

Gerência de Memória

- Multiprogramação
 - Vários processos na memória
 - Alociação eficiente
 - Número máximo de processos
- Funções da GM
 - Alocar memória p/ novos processos
 - Liberar memória ocupada por processos que terminaram
 - Fazer troca de processos (swapping e paginação)
 - quantidade de MP é insuficiente

Estratégias de gerência: classificação

- Estratégias de busca
 - quando transferir um processo /partes dele p/ MP (sob demanda, antecipada)
 - Estratégias de posicionamento
 - onde novos processos/partes dele serão alocados (primeiro, melhor ou pior encaixe)
 - Estratégias de substituição
 - quais processos/partes dele devem ser substituídos (aleatória, mais antigo, mais ocioso, não usado mais recentemente etc.)
-

Estratégias de gerência

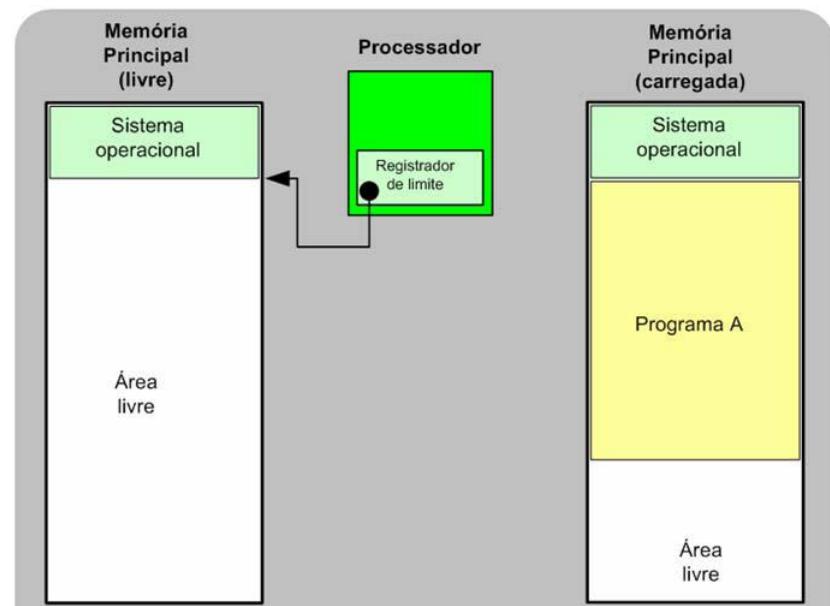
- Consideração inicial:
 - um programa tem que estar totalmente carregado na MP em um espaço contíguo
 - mais tarde, em memória virtual, essa limitação será relaxada
- GM controla:
 - particionamento da MP
 - o que está alocado
 - o que está livre

Alocação e controle

- Particionamento da MP pode ser de 3 formas:
 - **Alocação contígua simples**
 - usa toda a memória livre como um bloco único
 - **Alocação contígua particionada estática**
 - usa vários blocos de tamanhos pré-definidos
 - **Alocação contígua particionada dinâmica**
 - não usa o conceito de blocos, vai alocando os processos de forma adjacente

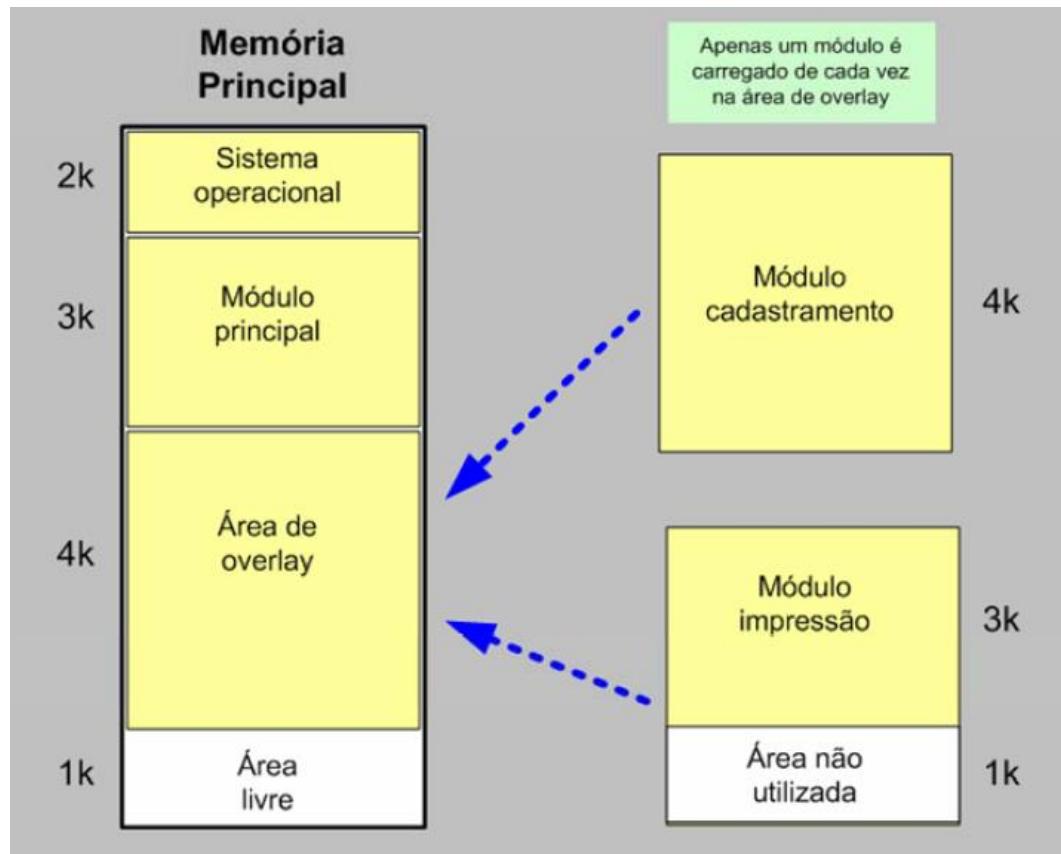
Alocação contígua simples (1)

- Primeiros SO's
 - sistemas monoprogramados
- Simples, fácil implementação
- MP é dividida em 2 partições:
 - SO
 - Processo de usuário
- Proteção
 - Registradores base e limite



Alocação contígua simples (2)

