Organização e Arquitetura de Computadores I

Memória Virtual

- os primeiros computadores (início dos anos 60) tinham memória principal muito reduzida
  - O PDP-1 funcionava com uma memória de 4096 palavras de 18 bits cada para rodar o sistema operacional e também os programas dos usuários.
- Por falta de memória, as vezes se implementavam algoritmos mais lentos.
- A solução tradicional para a falta de memória era o uso de memória secundária.

Overlays

- O programa era dividido em partes, chamadas ***overlays***, que cabiam na memória disponível.
- Cada *overlay* era carregado do disco para a memória, segundo a sequência do programa, e executado.
- O programador era responsável por gerenciar todo o processo de *overlays* sem qualquer ajuda do computador.

Overlays

- A definição de *overlays* evoluiu para o conceito de memória virtual.
  - 1961 - um grupo de pesquisadores ingleses apresentou um S.O. capaz de processar *overlays* automaticamente, implementando assim o conceito de memória virtual.
  - 1970 - essa ferramenta estava implementada em, praticamente, todas as arquiteturas computacionais.
  - Atualmente - existem sistemas sofisticados de memória virtual.

Paginação

- A paginação é uma implementação de memória virtual que usa os conceitos de:
  - **Espaço de endereços físicos**: espaço real disponível em memória.
  - **Espaço de endereços virtuais**: endereços virtuais do programa, que podem ser maior do que o espaço de memória física.

Paginação - Características

- Os programas são escritos com base no pressuposto de que o tamanho de memória principal é suficiente para todo o espaço de endereços virtuais.
- Os programas podem trazer, ou armazenar, qualquer palavra do, ou no, espaço virtual ou desviar para qualquer instrução situada dentro do espaço virtual, sem se preocupar com o tamanho da memória física.

Paginação - Características

- **A paginação dá ao programador a ilusão de uma memória principal grande**, com endereços contíguos e lineares, do mesmo tamanho da memória virtual.
- Como o programador pode escrever seu programa como se não existisse a paginação, esse mecanismo é chamado de **transparente**.
- Somente os programadores de sistemas operacionais (e os estudantes de OAC e de S.O.) precisam saber como essa ilusão se sustenta.

Paginação - Características

- **Um mapa de memória, ou tabela de páginas, relaciona os endereços virtuais com os endereços físicos.**
- Para permitir o mapeamento de endereços virtuais em endereços físicos e facilitar a transferência de informação entre memória principal e HD, o espaço de endereçamento virtual é dividido em blocos de endereços, tipicamente, de tamanho físico.
- Denominação dos blocos: páginas (tamanho típico 4K, 1K palavras de 32 bits).

Paginação - Características

- O espaço de endereçamento físico também é dividido em pedaços do mesmo tamanho do virtual. Essas partes são conhecidas como **molduras de página**.
- O mapeamento virtual-físico é realizado por um dispositivo conhecido com MMU (*Memory Management Unit* - **Unidade de Gerenciamento de Memória**).

a) Os primeiros 64K do espaço de endereços virtuais divididos em 16 páginas, de 4K cada uma.

b) Memória principal de 32 K dividida em 8 molduras de página, de 4K cada.

Página	Endereços virtuais
15	61440 – 65535
14	57344 – 61439
13	53248 – 57343
12	49152 – 53247
11	45056 – 49151
10	40960 – 45055
9	36864 – 40959
8	32768 – 36863
7	28672 – 32767
6	24576 – 28671
5	20480 – 24575
4	16384 – 20479
3	12288 – 16383
2	8192 – 12287
1	4096 – 8191
0	0 – 4095

