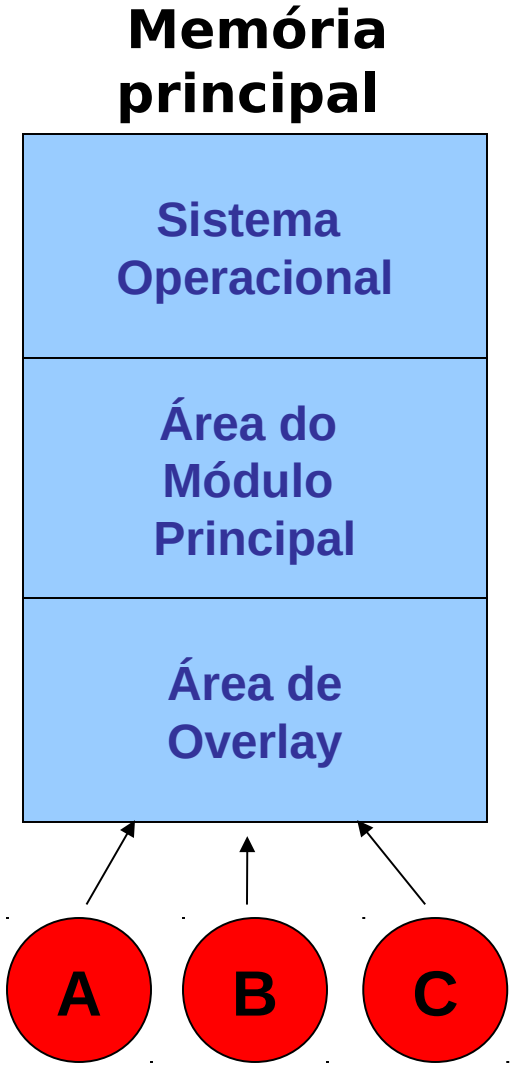


Alocação Contígua Simples (2)

- Registrador de proteção delimita as áreas do sistema operacional e do usuário;
- Sistema verifica acessos à memória em relação ao endereço do registrador;
- A forma de alocação era simples, mas não permitia utilização eficiente de processador e memória;

Alocação Contígua Simples (3)

- Programas de usuário limitados pelo tamanho da memória principal disponível.
- Solução: Overlay
  - Dividir o programa em módulos;
  - Permitir execução independente de cada módulo, usando a mesma área de memória;
- Área de Overlay
  - Área de memória comum onde módulos compartilham mesmo espaço.



Alocação Particionada

- Multiprogramação.
  - Necessidade do uso da memória por vários usuários simultaneamente.
- Ocupação mais eficiente do processador;
- A memória foi dividida em pedaços de tamanho fixo chamados partições;
- O tamanho de cada partição era estabelecido na inicialização do sistema;
- Para alteração do particionamento, era necessário uma nova inicialização com uma nova configuração.

Alocação Particionada Estática (1)

- Partições fixas
  - Tamanho fixo ; número de partições fixo

Alocação Particionada Estática Absoluta:

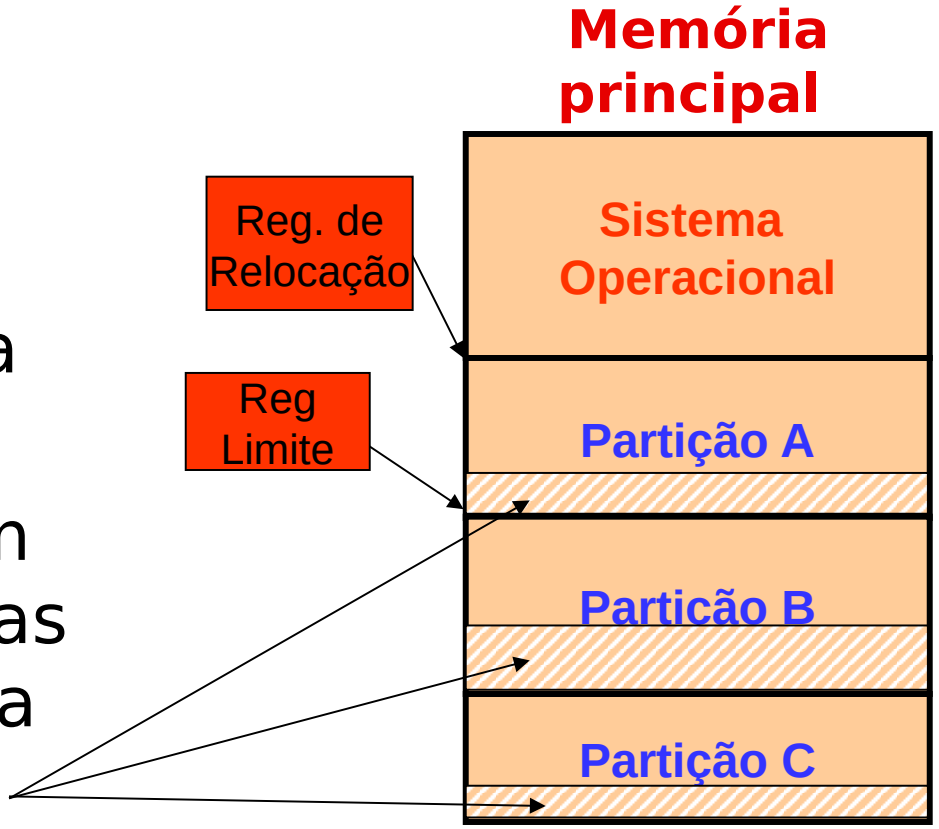
- Compiladores gerando código absoluto;
- Programas exclusivos para partições específicas.
- Simples de gerenciar
- E se todos os processos só pudessem ser executados em uma mesma partição (mesmo endereço base?)

Alocação Particionada Estática Relocável:

- Compiladores gerando código relocável;
  - Endereços relativos ao início da partição;
- Programas podem rodar em qualquer partição.

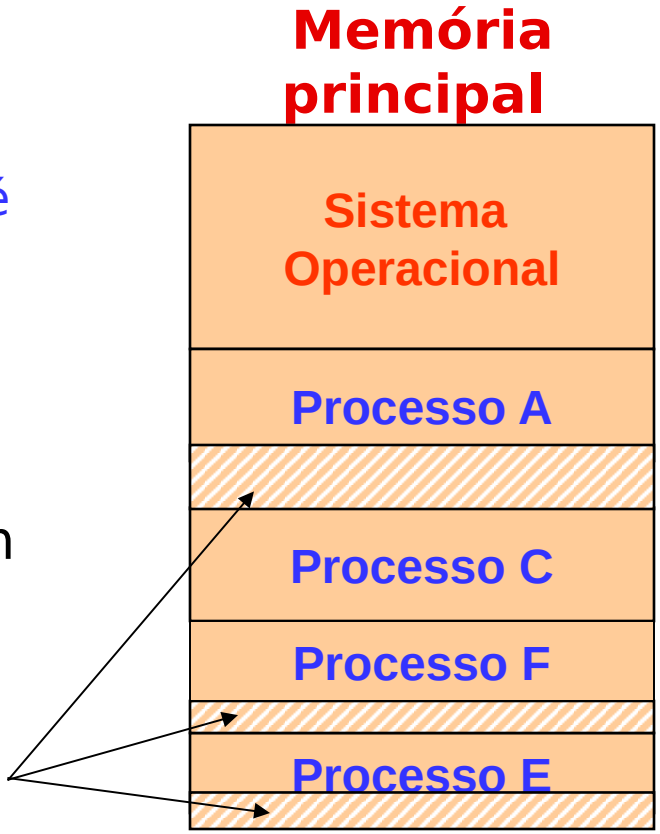
Alocação Particionada Estática (2)

- Proteção:
  - Registradores com limites inferior e superior de memória acessível.
- Programas não ocupam totalmente o espaço das partições, gerando uma fragmentação interna.