- Ultrapassar limite de memória imposto ???
 - *Overlay*

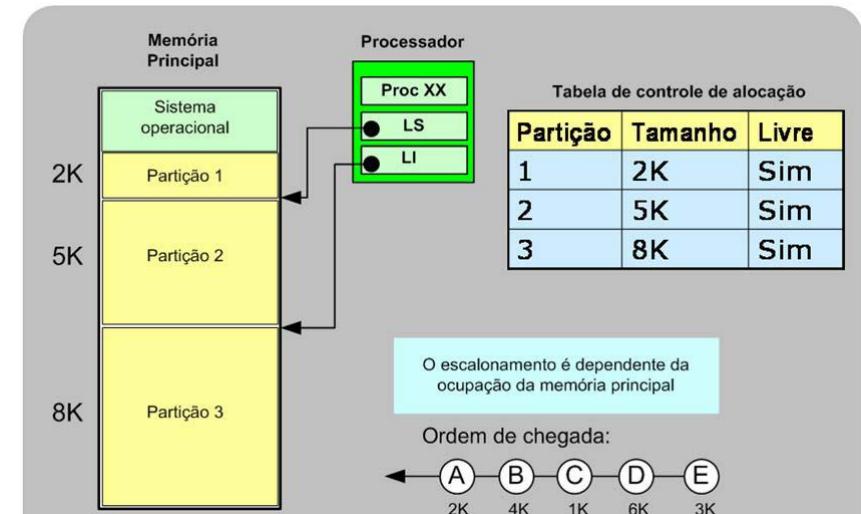


Alocação contígua particionada

- Sistemas multiprogramados
- SO precisa manter informações sobre:
 - Partições alocadas
 - Partições livres
 - Tamanho das partições
- **Estática (fixa)**: divide a memória em N partições fixas
- **Dinâmica (variável)**: divide a memória em N partições dinâmicas

Alocação contígua particionada estática (1)

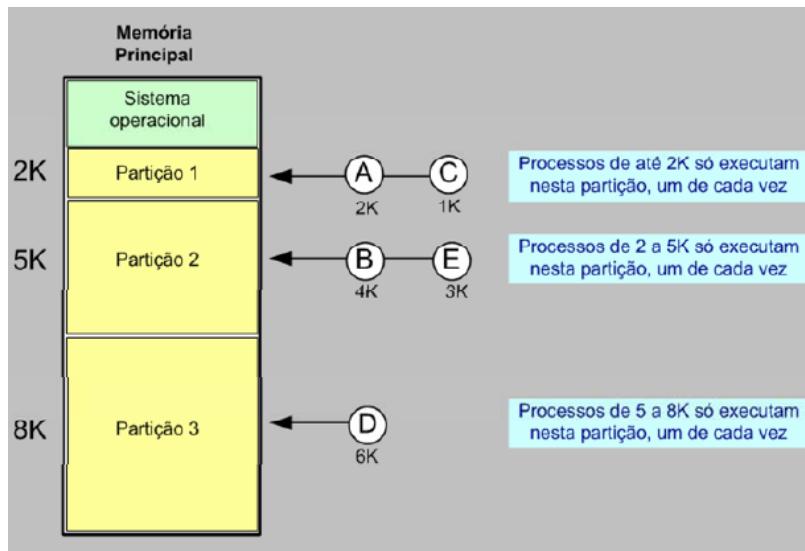
- **Partições fixas**
- Viabiliza multiprogramação, dividindo a memória em partições fixas:
 - tamanhos iguais OU tamanhos diferentes
- SO possui tabela de controle p/ gerir espaços alocados:
 - tamanhos dos espaços alocados
 - processos associados
 - espaços não alocados



Alocação contígua particionada estática (2)

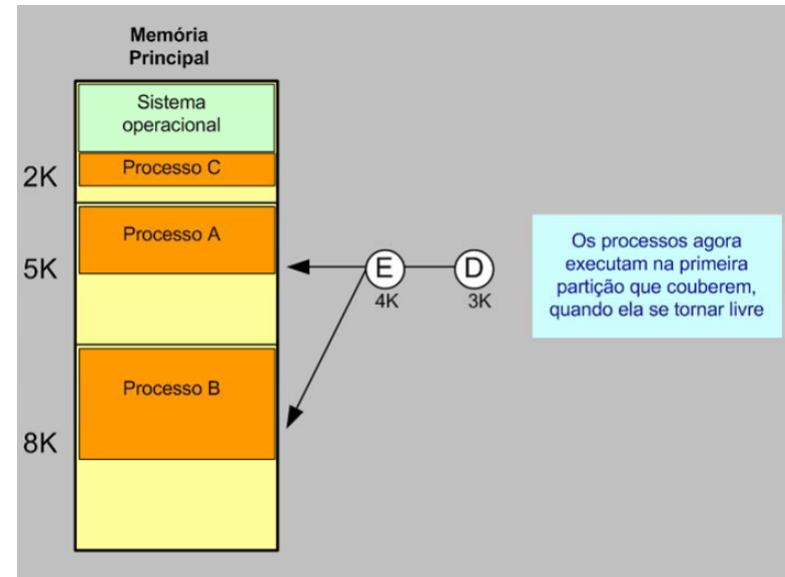
ABSOLUTA

- referências à MP baseadas em posições físicas, forçando sua execução em uma partição específica



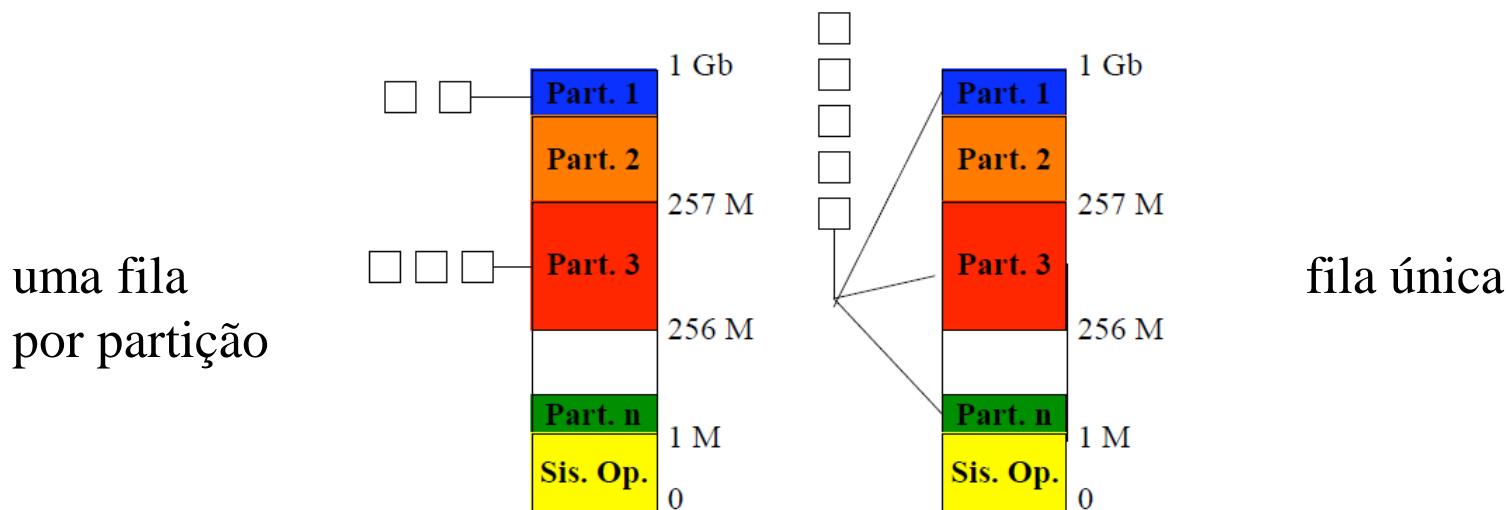
RELOCÁVEL

- referências à MP relativas apenas ao início do código, liberando sua execução em qualquer partição livre