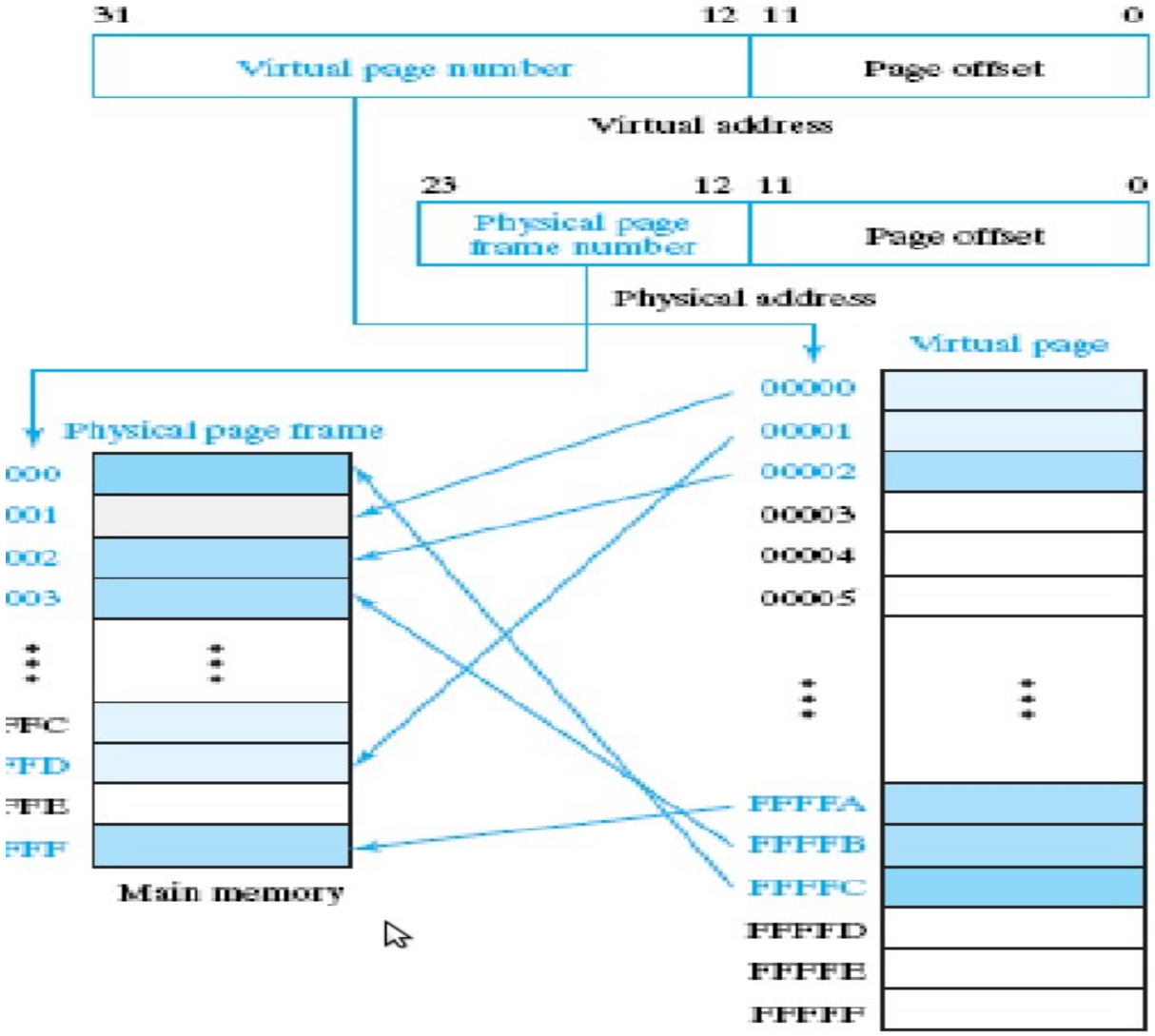
(a)

Moldura de página	32 K de memória principal Endereços físicos
7	28672 – 32767
6	24576 – 28671
5	20480 – 24575
4	16384 – 20479
3	12288 – 16383
2	8192 – 12287
1	4096 – 8191
0	0 – 4095

(b)

