Departamento de Computação

SISTEMAS OPERACIONAIS 1
21270 A

Departamento de Computação
Prof. Kelen Cristiane Teixeira Vivaldini

Departamento de Computação

Gerenciamento de Memória

- Recurso importante;
- Tendência atual do software
  - Lei de *Parkinson*: “Os programas se expandem para preencher a memória disponível para eles” (adaptação);
- Hierarquia de memória:
  - *Cache*;
  - Principal;
  - Disco;

Gerenciamento de Memória

- Idealmente os programadores querem uma memória que seja:
  - Grande
  - Rápida
  - Não Volátil
  - Baixo custo
- Infelizmente a tecnologia atual não comporta tais memórias

Hierarquia de Memória

- Hierarquia de Memórias:
  - Uma pequena quantidade de memória cache, volátil, muito rápida e de alto custo
  - Uma grande memória principal (RAM), volátil, com centenas de MB ou poucos GB, de velocidade e custo médios
  - Uma memória secundária, não volátil em disco, com gigabytes (ou terabytes), velocidade e custo baixos

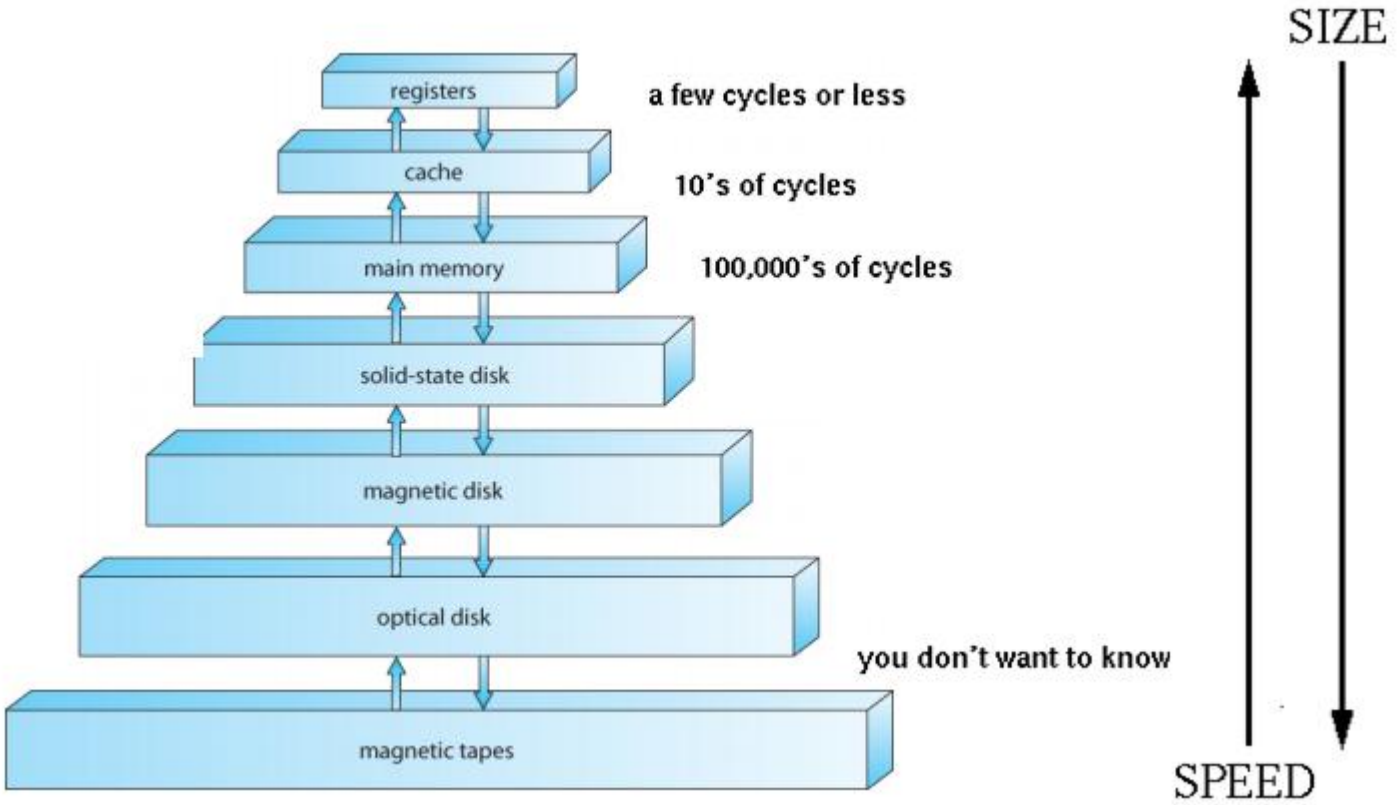
- O gerenciador de memória trata a hierarquia de memórias

Hierarquia de Memória

- **Cache**
  - Pequena quantidade
    - k bytes
  - Alto custo por byte
  - Muito rápida
  - Volátil
- **Memória Principal**
  - Quantidade intermediária
    - M bytes
  - Custo médio por byte
  - Velocidade média
  - Volátil
- **Memória secundária**
  - Grande quantidade –
    - G bytes
  - Baixo custo por byte
  - Lenta
  - Não volátil

Departamento de Computação

Hierarquia de Memória



Departamento de Computação

Gerenciamento de Memória

- Cabe ao **Gerenciador de Memória** gerenciar a hierarquia de memória

- Controlar as partes da memória que estão em uso e quais não estão, de forma a:
  - alocar memória aos processos, quando estes precisarem;
  - liberar memória quando um processo termina; e
  - tratar do problema do *swapping* (quando a memória é insuficiente).

Gerenciamento de Memória

- Para cada tipo de memória:
  - Gerenciar espaços livres/ocupados;
  - Alocar processos/dados na memória;
  - Localizar dado;

- Gerenciador de memória:
  - Responsável por alocar e liberar espaços na memória para os processos em execução;
  - Responsável por gerenciar chaveamento entre a memória principal e o disco, e memória principal e memória *cache*;

Alocação de Espaço e Técnicas de Gerenciamento

- Alocação contínua
  - máquinas antigas
  - malloc
- Alocação não-contínua
  - memória virtual

Gerenciamento Básico de Memória

- Sistemas de Gerenciamento de Memória, podem ser divididos em 2 classes:
  - Sistemas que, durante a execução levam e trazem processos entre a memória principal e o disco (troca de processos e paginação)
  - Sistemas mais simples, que não fazem troca de processos e nem paginação

Monoprogramação sem trocas de processos ou Paginação

Sistemas Mono-usuários

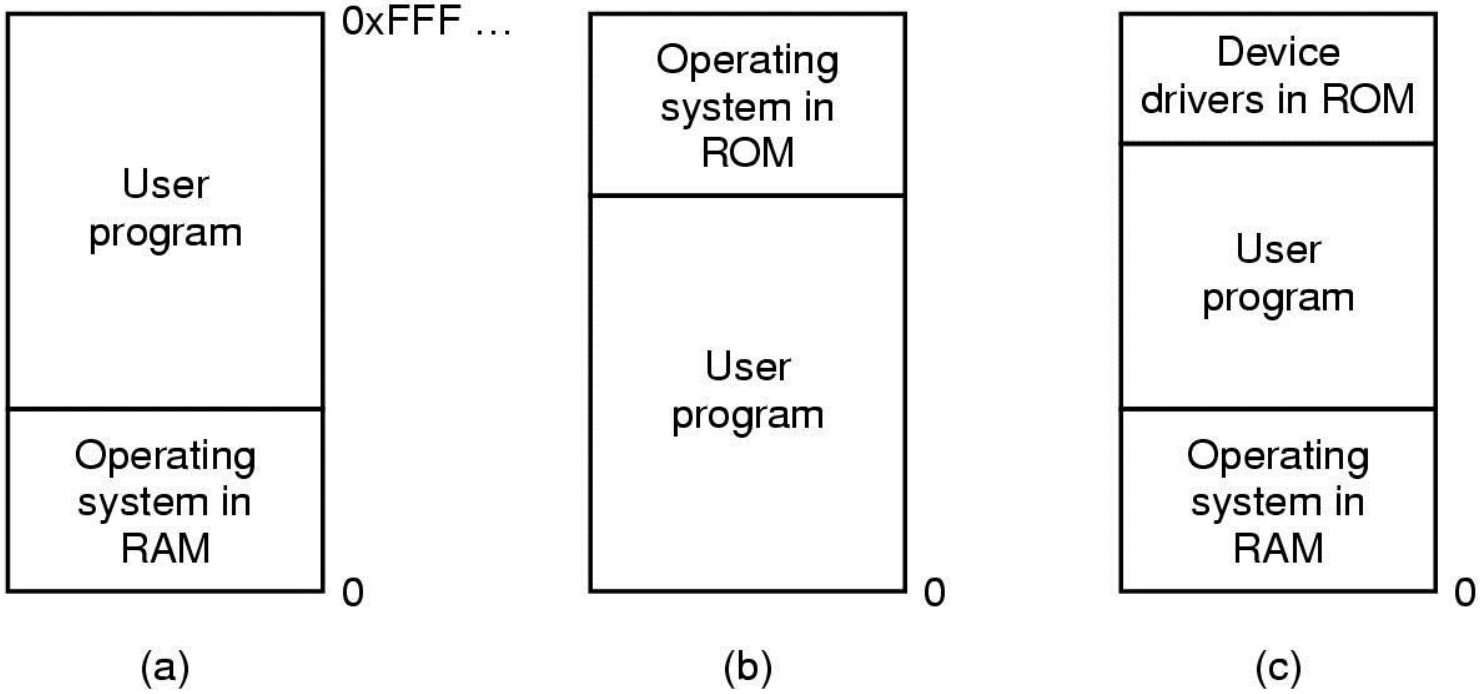
- O gerenciamento de memória é bem simples, pois toda a memória é alocada à próxima tarefa, incluindo a área do S.O.