Alocação contígua particionada estática (3)

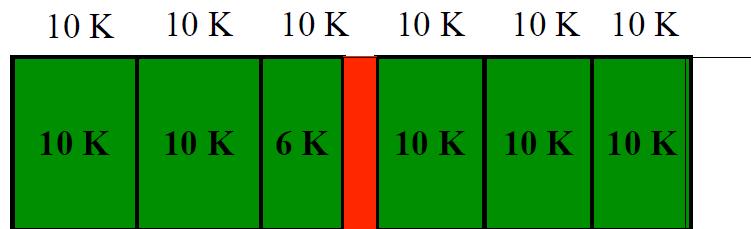
- Alocação: partições de tamanhos diferentes
 - Processo recebe a menor partição livre em que couber
 - Objetivo: minimizar fragmentação interna
 - 2 esquemas:



Alocação contígua particionada estática

(4)

- ☺ Implementação simples
- ☺ baixo overhead
- ☹ tamanho ideal p/ as partições ?
 - processos cujo uso de memória varia durante a execução (cresce/diminui)
- ☹ **fragmentação interna**
 - quando é alocada uma partição maior do que o necessário a um processo → há desperdício

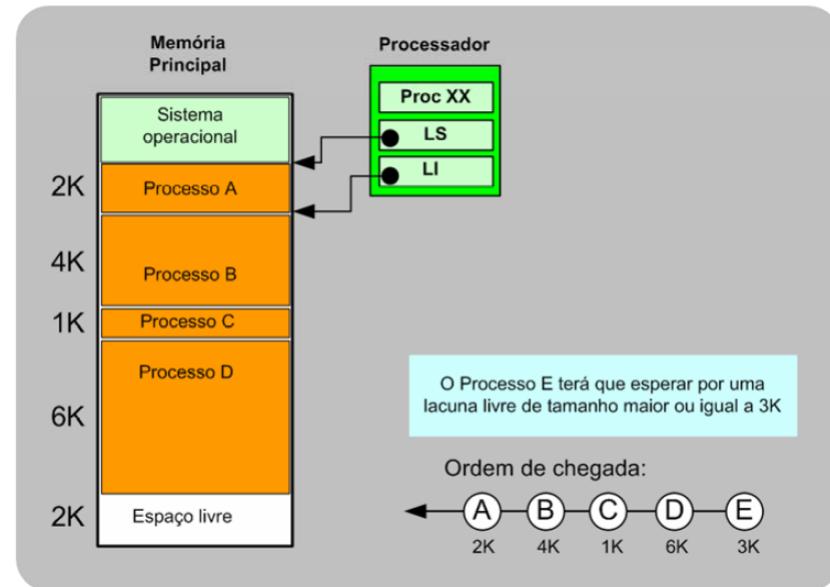


- Problema minimizado c/ uso de partições dinâmicas
-

Alocação contígua particionada dinâmica

(1)

- **Partições variáveis:** partições são em número e tamanho variáveis
- otimiza uso da memória – s/ fragmentação interna
 - início: toda a memória é considerada um bloco alocável (exceto área do SO)
 - blocos vão sendo alocados
 - processo encerra: libera o bloco que usou
 - gerência mais complexa
 - dificulta alocação e liberação de memória



Alocação contígua particionada dinâmica

(2)

- Gerência:
 - SO mantém uma lista “lacunas”
 - conforme a evolução dos processos, as lacunas vão se espalhando na memória
 - quando se aloca uma lacuna, o espaço não usado da mesma cria uma nova lacuna, menor
 - p/ alocar bloco: SO percorre a lista de lacunas
 - **Como percorrer a lista ????**

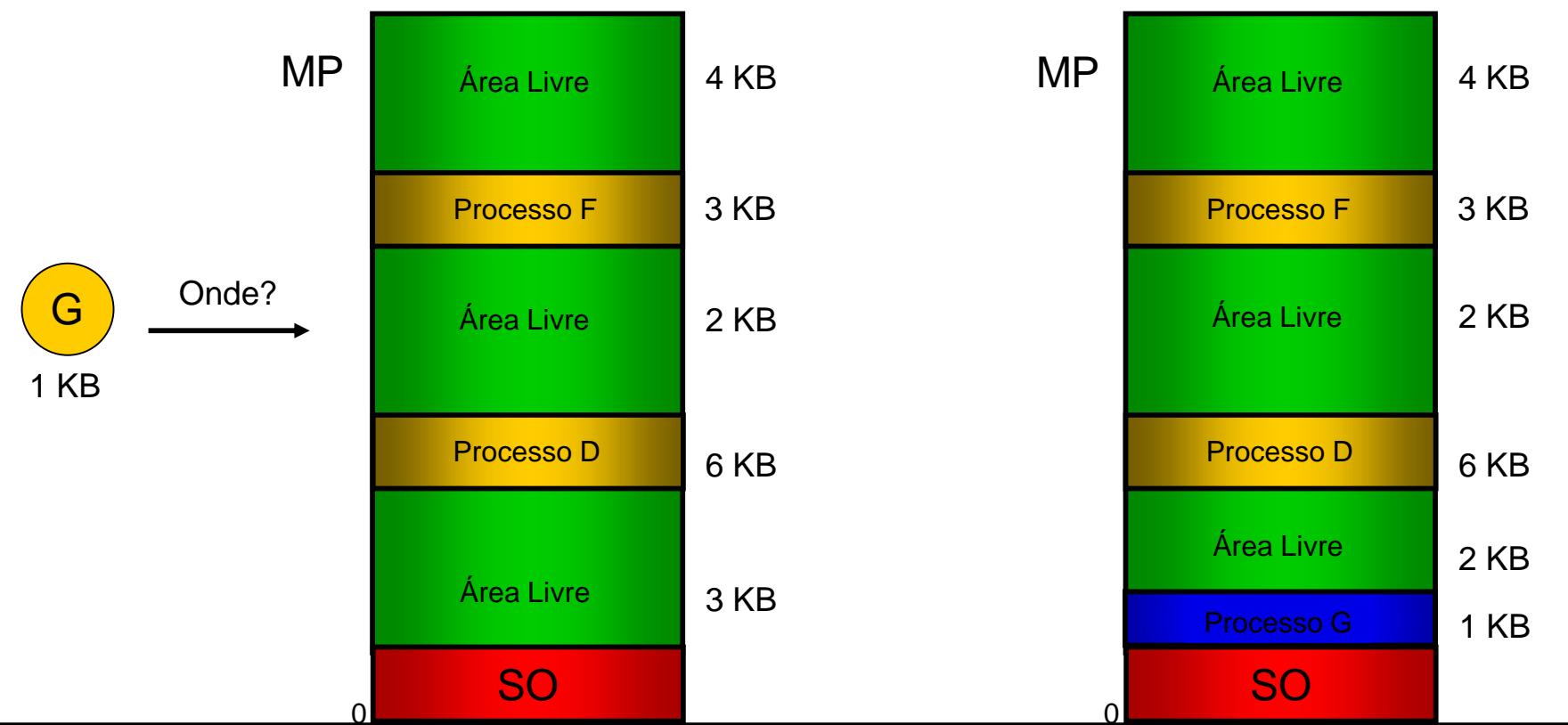
Alocação contígua particionada dinâmica: estratégias de alocação (3)