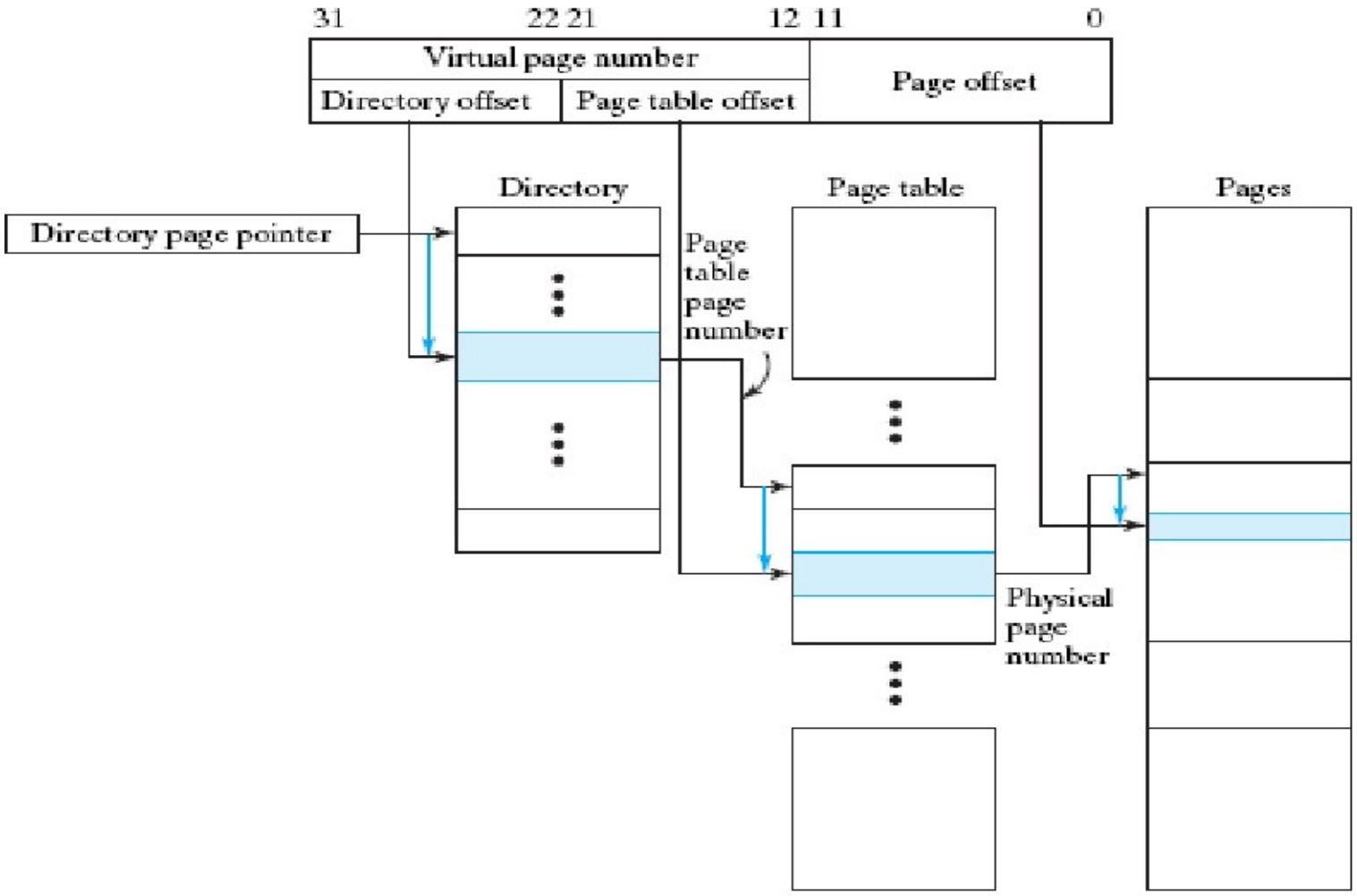
Paginação - Exemplo

- Funcionamento do dispositivo - Os 32 bits de um endereço virtual são divididos em 2 partes:
  - A primeira, com 20 bits, representando o número da página virtual
  - A segunda, com 12 bits, representando o deslocamento dentro da página
  - O número da página virtual é usado para indexar a tabela de páginas.


31
12
11
0
Virtual page number
Page offset
Virtual address
23
12
11
0
Physical page frame number
Page offset
Physical address
Virtual page
Physical page frame
00000
00001
00002
00003
00004
00005
FFFFA
FFFFB
FFFFC
FFFFD
FFFFE
FFFFF
000
001
002
003
FFC
FFD
FFE
FFF
Main memory

Paginação - Implementação

- Geralmente, existe um grande número de pagínas virtuais, cada uma das quais precisa ser mapeada (MP <-> HD).
- Elementos importantes:
  - ***Page table***: estrutura de dados que permite o armazenamento dos mapeamentos .
  - ***Page directory***: tabela especial fornece o mapeamento usado para localizar os programas nas *Page tables*.


31
22 21
12 11
0
Virtual page number
Directory offset
Page table offset
Page offset
Directory
Page table
Pages
Directory page pointer
Page table page number
Physical page number

Paginação por Demanda

- Quando é feita uma referência a um endereço situado em uma página que não está na memória principal diz-se que ocorreu uma **falta de página** (*page fault*).
- Neste caso é necessário que:
  - se leia, do disco, a página requisitada;
  - se coloque essa página na memória principal e;
  - se repita a referência ao endereço de interesse.
- Esse método de operação é chamado de paginação por demanda.