- Erros de execução podem vir a danificar o S.O.

- Neste caso, a destruição do S.O. é um pequeno inconveniente, resolvido pelo recarregamento do mesmo.

Gerenciamento Básico de Memória

Monoprogramação sem trocas de processos ou Paginação



Três esquemas simples de organização de memória

- Um sistema operacional com um processo de usuário

(a) Maninframes

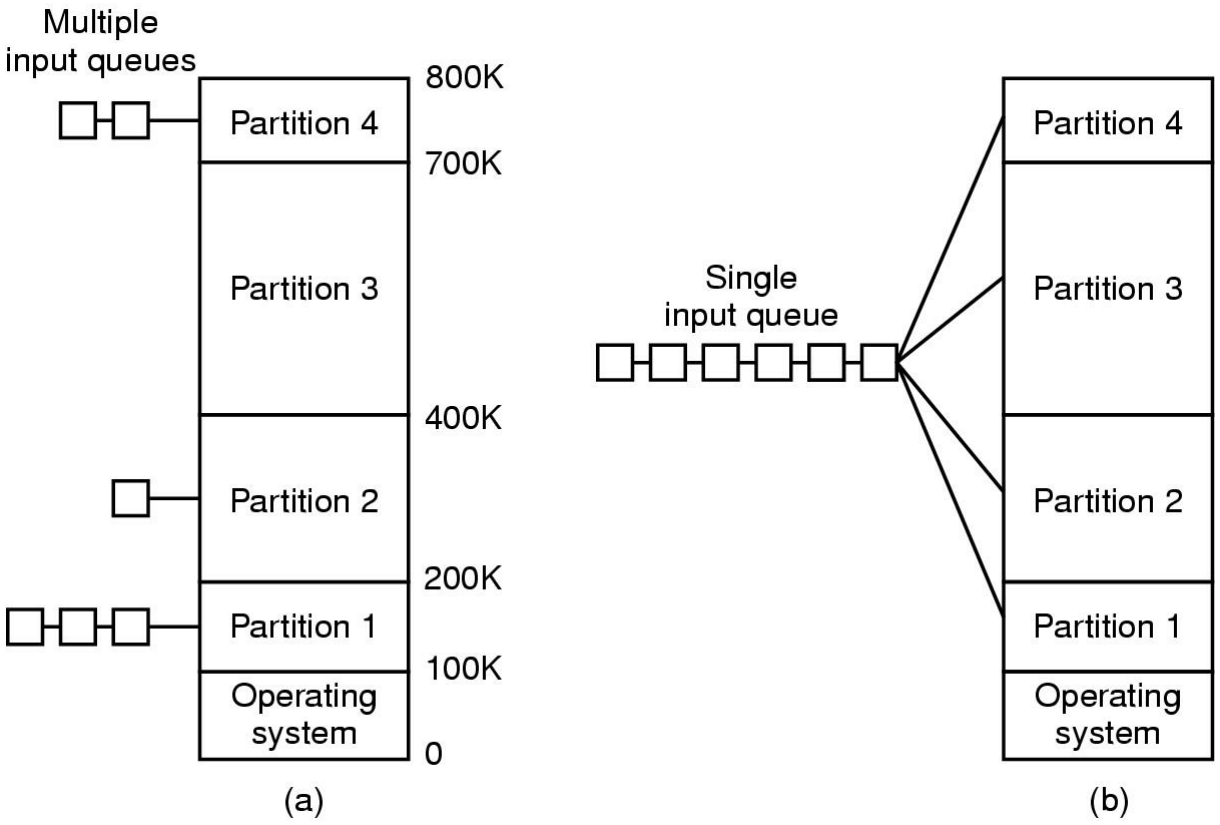
(b) palmtops

(c) MS-DOS

Multiprogramação com partições Fixas

- **Sistemas Monoprogramados**: raramente usados atualmente.
- **Sistemas modernos:** permitem multiprogramação
  - A maneira mais comum de realizar a multiprogramação é dividir simplesmente a memória em **n partições** (provavelmente de tamanhos diferentes).
  - Esta divisão pode ser feita de maneira manual, quando o sistema é inicializado
  - Ao chegar, um *job*, pode ser colocado em uma fila de entrada associada à menor partição, grande o suficiente para armazená-lo

Multiprogramação com partições Fixas



- Partições de memória fixa
  - fila separada para cada partição
  - uma única fila de entrada

Modelagem da Multiprogramação

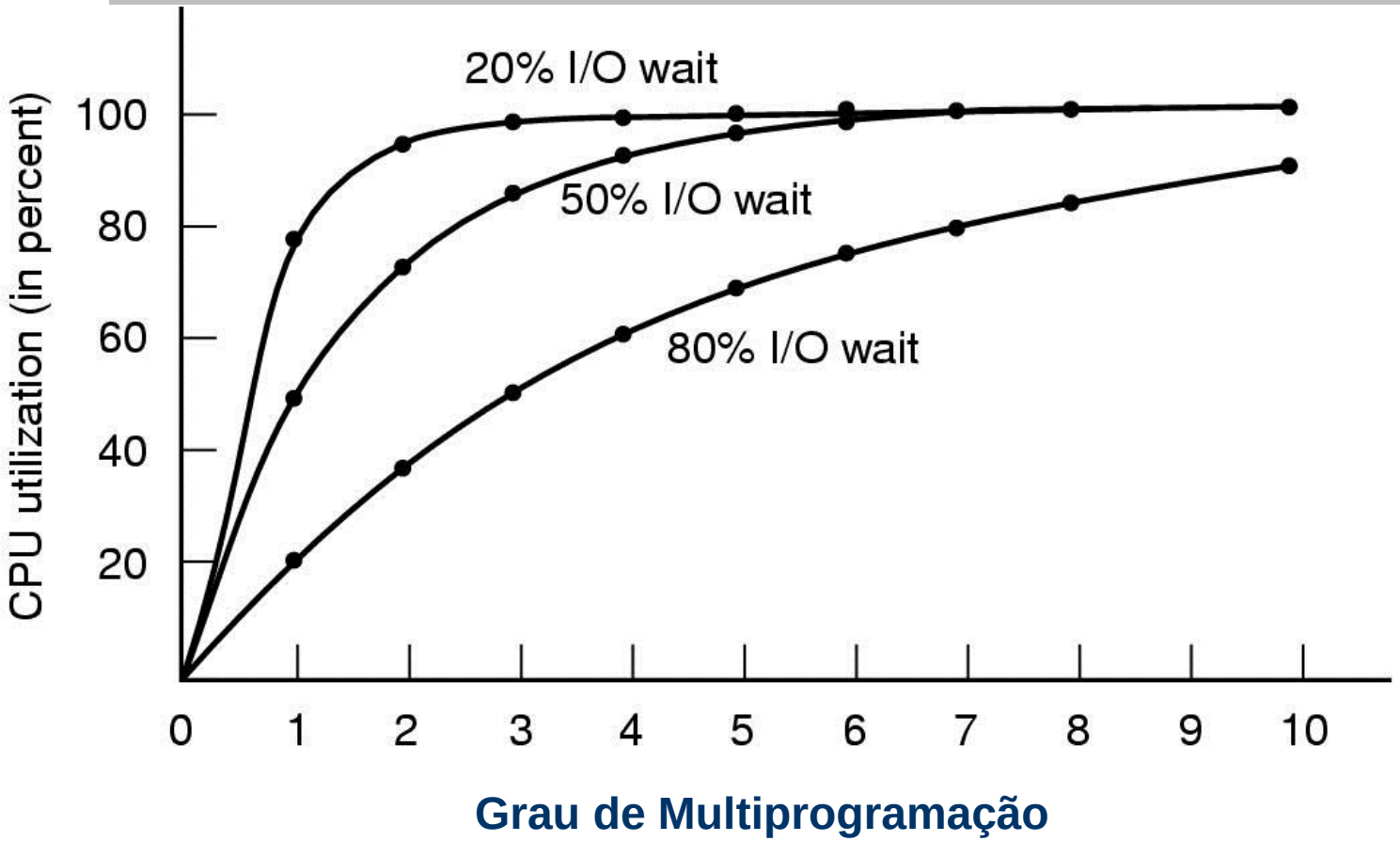
- O uso da Multiprogramação pode melhorar a utilização da UCP
- Considerando a utilização da UCP de modo probabilístico:

- Suponha que um processo gaste uma fração *p* de seu tempo esperando pela finalização de sua solicitação de E/S
- Com n processos simultâneos na memória, a probabilidade de todos os n processos estarem esperando por E/S (situação em que a UCP está ociosa) é p^n

- A utilização da UCP é dada pela fórmula:

$$\text{utilização da UCP} = 1 - p^n$$

Modelagem da Multiprogramação



A utilização da UCP como uma função do número de processos na memória

Modelagem da Multiprogramação

Job	Instante de chegada	Minutos necessários de CPU
1	10h00	4
2	10h10	3
3	10h15	2
4	10h20	2

(a)