- qual lacuna alocar a um novo processo ?
 - varredura na tabela de espaços livres
- Estratégias de alocação:
 - **FIRST-FIT e FIRST-FIT CIRCULAR**
 - **BEST-FIT**
 - **WORST-FIT**

Estratégias first-fit e first-fit circular

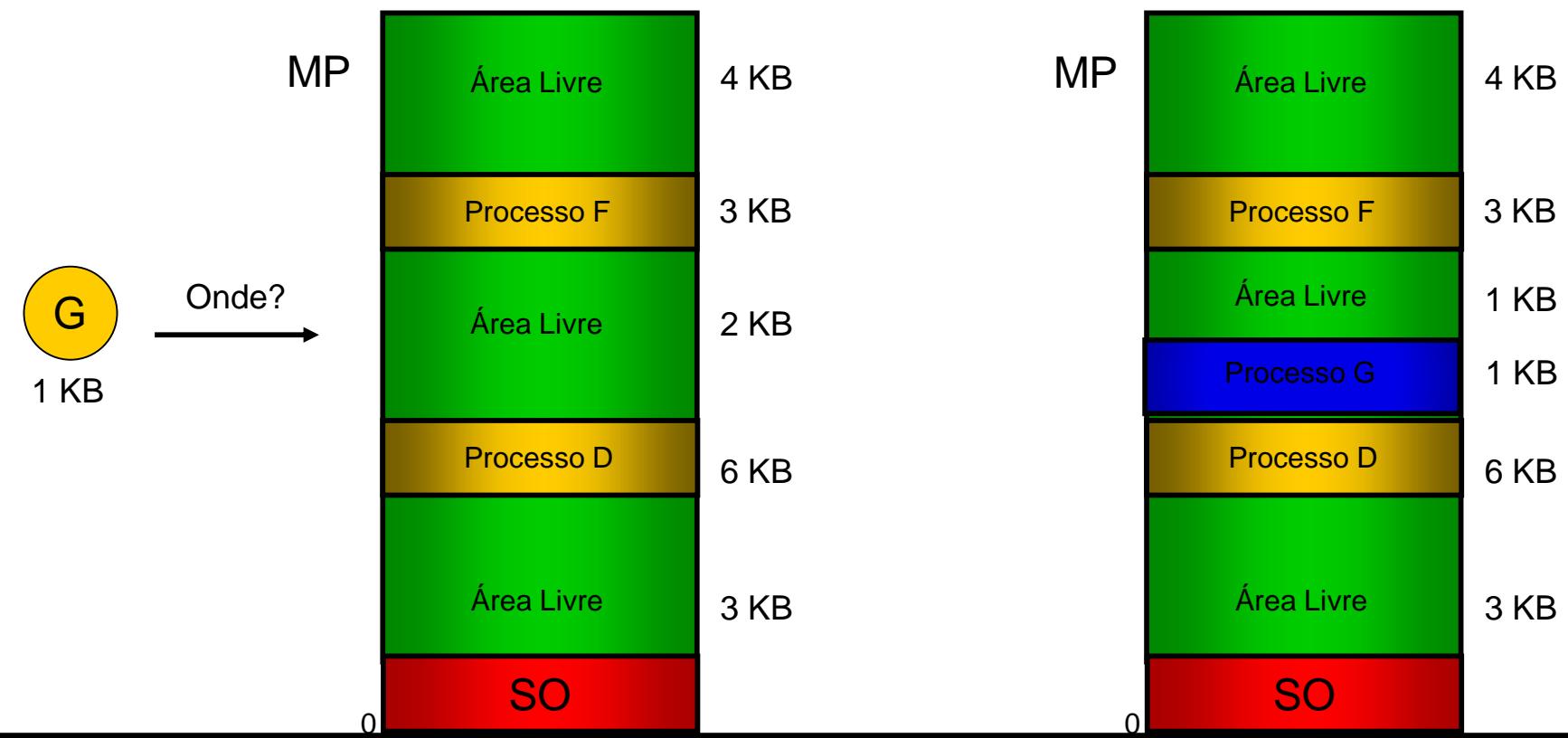
- **FIRST-FIT**: a primeira lacuna onde couber
 - bBusca: no início ou **a partir do último bloco alocado**