Paginação por Demanda

- Na paginação por demanda, as páginas são trazidas para a memória principal em função das requisições explícitas para cada uma delas e não antecipadamente.
- Quando existem vários processos sendo executados, num regime de compartilhamento de tempo, o mapeamento de páginas muda cada vez que há uma troca de contexto.
- Neste caso, a paginação por demanda pode ter um impacto negativo. Uma solução é usar um modelo, chamado **conjunto de trabalho**, que carrega, antecipadamente, as páginas necessárias à execução ou continuação de um processo.

Política de Substituição de Páginas

- Quando ocorre um problema de falha de página, o Sistema Operacional deve substituir uma moldura, página da memória física, por uma página do disco.
- O S. O. deve escolher, automaticamente, a página de mais baixa probabilidade de vir a pertencer ao conjunto de trabalho.
- Os algoritmos de substituição LRU e FIFO podem ser usados nesses casos.

Tamanho da Página e Fragmentação

- Se o programa, com seus dados, não couberem exatamente em um número inteiro de páginas haverá desperdícios. Alguns bytes da última página alocada não serão usados.
- O problema de desperdício de bytes é conhecido como **fragmentação interna**.
- Para páginas de n bytes se perde, em média, n/2 bytes. Isto sugere que páginas pequenas perdem menos espaço.

Tamanho da Página e Fragmentação

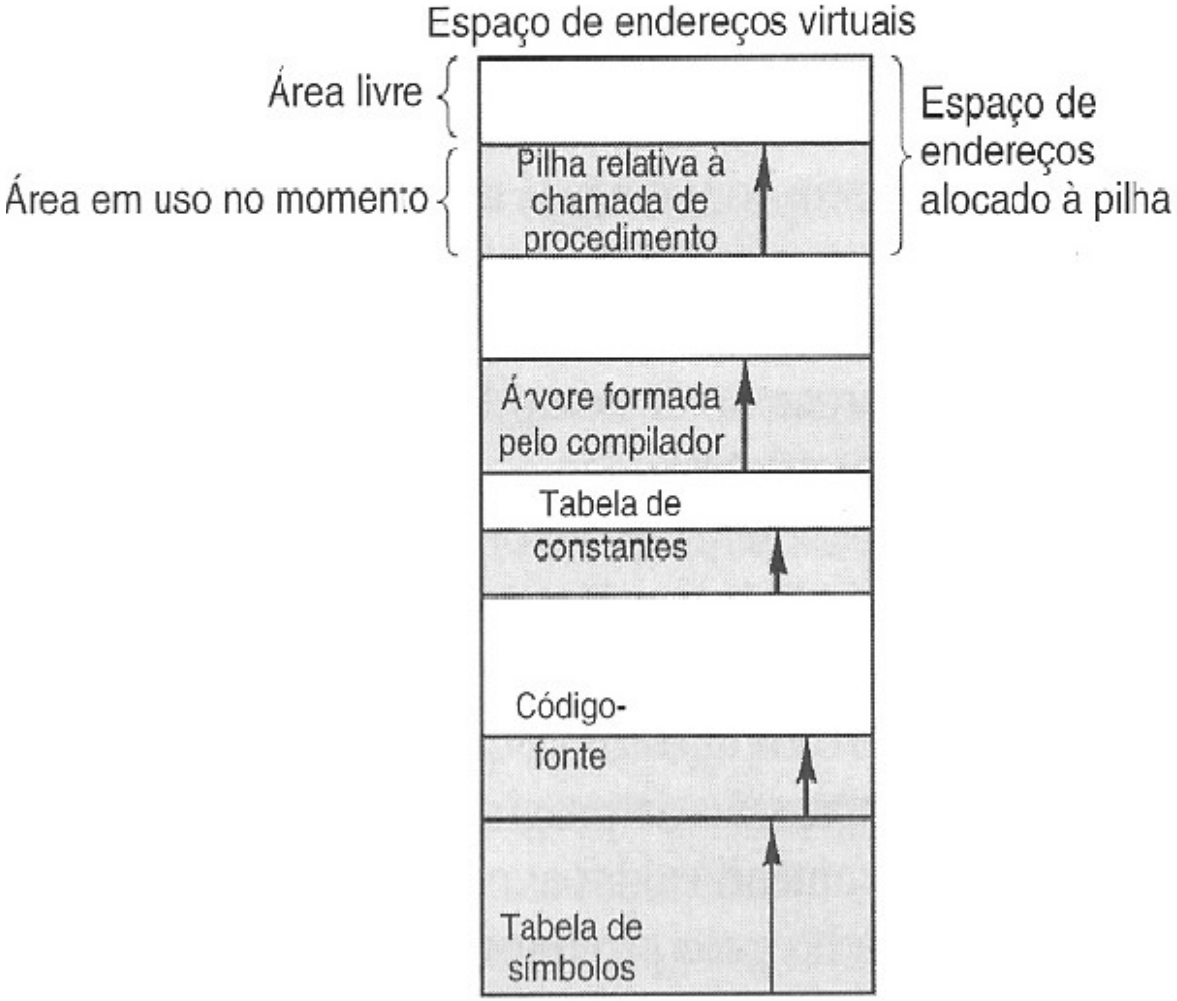
- Porém, quando as páginas são muito pequenas:
  - tem-se um número maior de páginas e, em consequência, uma tabela de páginas maior;
  - uma tabela de páginas maior requer mais registradores tornando o processador mais caro;
  - é maior o tempo gasto para mover páginas da memória virtual para a física.
- Portanto, tem que haver uma solução de compromisso para definição do tamanho das páginas.