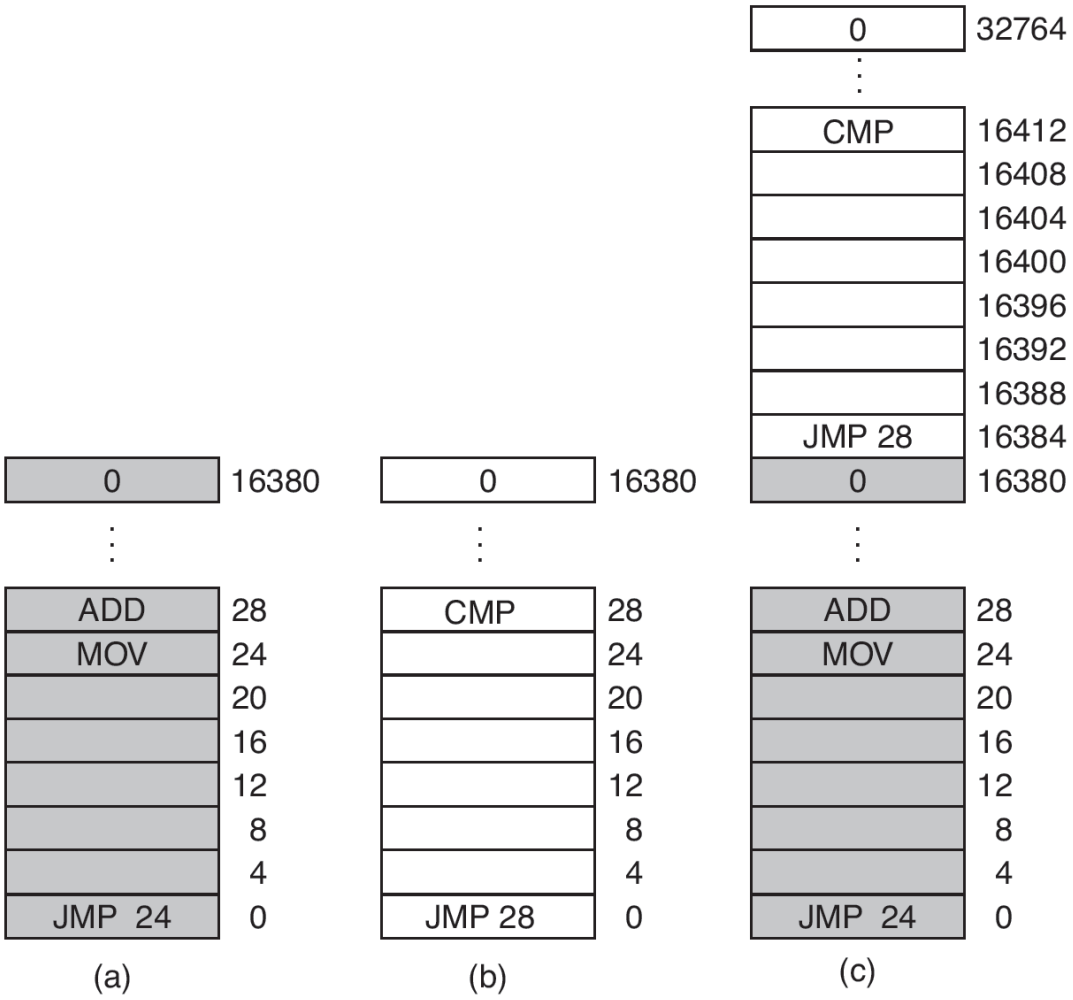
	# Processos			
	1	2	3	4
CPU ociosa	0,80	0,64	0,51	0,41
CPU ocupada	0,20	0,36	0,49	0,59
CPU/Processo	0,20	0,18	0,16	0,15

(b)

Utilização da CPU por até 4 Jobs com 80% d espera por E/S.

Gerenciamento de Memória

Multiprogramação sem abstração de memória



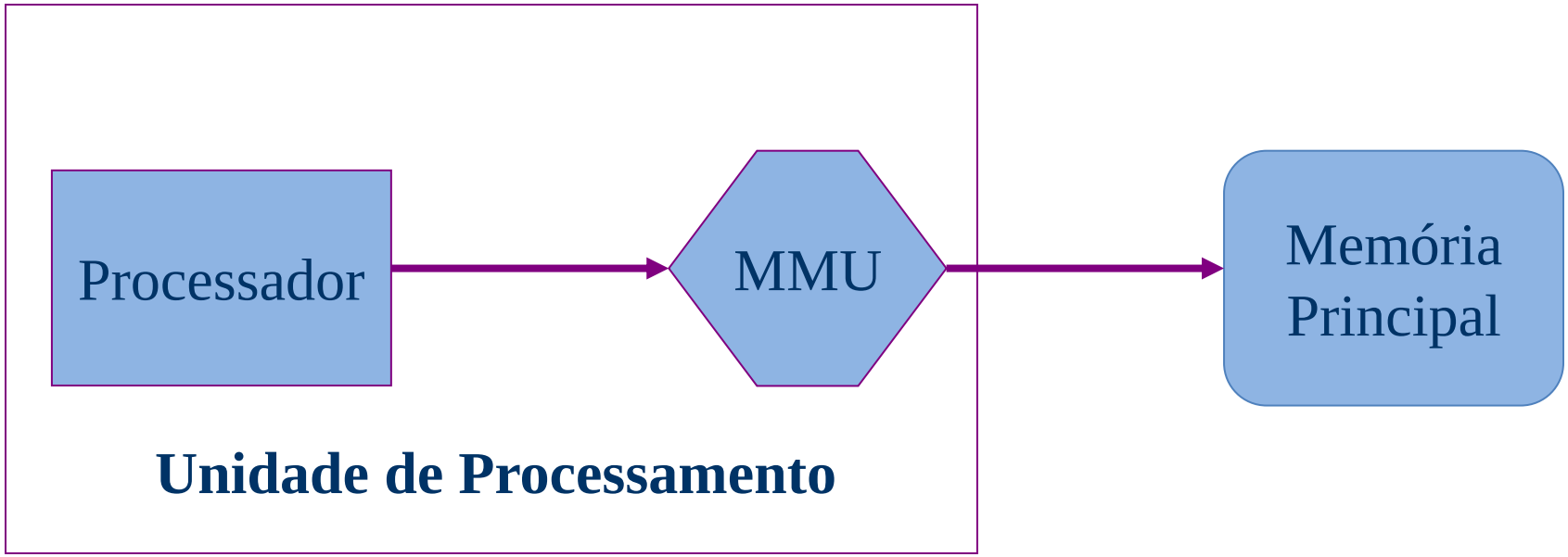
gura 3.2 Ilustração do problema de realocação. (a) Um programa de 16 KB. (b) Outro programa de 16 KB. (c) Os dois programas arregados consecutivamente na memória.

Gerenciamento de Memória

- Multiprogramação → Vários processos na memória:
  - Como proteger os processos uns dos outros e o *kernel* de todos os processos?
  - Como tratar a realocação?
- Todas as soluções envolvem equipar a CPU com um hardware especial → MMU (*memory management unit*);

Gerenciamento de Memória

- MMU – *Memory Management Unit*



Gerenciamento de Memória

- **MMU** (do inglês **Memory Management Unit**) é um dispositivo de hardware que transforma endereços virtuais em endereços físicos.
- Na MMU, o valor no registro de re-locação é adicionado a todo o endereço lógico gerado por um processo do utilizador na altura de ser enviado para a memória. O programa do utilizador manipula endereços lógicos; ele nunca vê endereços físicos reais.

Gerenciamento de Memória

- Normalmente o sistema atual de MMU divide o espaço de endereçamento virtual (endereços utilizados pelo processador) em páginas, cujo o tamanho é de 2^n, tipicamente poucos ***kilobytes***.

- A MMU normalmente traduz número de páginas virtuais para número de páginas físicas utilizando uma *cache* associada chamada *Translation Lookaside Buffer* (TLB). Quando o TLB falha uma tradução, um mecanismos mais lento envolvendo um *hardware* específico de dados estruturados ou um *software* auxiliar é usado.

Relocação e Proteção

- Pode não ter certeza de onde o programa será carregado na memória
  - As localizações de endereços de localização das variáveis e do código das rotinas não podem ser absolutos
  - Deve-se manter um programa fora das partições de outros processos

- Uso de valores de base e limite
  - Os endereços das localizações são somados a um valor de base para mapear um endereço físico
  - Valores de localizações maiores que um valor limite são considerados erro

Gerenciamento de Memória

- Realocação:
  - Quando um programa é *linkado* (programa principal + rotinas do usuário + rotinas da biblioteca → executável) o *linker* deve saber em que endereço o programa irá iniciar na memória;
  - Nesse caso, para que o *linker* não escreva em um local indevido, como por exemplo na área do SO (100 primeiros endereços), é preciso de realocação:
    - #100 + Δ → que depende da partição!!!

Gerenciamento de Memória

- Proteção:
  - Com várias partições e programas ocupando diferentes espaços da memória é possível acontecer um acesso indevido;
- Solução para ambos os problemas:
  - 2 registradores → base e limite
    - Quando um processo é escalonado o registrador-base é carregado com o endereço de início da partição e o registrador-limite com o tamanho da partição;
    - O registrador-base torna impossível a um processo uma remissão a qualquer parte de memória abaixo de si mesmo.