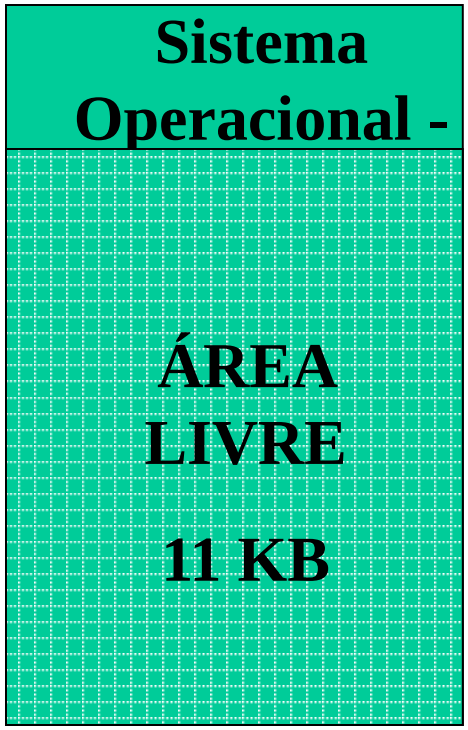


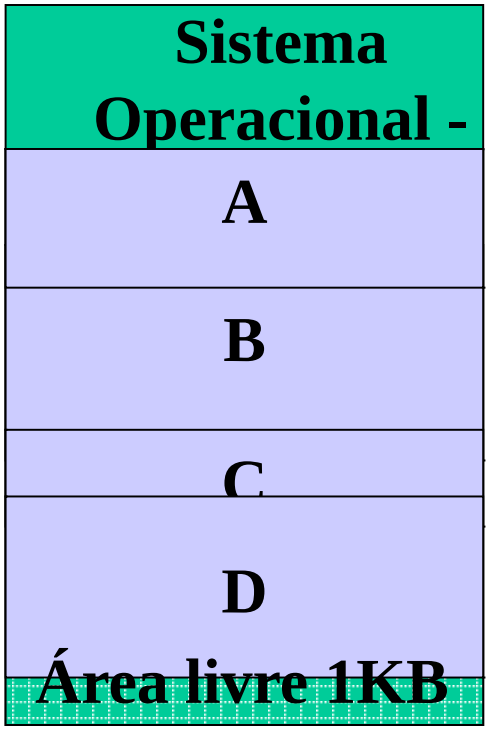
Alocação Particionada Dinâmica (1)

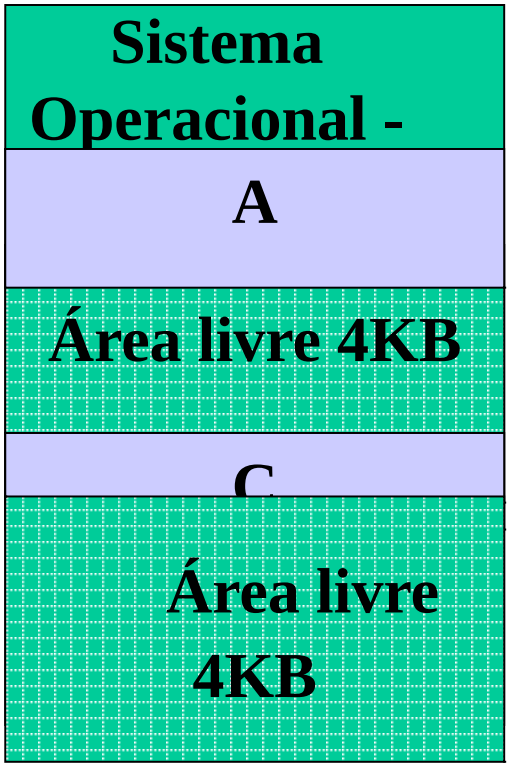
- Não existe realmente o conceito de partição dinâmica.
  - O espaço utilizado por um programa é a sua partição.
- Não ocorre fragmentação interna.
  - o tamanho da memória alocada é igual ao tamanho do programa
- Ao terminarem, os programas deixam espalhados espaços pequenos de memória, provocando a fragmentação externa.
  - os fragmentos são pequenos demais para serem reaproveitados