Segmentação

- O sistema de memória virtual, discutido até o momento é unidimensional. Os endereços variam de 0 até um endereço máximo, um endereço após o outro.
- As vezes são necessários dois ou mais espaços de endereçamento virtual separados.
  - Um compilador gera muitas tabelas que precisam ser construídas na medida que o processo de compilação evolui.

Segmentação

- Entre as tabelas geradas pelo compilador, tem-se:
  - Tabela de símbolos, com os nomes e os atributos das variáveis.
  - Tabela com o código fonte.
  - Tabela de constantes inteiras e de ponto flutuante usadas pelo programa.
  - Tabela analítica, com a análise sintática do programa.
  - A pilha usada para chamadas a procedimentos dentro do compilador.


Espaço de endereços virtuais
Área livre
Área em uso no momento
Pilha relativa à chamada de procedimento
Espaço de endereços alocado à pilha
Árvore formada pelo compilador
Tabela de constantes
Código-fonte
Tabela de símbolos

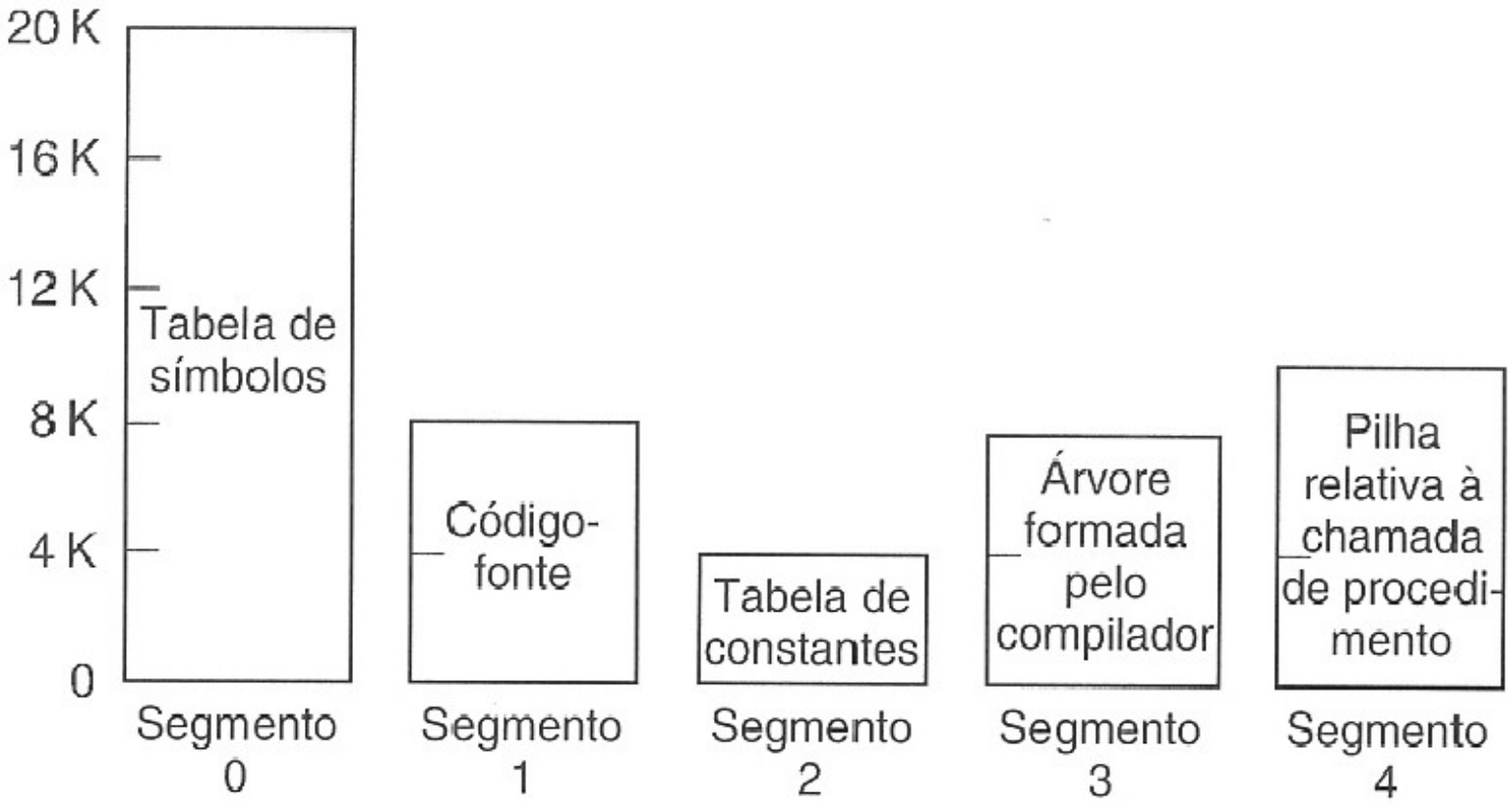
Segmentação

- Em um espaço de endereçamento unidimensional que suporte tabelas cujos tamanhos podem aumentar ou diminuir dinamicamente, existe o perigo de uma tabela invadir a área de outra. **As tabelas podem se expandir ou contrair durante a execução do programa**.
- A solução é fazer com que o sistema de memória suporte vários espaços de endereçamento **completamente independentes**. Esses espaços são denominados **segmentos**.

Segmentação

- Cada segmento é composto por uma sequência linear de endereços, de 0 até um valor máximo.
- Os tamanhos dos segmentos podem ser diferentes e variam durante a execução do programa.

Segmentação



Segmentação

- Alguns segmentos podem ser protegidos contra leitura ou escrita.
  - Exemplo: um segmento que contenha um procedimento pode ser especificado como segmento de código e, portanto, só pode ser lido.
- Como o usuário de uma memória segmentada tem a ilusão de que todos os segmentos estão em memória principal ao mesmo tempo, eles podem ser endereçados sem que tenhamos que nos preocupar com a administração do *overlay* da memória.