first-fit circular



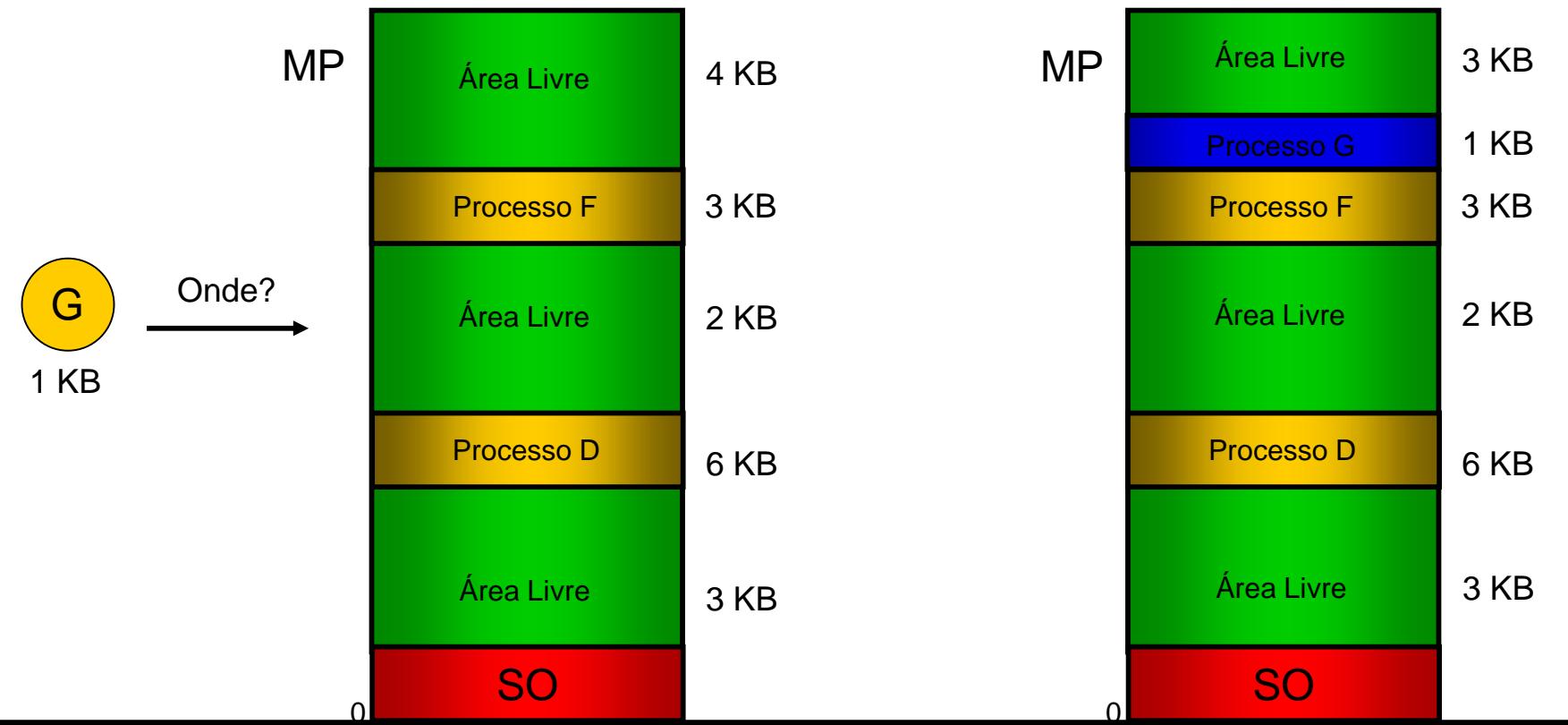
Estratégia best-fit

- **BEST-FIT:** a menor lacuna disponível
 - ideia: gerar a menor sobra



Estratégia worst-fit

- **WORST-FIT:** a maior lacuna disponível
 - ideia: gerar a maior sobra (reaproveitável ???)

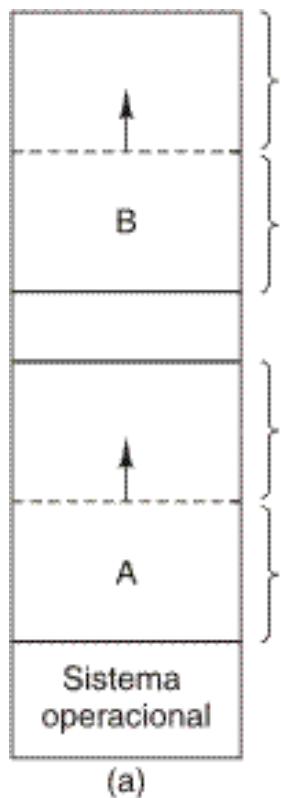


Alocação contígua particionada dinâmica: gerência do espaço livre (1)

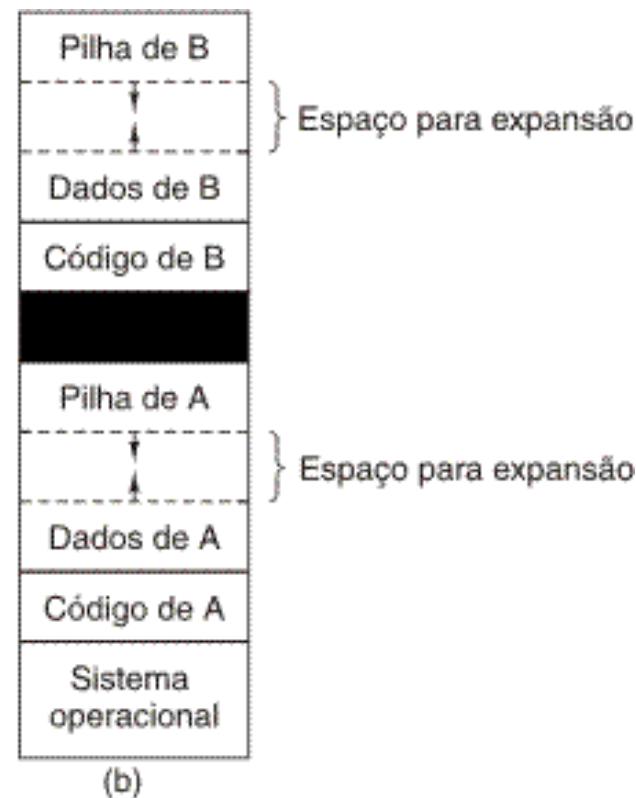
- alocação dinâmica de memória
 - Espaço p/ expansão
 - Mapa de bits (*bitmap*)
 - Lista encadeada

Alocação contígua particionada dinâmica: espaço para expansão (2)

- Pré-determinar espaço para expansão: inviável !!!