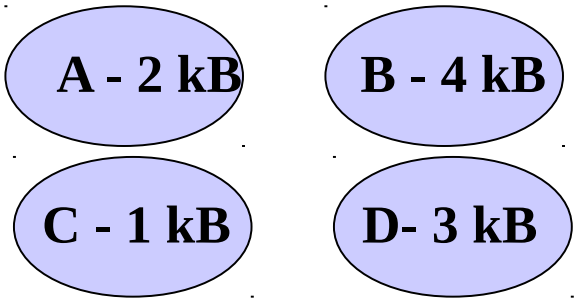


Alocação Particionada Dinâmica (2)





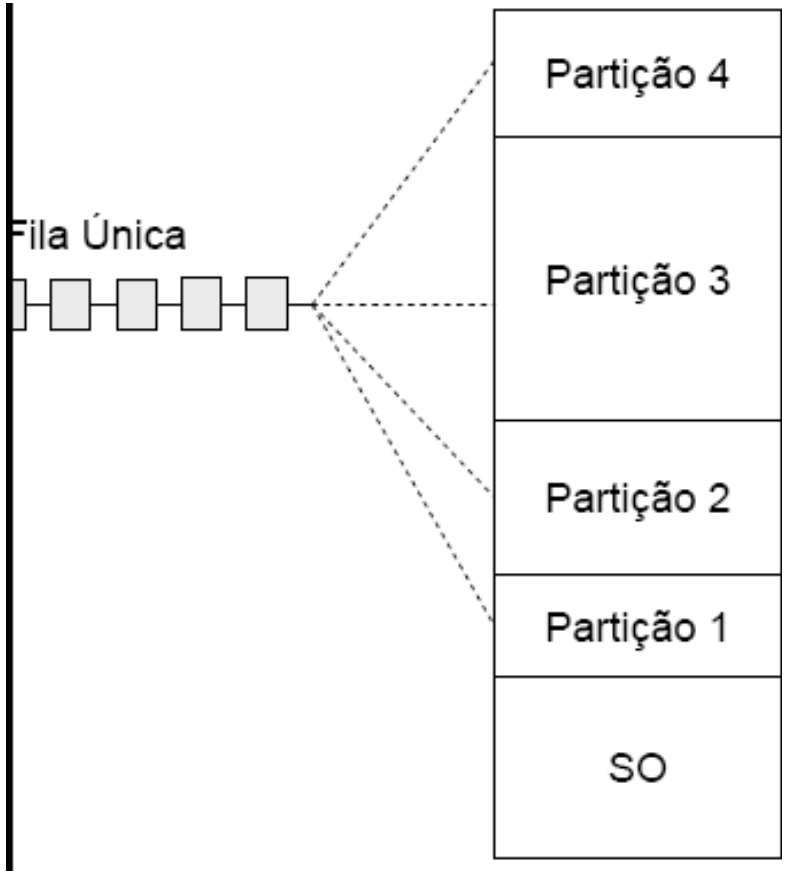




Alocação Particionada Dinâmica (3)

- Soluções:
  - Reunião dos espaços contíguos.
  - Realocar todas as partições ocupadas eliminando espaços entre elas e criando uma única área livre contígua-> Relocação Dinâmica de endereços:
    - Movimentação dos programas pela memória principal.
    - Resolve o problema da fragmentação.
    - Consome recursos do sistema
      - Processador, disco, etc.

