GSI018 – Sistemas Operacionais

Memória Virtual

Universidade Federal de Uberlândia
Faculdade de Computação
Prof. Dr. rer. nat. Daniel D. Abdala

Na Aula Anterior...

- Conceitos Geirais;
- Mapas de Bits;
- Listas Livres;
- Algoritmos de alocação de memória.

Nesta Aula

- Introdução ao mapeamento de memória;
- Overlays;
- O conceito de memória virtual;
- Paginação;
- Tabela de Páginas;
- TLB – Translation Lookaside Buffer.

Introdução ao Mapeamento de Memória

- Em sistemas de multiprogramação os requisitos de memória são muito maiores do que a memória efetivamente disponível no sistema;
- Solução: não manter todo o programa/dados na memória;

Overlays

- O problema dos programas serem maiores que a memória física é quase tão antigo quanto a computação;
- Por volta de 1960, computadores começaram a ser utilizados para modelagem e simulação de sistemas inacreditavelmente complexos:
  - Modelagem de aviões;
  - Simulação de explosões nucleares;
  - etc...
- Os programas eram maiores do que a memória disponível;
- Sistemas Operacionais nesta época eram em geral monotarefa e em lote;

Overlays

- O conceito de overlays (sobreposições) foi criado;
- A ideia era quebrar o programa em overlays (pedaços);
- Naturalmente, uma cópia completa do programa/dados deve existir em algum lugar, p. ex. o disco rígido;
- Ao iniciar o programa, apenas um artefato de software chamado gerenciador de sobreposições – GS – era carregado;
- A seguir o GS carregava o overlay 0 e o começava a executar;
- Caso dado ou instrução de outro overlay fosse acessada e não estivesse na memória, o overlay correspondente era carregado na memória;
- Caso não houvesse memória disponível para o carregamento, uma política de sobreposição era adotada para substituir um dos overlays já na memória;

Overlays

- Separar um programa em overlays é um trabalho tedioso e muito propenso a erros;
- Poucos programadores eram bons em organizar programas em overlays;
- Não demorou muito para que se começasse a pensar em relegar esta tarefa ao computador;
- A forma de fazer isso, era colocar o gerenciador de overlays como parte do SO, e algumas outras mudanças que deram origem à memória virtual;

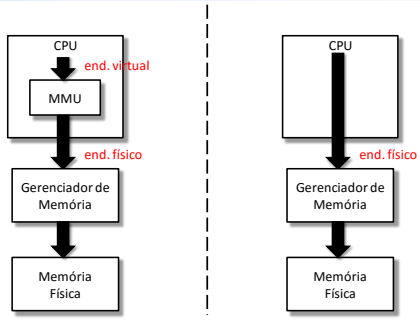
7

Memória Virtual

- Os endereços de memória utilizados pelo programa são virtuais, ou seja, precisam ser traduzidos para representarem um endereço físico de memória;
- O espaço de endereçamento do programa é todo dividido em páginas;
- A memória física é subdividida em molduras de mesmo tamanho que a página;
- Um dispositivo de hardware adicional MMU – Memory Management Unit – é responsável pela tradução de um endereço virtual em físico;

8

Memória Virtual



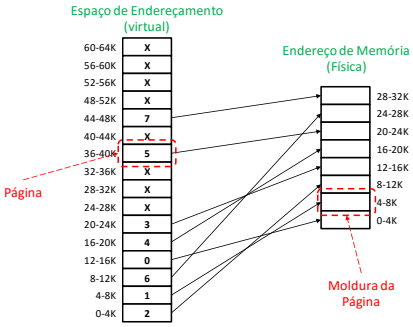
9

Memória Virtual

- Para a memória virtual funcionar, é essencial definir um tamanho de página;
- NO LINUX: `$getconf PAGESIZE`
- Requer um dispositivo de hardware adicional chamado MMU – Memory Management Unit;