Gerenciamento de Memória

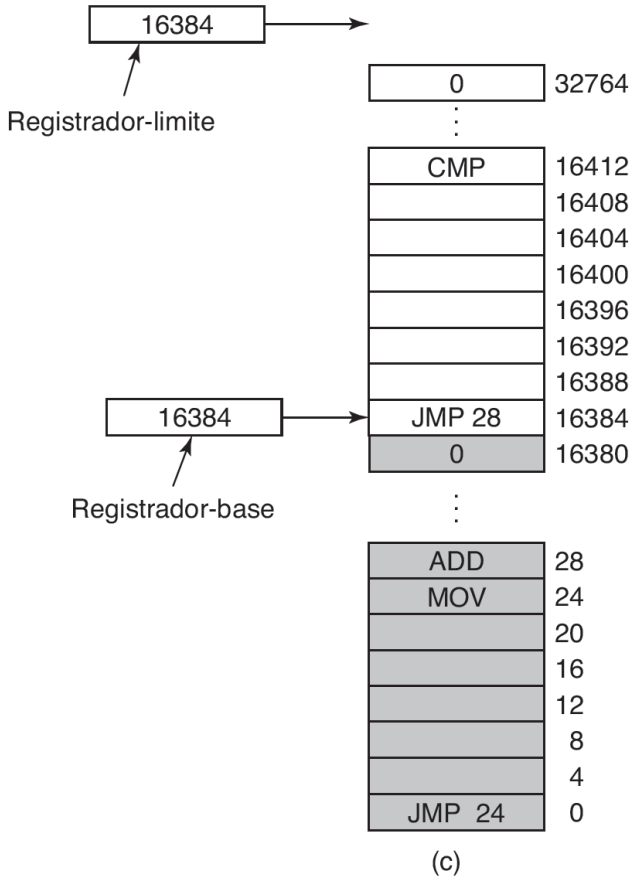


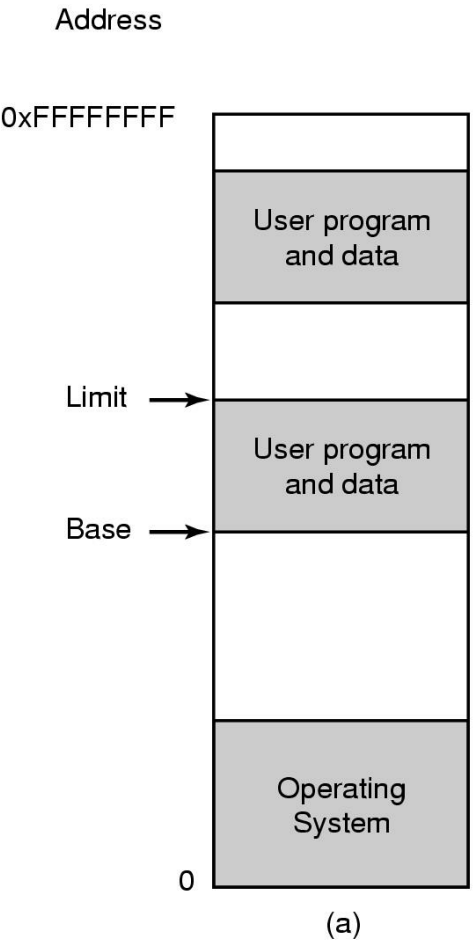
Figura 3.3 O registrador-limite e o registrador-base podem ser usados para dar a cada processo um espaço de endereçamento independente.

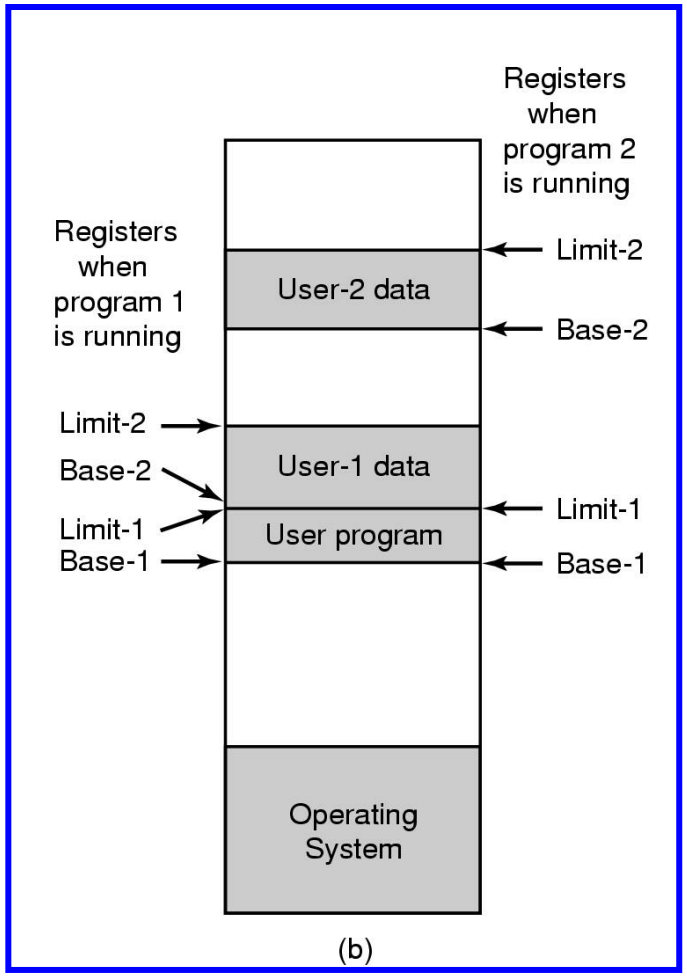
Gerenciamento de Memória

- 2 registradores → base e limite
  - Automaticamente, a MMU adiciona o conteúdo do registrador-base a cada endereço de memória gerado;
  - Endereços são comparados com o registrador-limite para prevenir acessos indevidos;

Gerenciamento de Memória

Registradores base e limite





b) MMU mais sofisticada → dois pares de registradores: segmento de dados usa um par separado;

- MMU modernas têm mais pares de registradores.

Gerenciamento de Memória

- Tipos básicos de gerenciamento:
  - Com paginação (chaveamento): Processos são movidos entre a memória principal e o disco; artifício usado para resolver o problema da falta de memória;
    - Se existe MP suficiente não há necessidade de se ter paginação;
  - Sem paginação: não há chaveamento;

Gerenciamento de Memória
Alocando memória

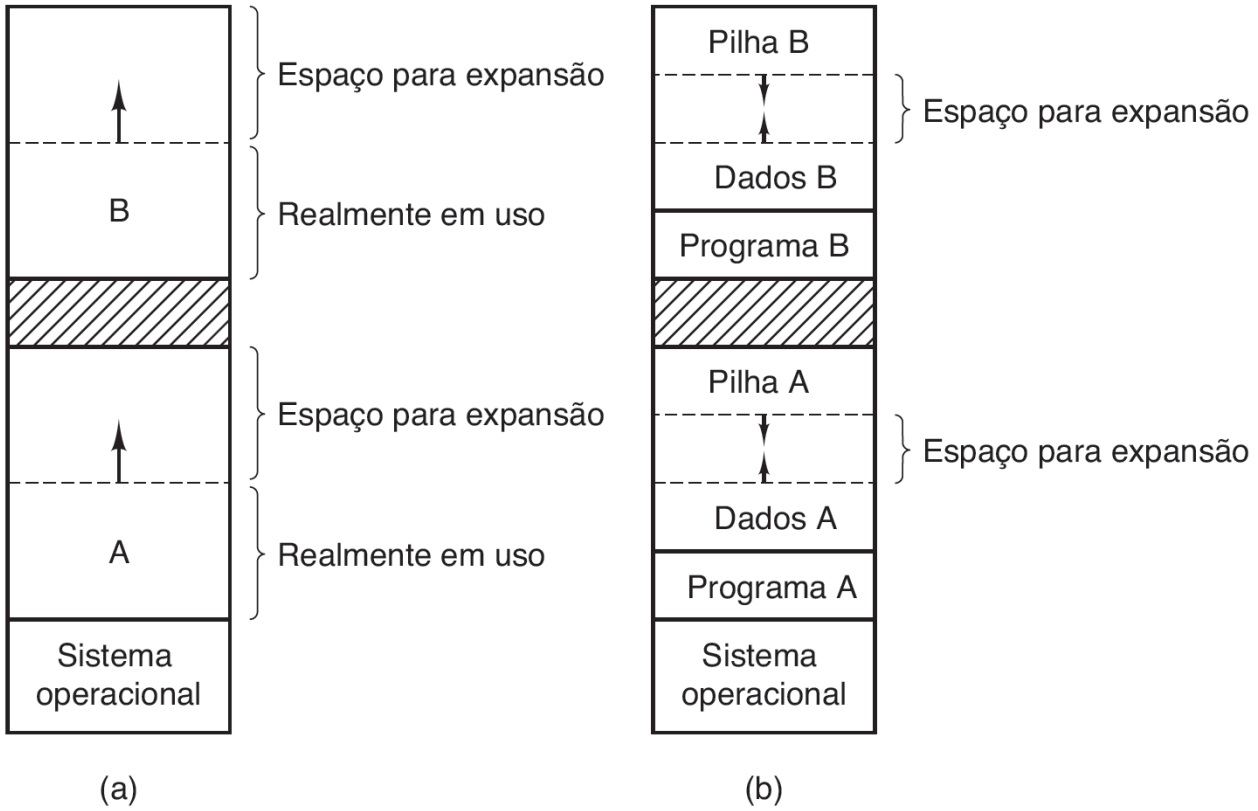


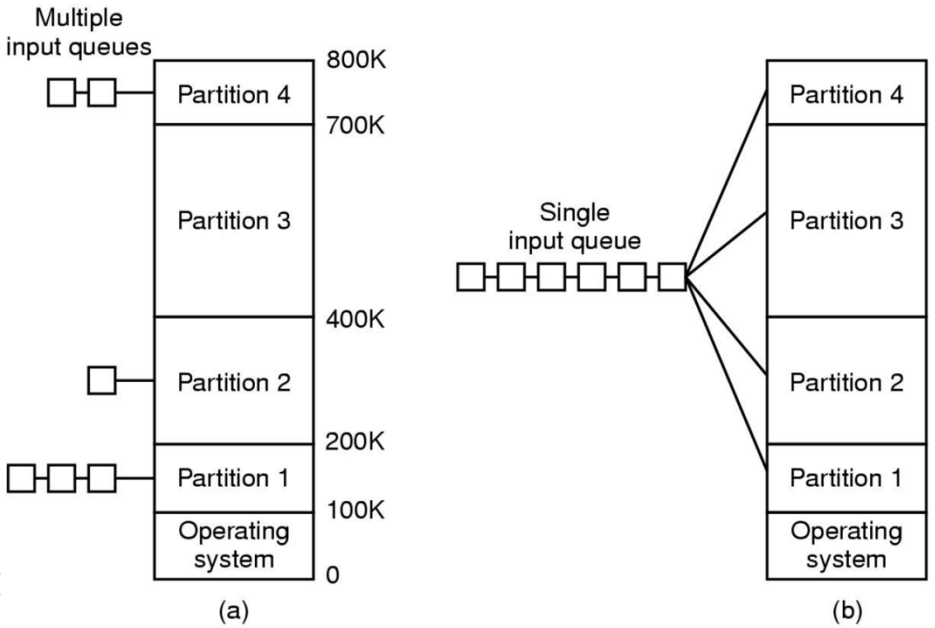
Figura 3.5 (a) Alocação de espaço para um segmento de dados em expansão. (b) Alocação de espaço para uma pilha e um segmento de dados em crescimento.