Segmentação - Implementação

- A segmentação pode ser implementada de duas maneiras: por ***swapping*** ou por **paginação**.
- Características da implementação por *swapping*:
  - é muito parecida com a paginação por demanda, os segmentos inteiros vão e vem do disco para a memória principal e vice-versa, na medida que são necessários;
  - a principal diferença é que as páginas têm tamanho fixo e os segmentos não.

Segmentação - Implementação

- Características da implementação por paginação :
  - é uma mistura de segmentação com paginação
  - há uma divisão de cada segmento em um conjunto de páginas e passa a trabalhar como no esquema de paginação por demanda.
  - exige-se que seja criada uma tabela de páginas, individual, para cada segmento.

Memória Virtual e Memória Cache

- Embora pareça que os assuntos memória virtual (paginação por demanda) e memória cache não têm qualquer relação entre si, conceitualmente eles são muito semelhantes.
- No caso de emprego da memória virtual:
  - Todo o programa é mantido no disco, dividido em páginas de tamanho fixo.
  - Um subconjunto dessas páginas é mantido na memória principal.
  - Se um programa usar as páginas de memória com muita frequência, serão geradas poucas falhas de página, e o programa vai rodar rapidamente.

Memória Virtual e Memória Cache

- No caso de emprego da memória cache:
  - Todo programa é mantido na memória principal, dividido em blocos da cache de tamanho fixo.
  - Um subconjunto desses blocos deve ser mantido na cache.
  - Se um programa usar os blocos que estiverem na cache com muita frequência, serão geradas poucas falhas no acesso a cache e o programa vai rodar rápido.

Memória Virtual e Memória Cache

- Conceitualmente, essas duas memórias são idênticas; a única diferença é que elas atuam em níveis diferentes de hierarquia.
- As principais diferenças entre elas são:
  - As falhas no acesso à cache são tratadas por hardware, enquanto as falhas de páginas são tratadas pelo Sistema Operacional.
  - Os blocos da cache são muito menores do que as páginas.