10

Exemplo Memória Virtual



11

Exemplo Memória Virtual

- Considere a seguinte instrução:

```
lw $a0, 0($s0)  #s0 contém 0x00000030
```

- A semântica da instrução dita:

```
$a0 ← MEM[0+48]
```

- Consultando a figura acima, observa-se que o endereço virtual 48 encontra-se na página 2 que é mapeada para a moldura 8-12K;
- A MMU mapeia então o endereço virtual a partir do limite inferior da moldura:

$$Addr_{físico} = \underbrace{(Addr_{virtual} - base_{virtual})}_{\text{deslocamento}} + base_{pageFrame}$$

12

Exemplo Memória Virtual

```
li    $s0, 42
addi  $t0, 0xb000
sw    $s0, 0($t0)
addi  $s1, $s0, 42
sw    $s1, 4102($t0)
```

- O primeiro sw gera a seguinte tradução de endereços:

```
Página = 7
  (virt(45056~49148 – 0xb000~0xbffc)
   fisc(28672~32764 – 0x7000~0x7ffc))
Addr_fisico=(0xb000-0xb000)+ 0x7000 = 0x7000
MEM[0x7000] = $s0
```

- O segundo sw, pula uma página, pois 4102 é maior que 4096 (tamanho da página/frame);

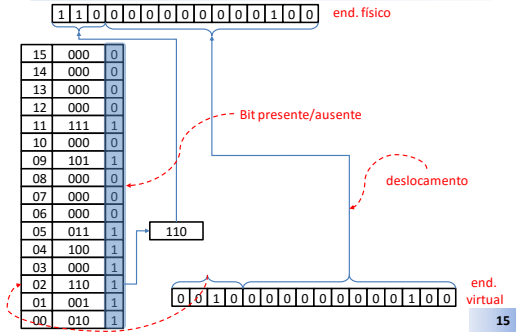
13

Paginação

- Controla:
  - que páginas da memória virtual estão carregadas na memória física;
  - Detecta se a página de um endereço está na memória;
  - Define as políticas de substituição de páginas na memória caso não haja mais espaço livre na memória física;

14

Tabela de Páginas



15

Paginação

- Dois problemas:
  - O mapeamento virtual→físico deve ser rápido;
  - Se o espaço de endereçamento for grande a tabela de páginas será grande;

16

Exemplo

- Quantas páginas terá uma tabela de paginação considerando que o espaço de endereçamento seja de 32 bits e o tamanho da página seja de 4K bytes?

```
32 bits → 4.294.967.296 endereços
        → 4 GB endereços
Página  → 4KB = 4096 endereços
4GB/4KB → 1M = 1.048.576 páginas
```

17

Exemplo

- Quantas páginas terá uma tabela de paginação considerando que o espaço de endereçamento seja de 32 bits e o tamanho da página seja de 16K bytes?

```
32 bits → 4.294.967.296 endereços
        → 4 GB endereços
Página  → 16KB = 16.384 endereços
4GB/16KB → 256K = 262.144 páginas
```

18

TLB – Translation Lookaside Buffer

- Tabelas de páginas são grandes;
- Elas devem ser mantidas na memória;
- A MMU precisa acessar a memória para consultar a entrada na tabela requisitada e então calcular o endereço físico de uma instrução de acesso a memória;
- 1 acesso a memória no programa → 2 acessos a memória devido a memória virtual;
- Uma solução para este problema e manter em dentro da MMU um buffer contendo as páginas mais utilizadas;
- Memória associativa → TLB

19

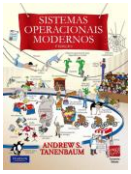
Válida	Página virtual	Modificada	Proteção	Moldura da página
1	140	1	RW	31
1	20	0	RX	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	RX	50
1	21	0	RX	45
1	860	1	RW	14
1	861	1	RW	75

Exemplo de TLB para acelerar a paginação

(Imagem retirada do TANEMBAUM, A. S. Sistemas Operacionais Modernos, 2ª Edição, Pearson, 2003.)

20

Bibliografia - Básica



- 3ª Edição
- Páginas 114-121

21