Gerenciamento de Memória
Partições/Alocação

- Particionamento da memória pode ser realizado de duas maneiras:
  - Partições fixas (ou alocação estática);
  - Partições variáveis (ou alocação dinâmica);
- **Partições Fixas**:
  - Tamanho e número de partições são fixos (estáticos);
  - Não é atrativo, porque partições fixas tendem a desperdiçar memória (Qualquer espaço não utilizado é literalmente perdido)
  - Mais simples;

Gerenciamento de Memória
Partições Fixas

- **Partições Fixas:**
  - Filas múltiplas:
    - Problema: filas não balanceadas;
  - Fila única:
    - Facilita gerenciamento;
    - Implementação com Lista:
      - Melhor utilização da memória, pois procura o melhor processo para a partição considerada;
      - Problema: processos menores são prejudicados;



Gerenciamento de Memória
Partições Fixas

- Problemas com fragmentação:
  - **Interna**: desperdício dentro da área alocada para um processo;
    - Ex.: processo de tamanho 40K ocupando uma partição de 50k;
  - **Externa**: desperdício fora da área alocada para um processo;
    - Duas partições livres: PL1 com 25k e PL2 com 100k, e um processo de tamanho 110K para ser executado;
    - Livre: 125K, mas o processo não pode ser executado;

Gerenciamento de Memória
Partições Variáveis

- **Partições Variáveis**:
  - Tamanho e número de partições variam;
  - Otimiza a utilização da memória, mas complica a alocação e liberação da memória;
  - Partições são alocadas dinamicamente;
  - SO mantém na memória uma lista com os espaços livres;
  - Menor fragmentação interna e grande fragmentação externa;
    - Solução: Compactação;

Gerenciamento de Memória
Partições Variáveis

- **Partições Variáveis:**

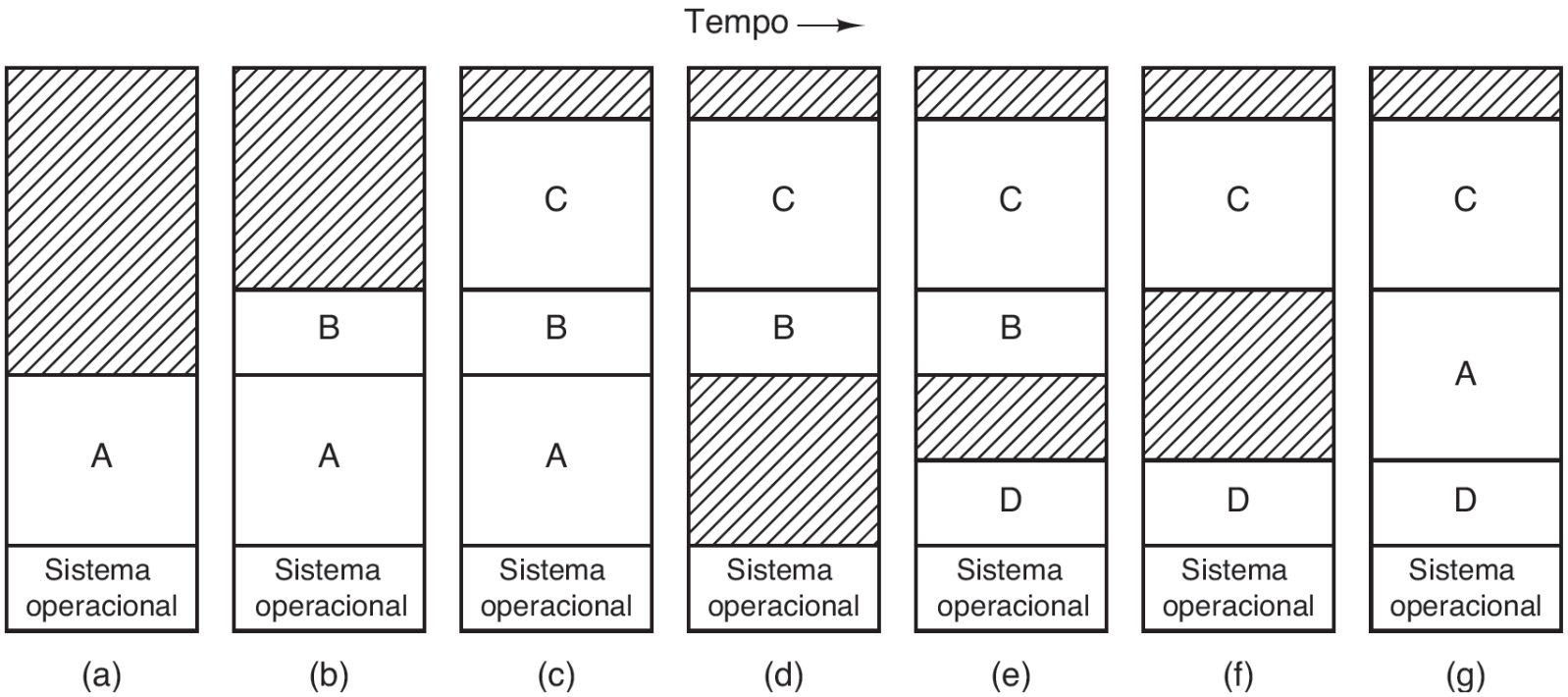


Figura 3.4 Alterações na alocação de memória à medida que processos entram e saem dela. As regiões sombreadas correspondem a regiões da memória não utilizadas naquele instante.

Swapping

Com um sistema em lote, é simples e eficiente organizar a memória em partições fixas.

Em sistemas de tempo compartilhado:

- Pode não existir memória suficiente para conter todos os processos ativos
- Os processos excedentes são mantidos em disco e trazidos dinamicamente para a memória a fim se serem executados

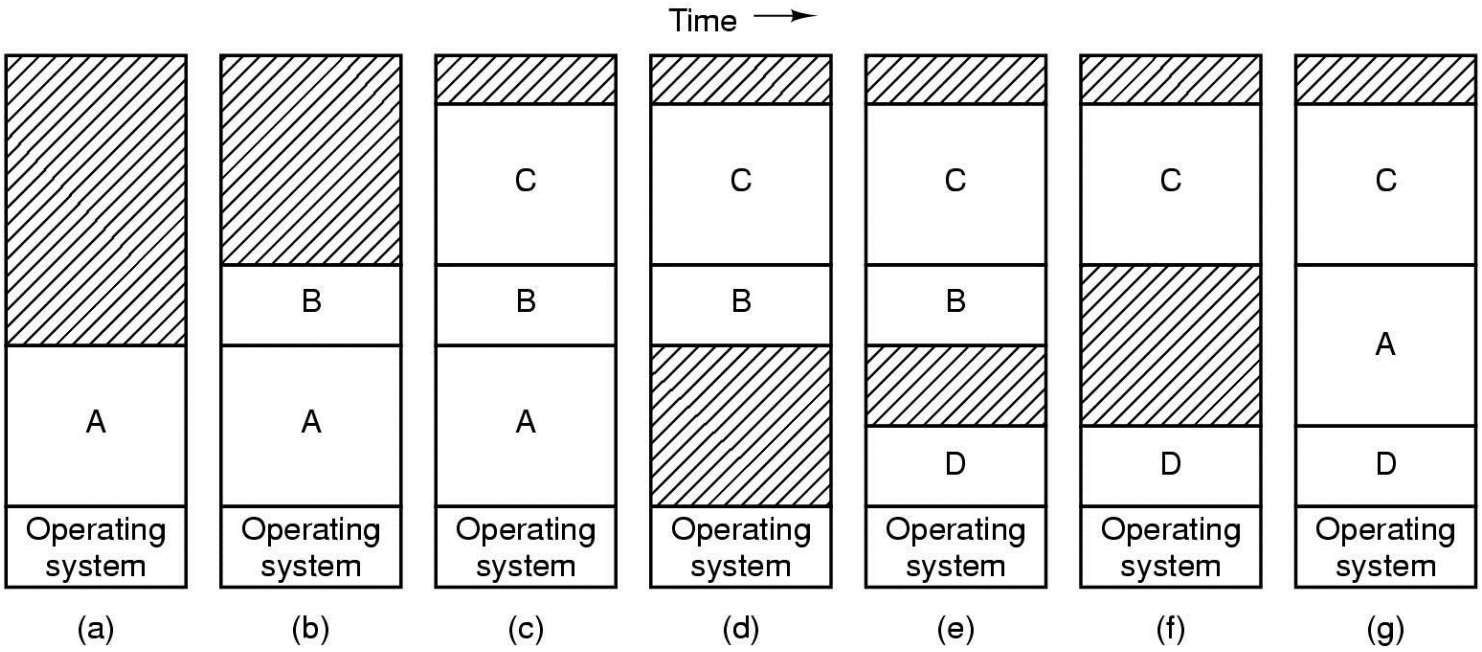
Swapping

Existem 2 processos gerais que podem ser usados:

- *Swapping(troca de Processos*): forma mais simples, consiste em trazer totalmente cada processo para a memória, executá-lo durante um tempo e, então, devolvê-lo ao disco

- Memória Virtual: permite que programas possam ser executados mesmo que estejam parcialmente carregados na memória principal.