Espaço para expansão



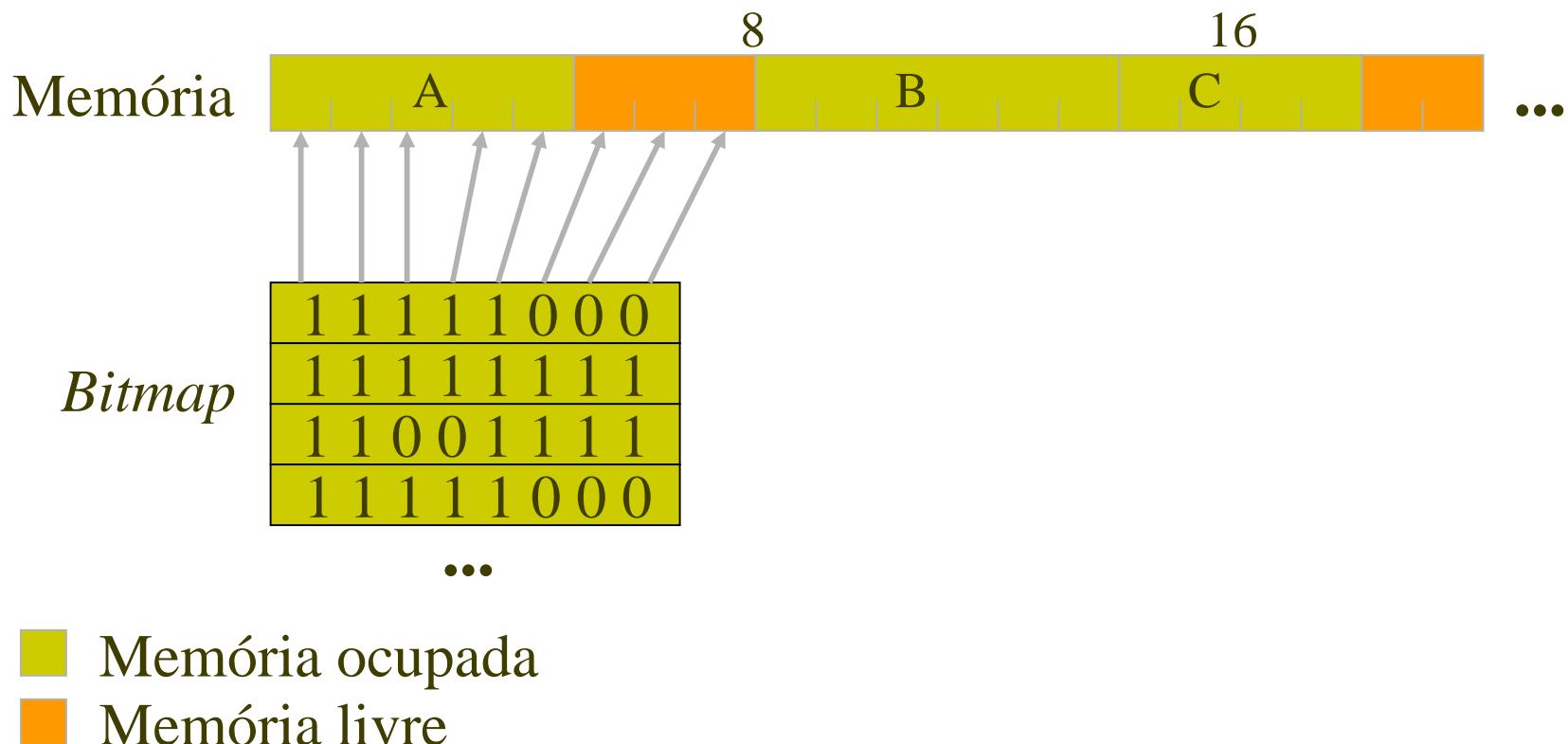
Espaço para expansão

Espaço para expansão

(b)

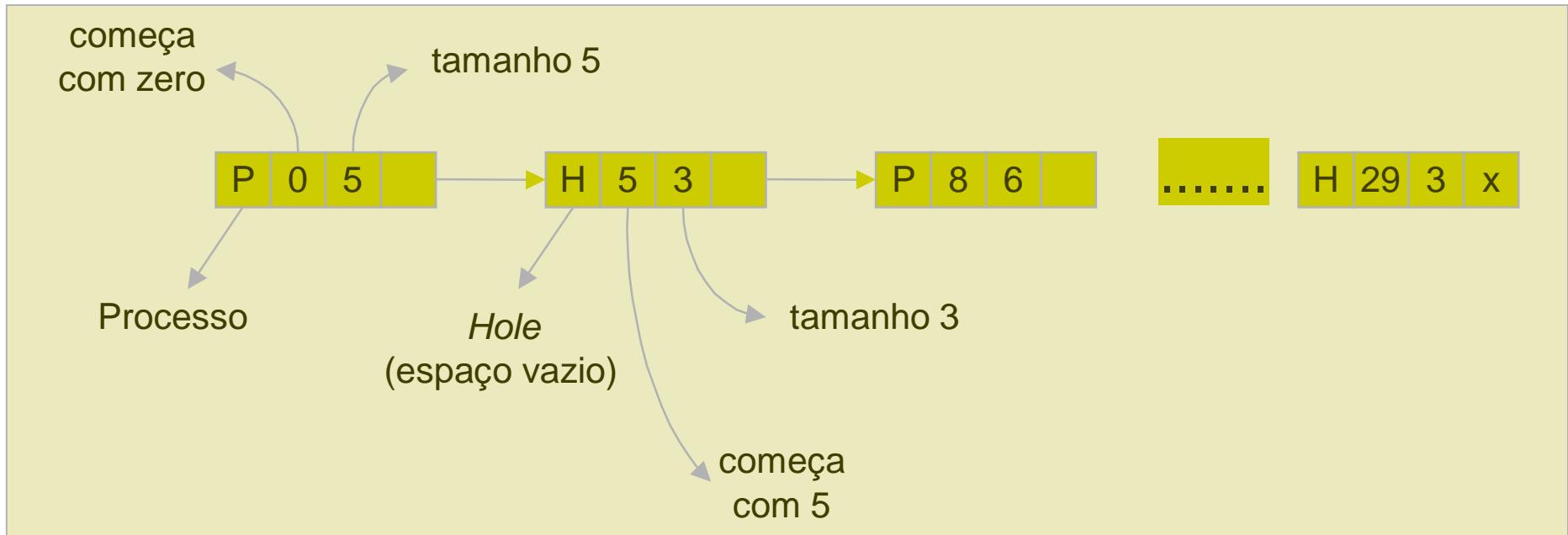
Alocação contígua particionada dinâmica: mapa de bits (3)

- cada unidade de alocação da memória corresponde a 1 bit do mapa (0 – livre; 1 – ocupada)