Alocação Particionada Dinâmica (4)



- A multiprogramação implica em um problema
  - Ao mudar de partição o programa necessita ser relocado
- Relocação implica em correção de endereços de instruções
  - Via software (mapa de correções)
  - Via hardware (reg. base e limite)
- Proteção
  - Não correção ou correção errada implica em acesso a outra partição

Alocação Particionada Dinâmica (5)

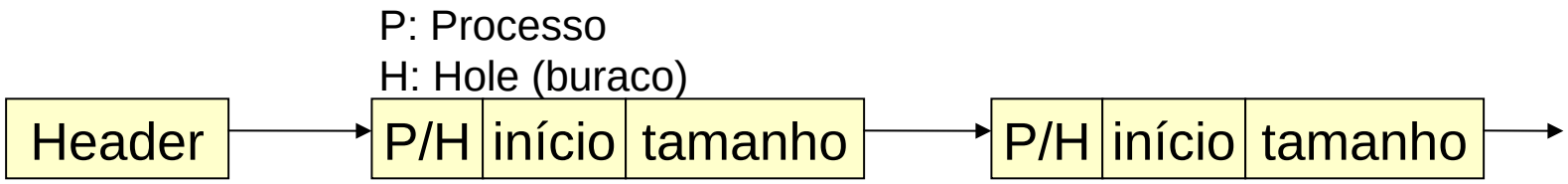
- Definição do tamanho das partições pode ser difícil
  - Processos crescem quando em execução
  - É bom definir áreas extras para dados e pilhas
- Como gerenciar as partições alocáveis de memória
  - Mapamento de bits
  - Mapeamento da Memória com listas encadeadas

Mapa de bits

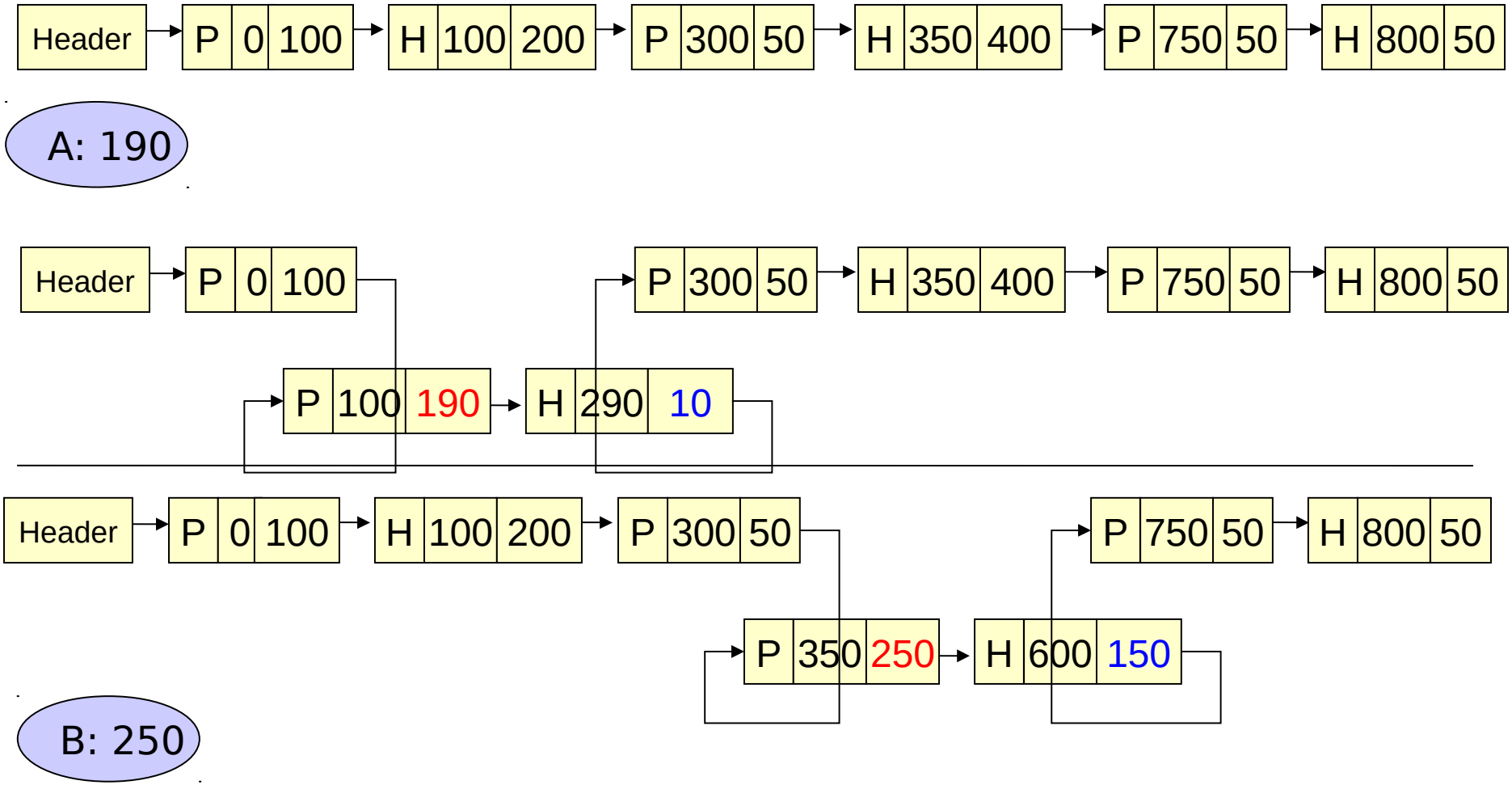
- Usado para o gerenciamento com alocação dinâmica
- Memória é dividida em unidades de alocação
  - De algumas palavras a vários kilobytes
    - Qto menor → maior o mapa de bits
    - Qto maior → desperdício na última unidade
- A cada unidade é associado um bit que descreve a disponibilidade da unidade
  - Disponível = 0
  - Ocupada = 1
- Principal problema
  - Busca de k zeros consecutivos para alocação de k unidades
  - Raramente é utilizado atualmente.
    - É muito lenta