Swapping



A Alocação de memória muda a medida que

- Os processos chegam à memória
- Os processos deixam a memória

As regiões sombreadas (na Figura) representam a memória não usada

Gerenciamento de Memória

- Minimizar espaço de memória inutilizados:
  - Compactação: necessária para recuperar os espaços perdidos por fragmentação; no entanto, muito custosa para a CPU;
- Técnicas para alocação dinâmica de memória:
  - *Bitmaps*;
  - Listas Encadeadas;

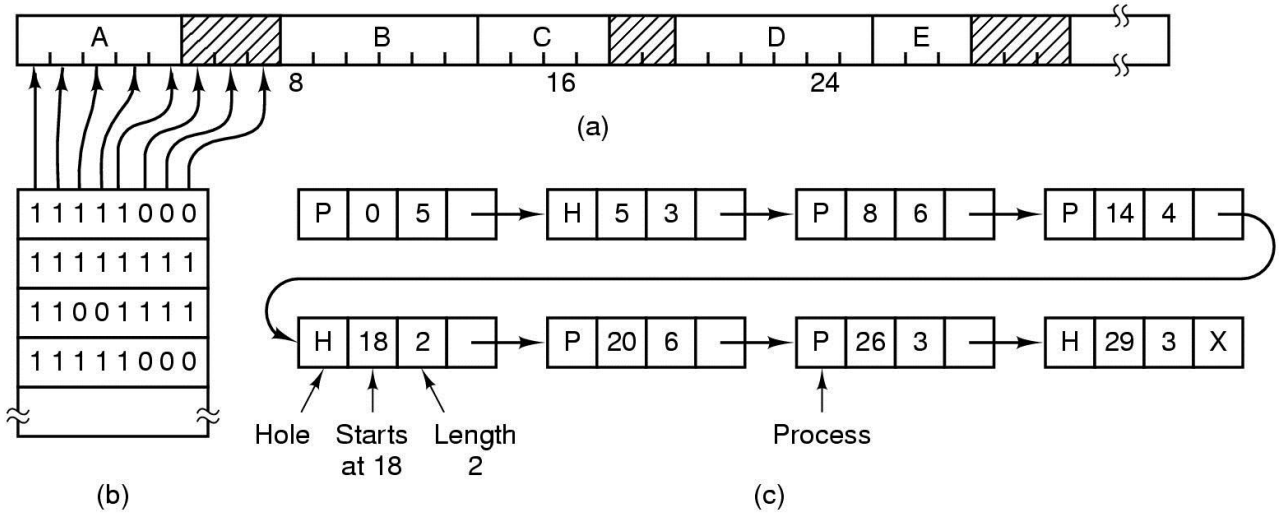
Gerenciamento de Memória

- Técnica com *Bitmaps*:
  - Memória é dividida em unidades de alocação em kbytes;
  - Cada unidade corresponde a um *bit* no *bitmap*:

    0 → livre

    1 → ocupado
  - Tamanho do *bitmap* depende do tamanho da unidade e do tamanho da memória;
  - Ex.:
    - unidades de alocação pequenas → *bitmap* grande;
    - unidades de alocação grandes → perda de espaço;

Gerenciamento de memória com Mapas de Bits



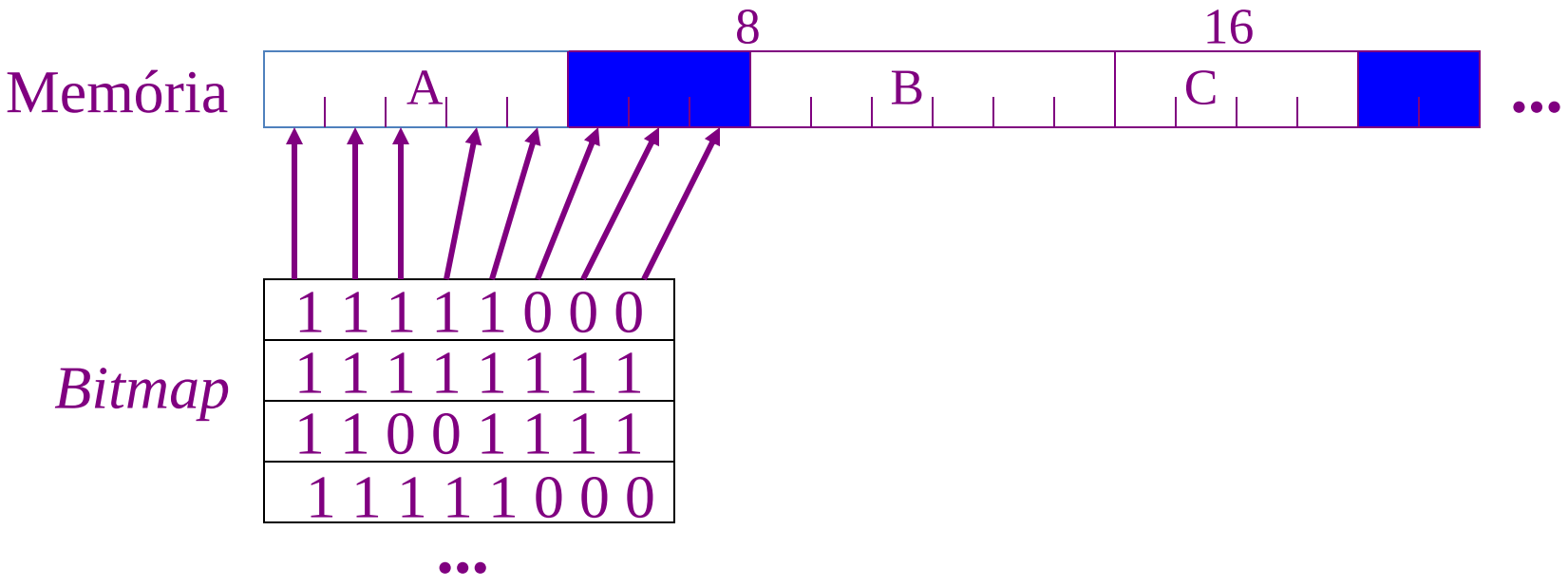
(a) Uma parte da memória com 5 processos e 3 buracos
- As regiões em branco (1 no bitmap) marcam as unidades já alocadas
- As regiões sombreadas (0 no bitmap) marcam unidades desocupadas

(b) O Bitmap correspondente

(c) A mesma informação como uma lista encadeada

Gerenciamento de Memória

- Técnica com *Bitmaps*:



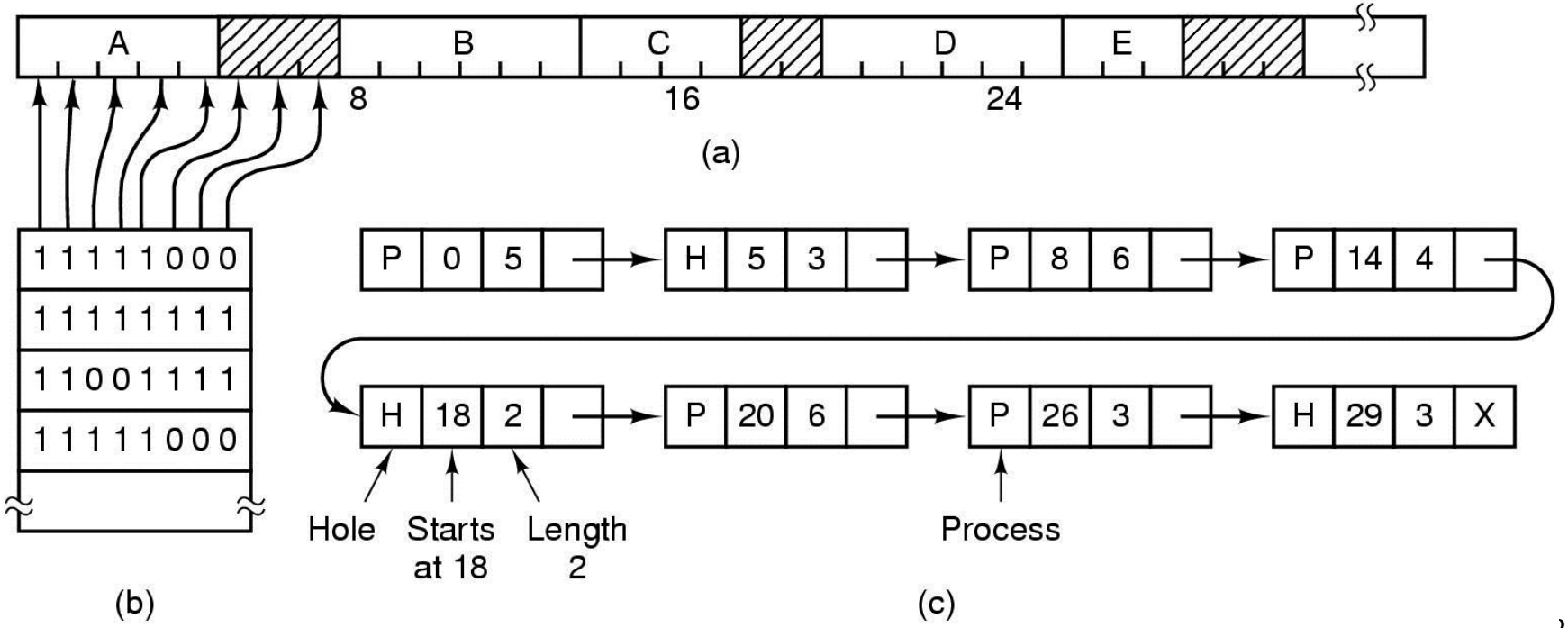
Memória ocupada
Memória livre

Gerenciamento de Memória com Listas encadeadas

Outra maneira de gerenciar memória é manter uma lista encadeada de segmentos de memória alocados e segmentos disponíveis