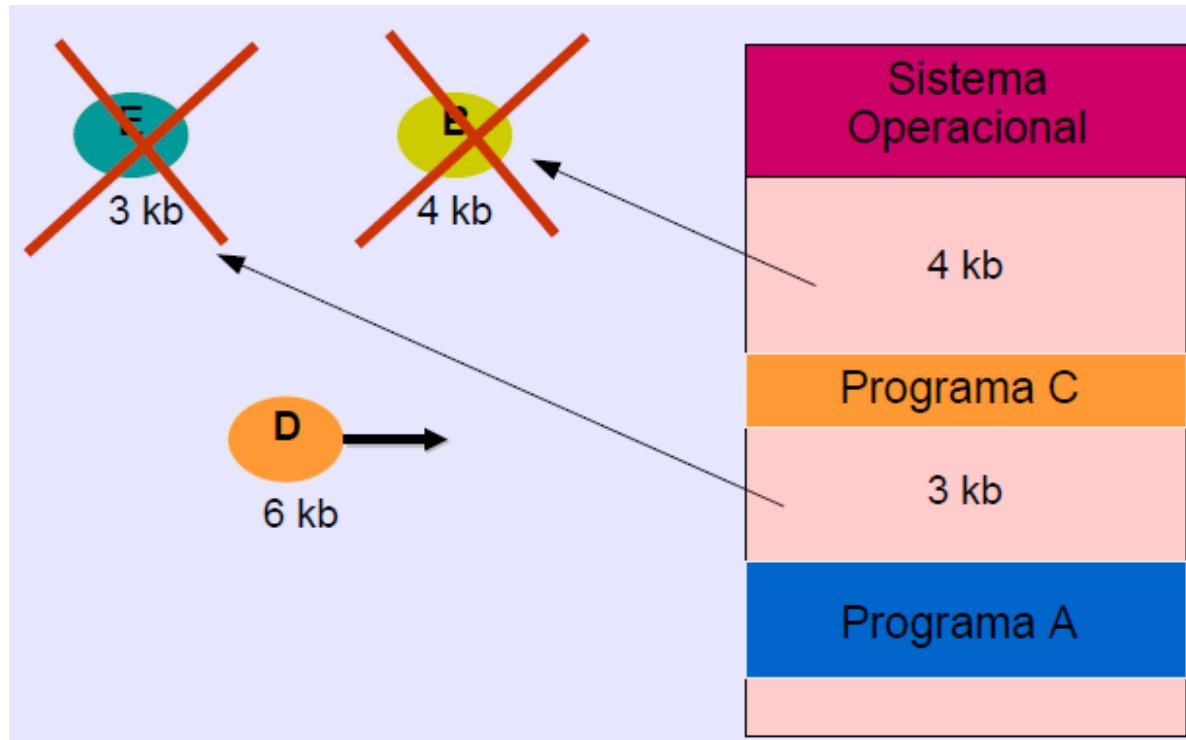
Alocação contígua particionada dinâmica: lista encadeada (4)

- uma lista para espaços livres e outra para espaços ocupados ou uma lista para ambos



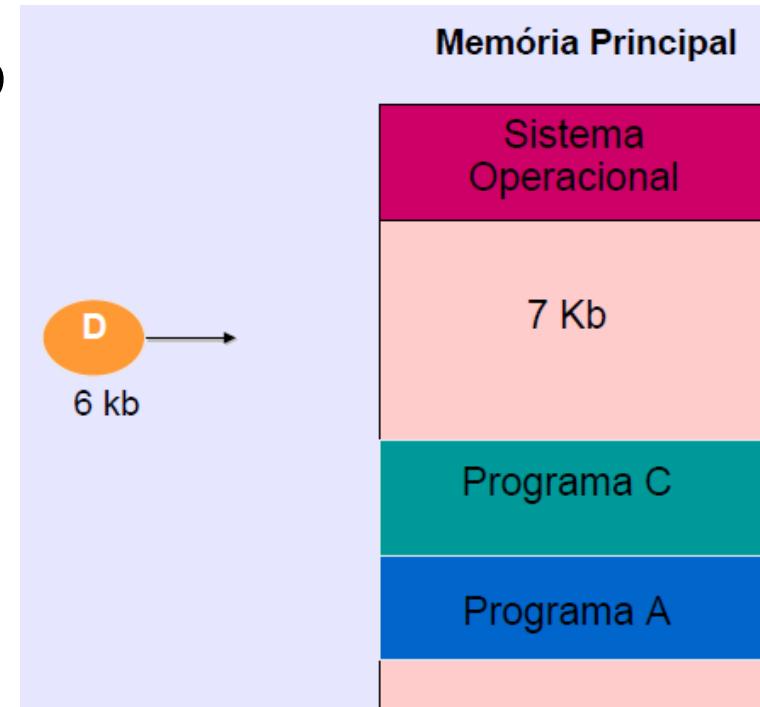
Alocação contígua particionada dinâmica: fragmentação externa (5)

- Formação de buracos
 - Fragmentação externa ainda existe



Alocação contígua particionada dinâmica: fragmentação externa – “soluções” (6)

- Coalescência
 - fusão de partições livres adjacentes
- Relocação ou Compactação
 - realoca as partições eliminando os espaços entre elas, criando uma área contígua



Alocação contígua particionada dinâmica: fragmentação externa – “soluções” (7)

- Outras: Permitir que o endereçamento lógico de um processo não seja contíguo
 - **Paginação**
 - Segmentação
 - Combinação de ambas as técnicas

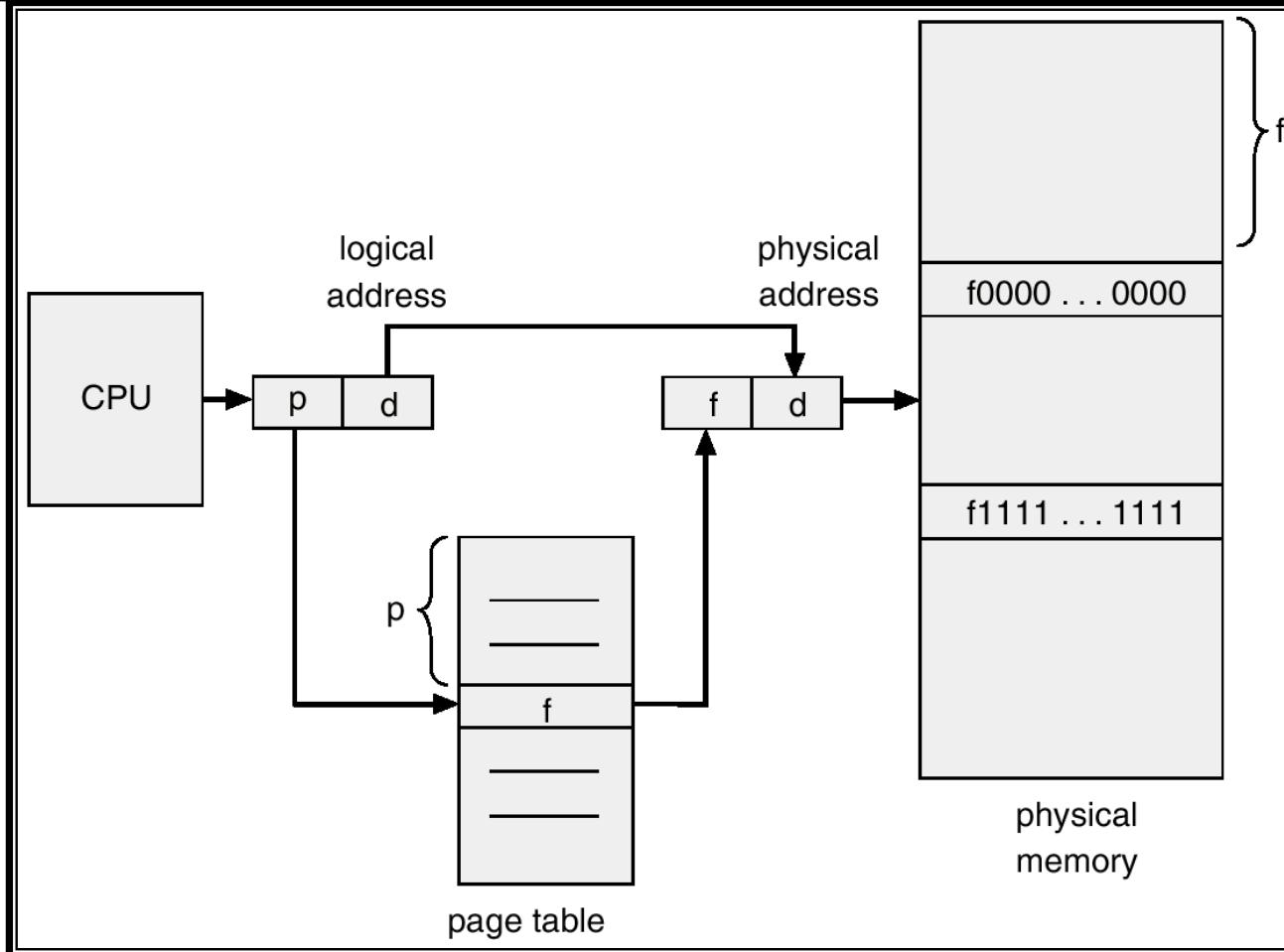
Paginação (1)