Mapeamento da Memória com lista encadeada

- Também usado para gerenciar a alocação dinâmica.
- Lista ligada de segmentos alocados ou livres
- Um segmento é uma área de memória alocada ou livre
- Cada elemento da lista indica
  - Estado do segmento (P) Alocado por um processo ou (H) Livre
  - Unidade em que inicia
  - Tamanho em unidades
- Lista duplamente encadeada facilita de concatenação de segmenos
- Lista ordenada por endereço permite vários algoritmos de alocação



Mapeamento da Memória com lista encadeada



A escolha da partição ideal (1)

- Existem 4 maneiras de percorrer a lista de espaços livre atrás de uma lacuna de tamanho suficiente, são eles:
  - Best-fit (utiliza a lacuna que resultar a menor sobra)
    - Espaço mais próximo do tamanho do processo;
    - Tempo de busca grande;
    - Provoca fragmentação.
  - Worst-Fit (utiliza a lacuna que resultar na maior sobra):
    - Escolhe o maior espaço possível;
    - Tempo de busca grande;
    - Não apresenta bons resultados.

A escolha da partição ideal (2)

- First-Fit (primeira alocação):
  - utiliza a primeira lacuna que encontrar com tamanho suficiente
  - Melhor performance.
- Circular-fit ou Next-Fit (próxima alocação):
  - como first-fit mas inicia a procura na lacuna seguinte a última sobra
  - Performance inferior ao First-Fit.

A escolha da partição ideal (3)

- Considerações sobre Mapeamento da Memória com listas ligadas :
  - Todos melhoram em performance se existirem listas distintas para processos e espaços, embora o algoritmo fique mais complexo.
  - Listas ordenadas por tamanho de espaço melhoram a performance.