Cada elemento desta lista especifica:

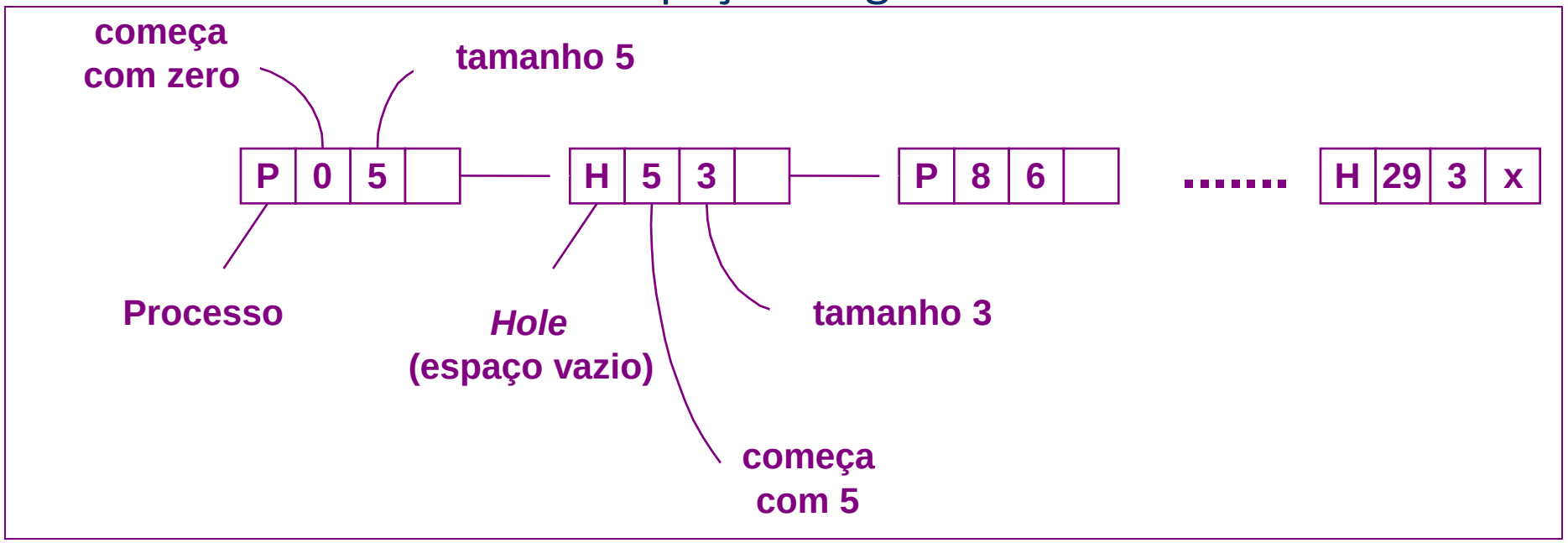
- um segmento disponível (H), ou alocado a um processo (P),
- o endereço onde se inicia este segmento
- e um ponteiro para o próximo elemento



Gerenciamento de Memória

- Técnica com Listas Encadeadas:
  - Uma lista para os espaços vazios e outra para os espaços cheios, ou uma lista para ambos!

"espaço ≡ segmento"



Gerenciamento de Memória

- Técnica com Listas Encadeadas

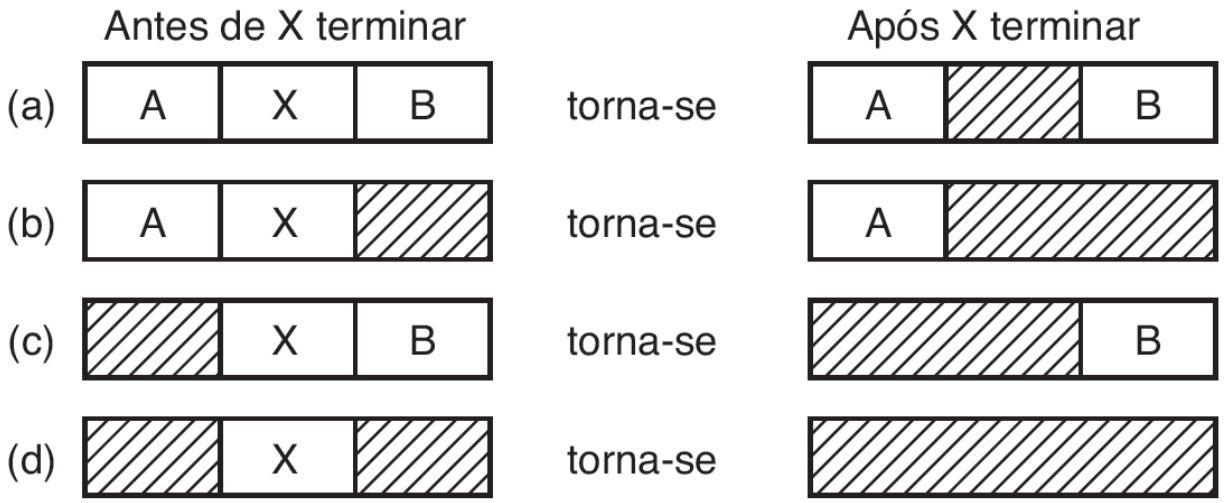


Figura 3.7 Quatro combinações de vizinhos para o processo que termina, *X*.

Algoritmos para alocação de memória

- Existem três métodos que podem ser usados para selecionar uma região para um processo. Os algoritmos de alocação são:
  - *First fit*: ir sequencialmente através dos blocos, e tomar o primeiro grande o suficiente.
  - *Next fit*: proximo bloco grande o suficiente ´
  - *Best fit*: colocar o processo no bloco com o mínimo resto de memória;
  - *Worst fit:* usar o bloco com o maior resto de memória;

Gerenciamento de Memória

- Algoritmos de Alocação → quando um novo processo é criado:
  - *FIRST FIT*
    - 1º segmento é usado;
    - Rápido, mas pode desperdiçar memória por fragmentação;

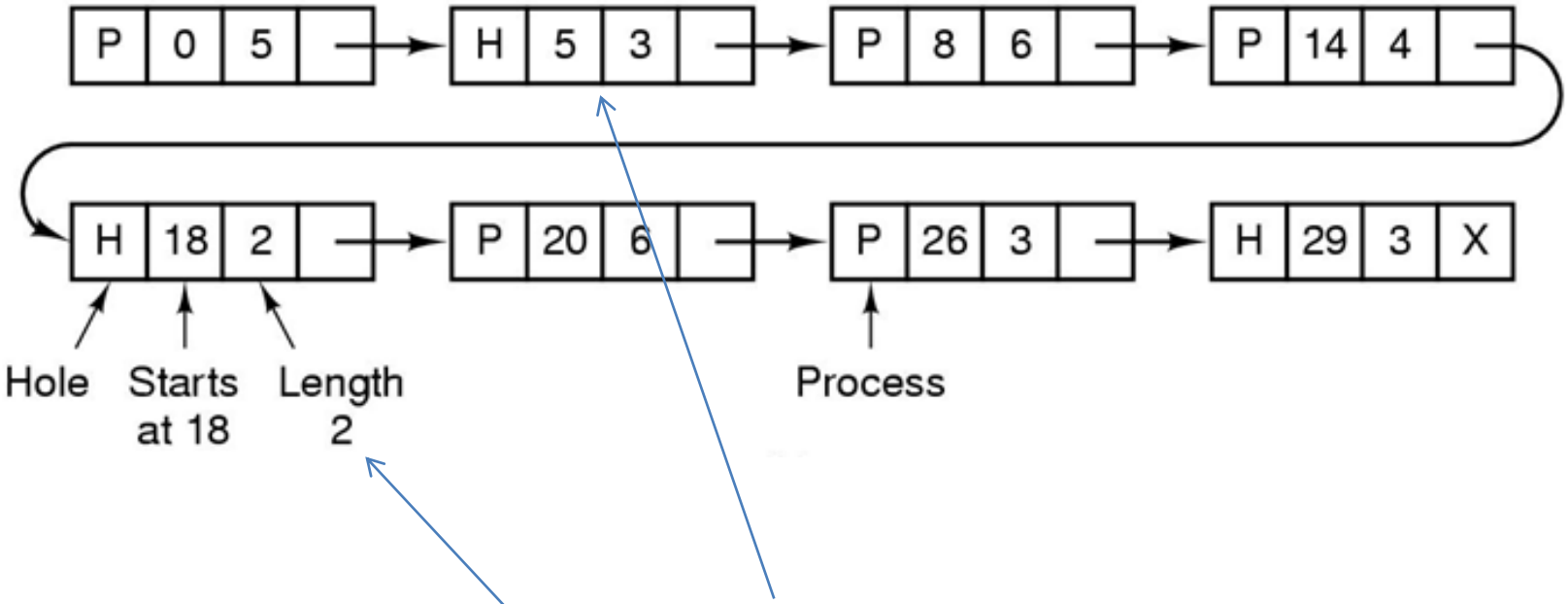
Gerenciamento de Memória

- Algoritmos de Alocação → quando um novo processo é criado:
  - *FIRST FIT*
    - 1º segmento é usado;
    - Rápido, mas pode desperdiçar memória por fragmentação;
  - *NEXT FIT*
    - 1º segmento é usado;
    - Mas na próxima alocação inicia busca do ponto que parou anteriormente;
    - SIMULAÇÕES (Bays, 1977): desempenho ligeiramente inferior;