- Na verdade, o espaço de endereçamento lógico deve ser contíguo, o espaço físico não precisa ser
 - Paginação
 - Processo (memória lógica) é dividido em **páginas** (blocos de processos)
 - MP (memória física) é dividida em **frames** do mesmo tamanho
 - Visão do usuário: espaço de endereçamento contíguo
 - Visão do sistema: processo é “esparramado” na memória física
-

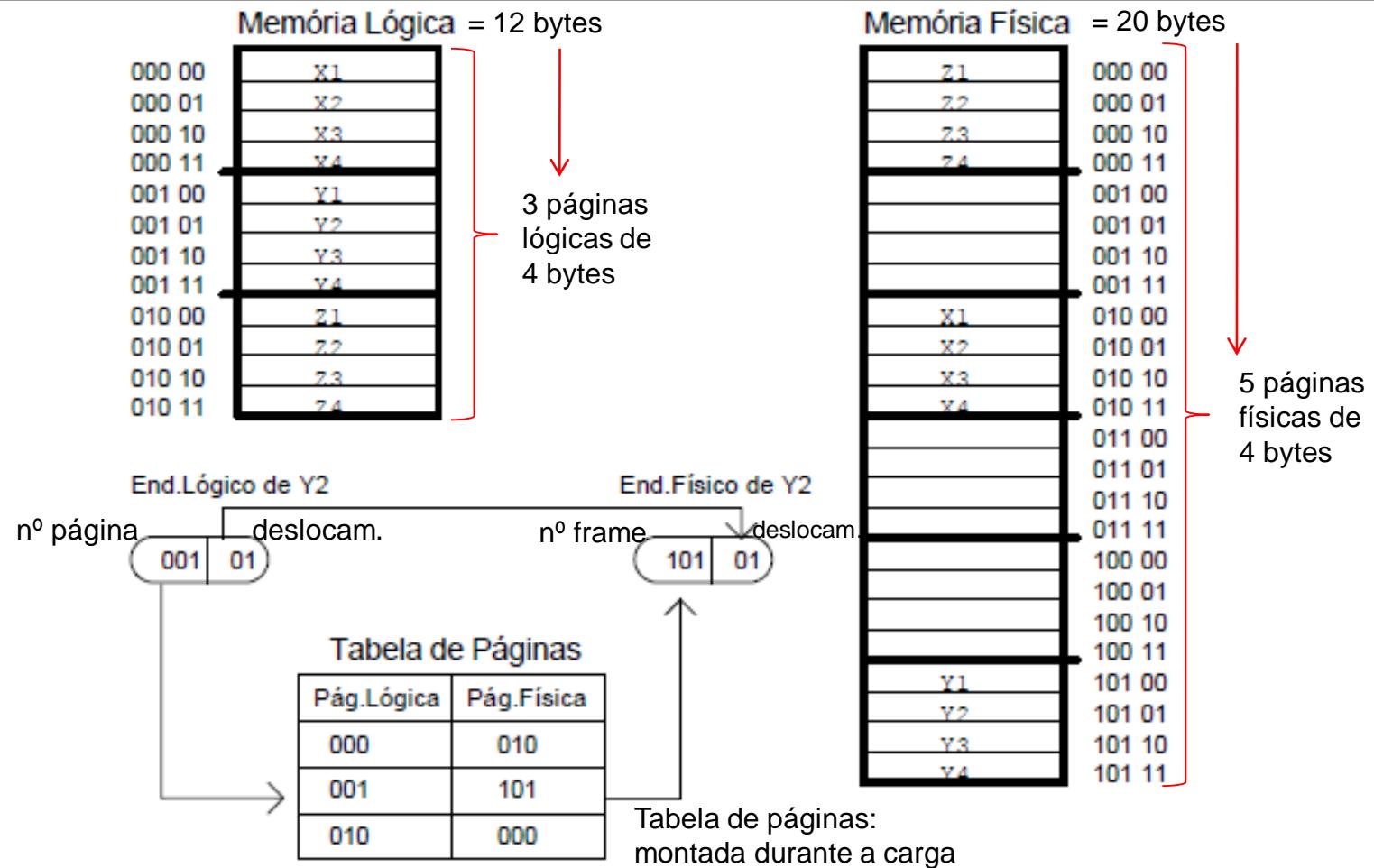
Paginação (2)

- mantém registro dos frames livres
 - executar processo com n páginas exige n frames livres na memória
 - páginas são carregadas em qualquer frame livre
 - Tabela de página (page table)
 - traduz endereço lógico (página) em físico (frame)
 - tamanho da página: imposto pelo hardware (MMU)
 - valores típicos: entre 1 e 8 kbytes
 - elimina fragmentação externa
 - reduz fragmentação interna (restrita à última página)
-

Arquitetura para Tradução de Endereços



Mecanismo básico de paginação



Gerência de páginas: proteção

- Bit de proteção: bit válido/inválido anexado a cada entrada na tabela de páginas
 - Válido – página \in espaço end. processo – página legal
 - Inválido – página \notin espaço end. processo