Gerenciamento de Memória

- *BEST FIT*
  - Procura na lista toda e aloca o espaço que mais convém;
  - Menor fragmentação;
  - Mais lento;

Gerenciamento de Memória



Exemplo de *Best fit e First fit*

Gerenciamento de Memória

- *BEST FIT*
  - Procura na lista toda e aloca o espaço que mais convém;
  - Menor fragmentação;
  - Mais lento;
- *WORST FIT*
  - Aloca o maior espaço disponível;

Gerenciamento de Memória

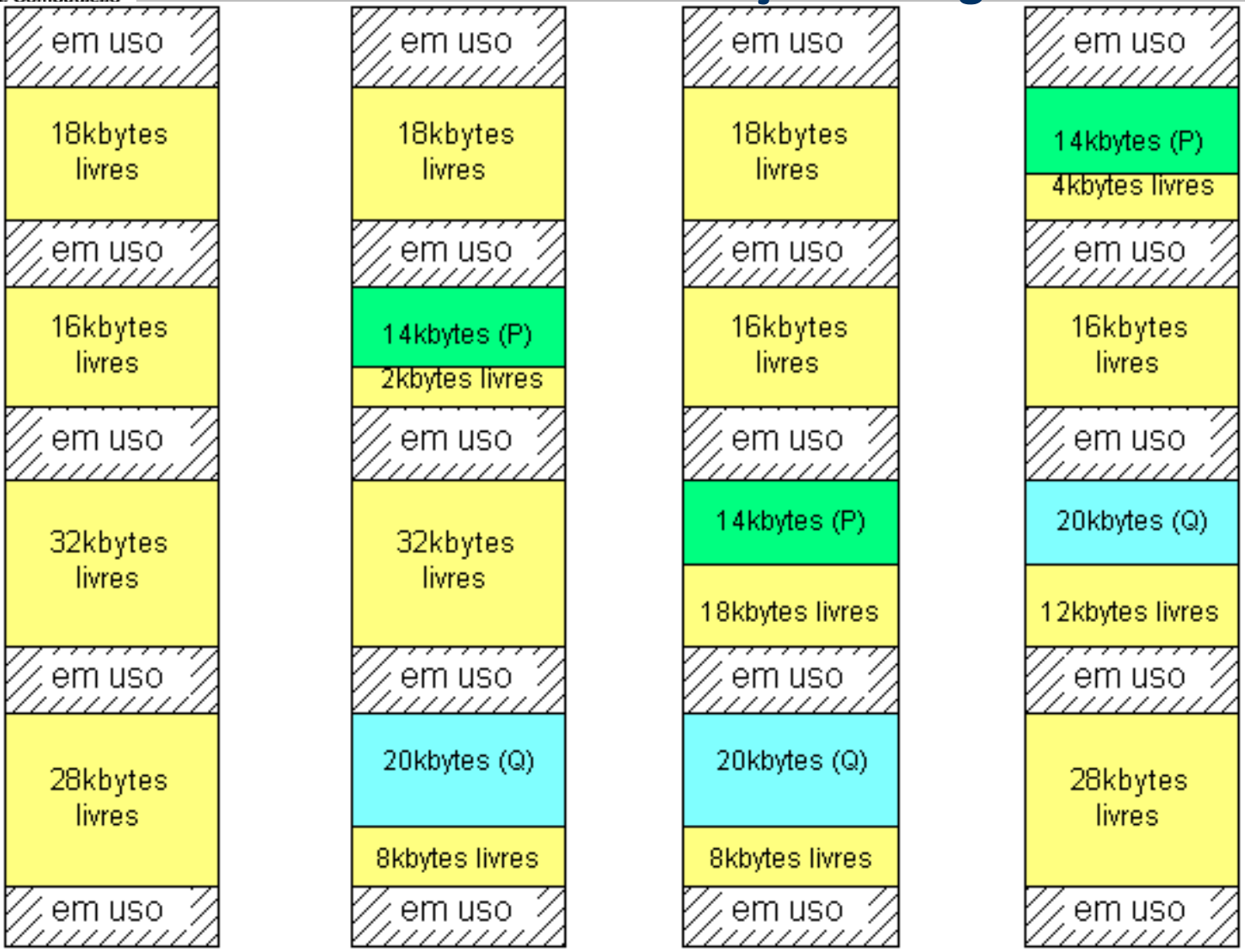
- *BEST FIT*
  - Procura na lista toda e aloca o espaço que mais convém;
  - Menor fragmentação;
  - Mais lento;
- *WORST FIT*
  - Aloca o maior espaço disponível;
- *QUICK FIT*
  - Mantém listas separadas para os espaços mais requisitados;

Gerenciamento de Memória

- Cada algoritmo pode manter listas separadas para processos e para espaços livres:
  - Vantagem:
    - Aumenta desempenho;
  - Desvantagens:
    - Aumenta complexidade quando espaço de memória é liberado – gerenciamento das listas;
    - Fragmentação;

Gerenciamento de Memória

Alocação de segmentos livres



Áreas livres iniciais | Best Fit | Worst Fit | First Fit

Gerenciamento de Memória

Alocação de segmentos livres

- **Principais Conseqüências**

- **Best fit:** deixa o menor resto, porém após um longo processamento poderá deixar "buracos" muito pequenos para serem úteis.

- **Worst Fit:** deixa o maior espaço após cada alocação, mas tende a espalhar as porções não utilizadas sobre áreas não contínuas de memória e, portanto, pode tornar difícil alocar grandes jobs.

- **First Fit:** tende a ser um meio termo entre a melhor e a pior escolha, com a característica adicional de fazer com que os espaços vazios migrem para o final da memória.

Departamento de Computação

Gerenciamento de Memória

Alocação de segmentos livres

List of Jobs	Size	Turnaround
Job 1	100k	3
Job 2	10k	1
Job 3	35k	2
Job 4	15k	1
Job 5	23k	2
Job 6	6k	1
Job 7	25k	1
Job 8	55k	2
Job 9	88k	3
Job 10	100k	3

Memory Block	Size
Block 1	50k
Block 2	200k
Block 3	70k
Block 4	115k
Block 5	15k

Gerenciamento de Memória

Alocação de segmentos livres

- **First fit**

Memory Block	Size	List of Jobs
Block 1	50k	**Job 2**
Block 2	200k	**Job 1**
Block 3	70k	**Job 3**
Block 4	115k	**Job 4**
Block 5	15k	**Job 6**

Memory Block	Size	List of Jobs
Block 1	50k	**Job 5**
Block 2	200k	**Job 1**
Block 3	70k	**Job 3**
Block 4	115k	**Job 7**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	**Job 5**
Block 2	200k	**Job 1**
Block 3	70k	**Job 8**
Block 4	115k	**Job 9**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	**Job 10**
Block 3	70k	**Job 8**
Block 4	115k	**Job 9**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	**Job 10**
Block 3	70k	
Block 4	115k	**Job 9**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	**Job 10**
Block 3	70k	
Block 4	115k	
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	
Block 3	70k	
Block 4	115k	
Block 5	15k	

Gerenciamento de Memória

Alocação de segmentos livres

- **Best fit**

Memory Block	Size	List of Jobs
Block 1	50k	**Job 3**
Block 2	200k	**Job 5**
Block 3	70k	**Job 4**
Block 4	115k	**Job 1**
Block 5	15k	**Job 2**

Memory Block	Size	List of Jobs
Block 1	50k	**Job 3**
Block 2	200k	**Job 5**
Block 3	70k	**Job 7**
Block 4	115k	**Job 1**
Block 5	15k	**Job 6**

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	**Job 9**
Block 3	70k	**Job 8**
Block 4	115k	**Job 1**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	**Job 9**
Block 3	70k	**Job 8**
Block 4	115k	**Job 10**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	**Job 9**
Block 3	70k	
Block 4	115k	**Job 10**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	
Block 3	70k	
Block 4	115k	**Job 10**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	
Block 3	70k	
Block 4	115k	
Block 5	15k	

Gerenciamento de Memória

Alocação de segmentos livres

- **Worst fit**

Memory Block	Size	List of Jobs
Block 1	50k	**Job 4**
Block 2	200k	**Job 1**
Block 3	70k	**Job 3**
Block 4	115k	**Job 2**
Block 5	15k	**Job 6**

Memory Block	Size	List of Jobs
Block 1	50k	**Job 7**
Block 2	200k	**Job 1**
Block 3	70k	**Job 3**
Block 4	115k	**Job 5**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	**Job 1**
Block 3	70k	**Job 8**
Block 4	115k	**Job 5**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	**Job 9**
Block 3	70k	**Job 8**
Block 4	115k	**Job 10**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	**Job 9**
Block 3	70k	
Block 4	115k	**Job 10**
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	
Block 3	70k	
Block 4	115k	
Block 5	15k	

Memory Block	Size	List of Jobs
Block 1	50k	
Block 2	200k	**Job 9**
Block 3	70k	
Block 4	115k	**Job 10**
Block 5	15k	