

LPRM
Laboratório de Pesquisa em Redes e Multimídia

UNIX – Gerência de Memória

Universidade Federal do Espírito Santo
Departamento de Informática

LPRM
Laboratório de Pesquisa em Redes e Multimídia

Introdução

- Unix implementado sobre muitos computadores diferentes
 - baseada, segmentada, paginada, segmentada/paginada
- A gerência de memória garante:
 - Proteção do espaço de endereçamento
 - Permitir modificação dinâmica do espaço de endereçamento
- Existem ainda requisitos de desempenho:
 - Manter em memória o maior número de processos
 - Minimizar as transferências entre disco e memória
- Dois grupos de implementações:
 - Transferência (Swapping) nas arquiteturas baseadas e segmentadas
 - Paginação nas arquiteturas paginadas e segmentadas/paginadas

Profª. Patrícia D. Costa LPRM/DI/UFES 2 Sistemas Operacionais 2008/1

LPRM
Laboratório de Pesquisa em Redes e Multimídia

Transferência (*Swapping*) (1)

- Arquiteturas baseadas
 - Processos são carregados na memória física por inteiro
- Arquiteturas segmentadas:
 - regiões (texto, dados, stack) carregadas de forma contínua em memória
- Transfere para disco processos que estejam bloqueados ou com menor prioridade

Physical Memory

Swap Area on Disk

Operating System
P0
P1
P2

t = 10

Operating System
P3
P1
P2

t = t1

Operating System
P3
P0
P2

t = t2

unused memory

Profª. Patrícia D. Costa LPRM/DI/UFES 3 Sistemas Operacionais 2008/1

LPRM
Laboratório de Pesquisa em Redes e Multimídia

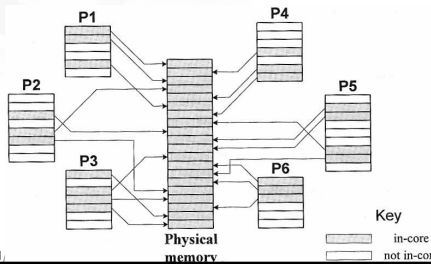
Transferência (*Swapping*) (2)

- Existem 4 casos que provocam a transferência:
 - chamada fork - é preciso espaço para o novo processo
 - chamada brk - expande o segmento de dados do processo
 - A função malloc da biblioteca em C deflagra brk se não houver espaço livre para satisfazer a solicitação
 - crescimento natural do stack
 - sistema operacional precisa de espaço para carregar em memória um processo que estava swapped-out
- Principal desvantagem
 - Apenas um pequeno número de processos podem encontrar-se simultaneamente na memória física

Profª. Patrícia D. Costa LPRM/DI/UFES 4 Sistemas Operacionais 2008/1

Paginação – Conceitos Gerais (1)

- A introdução de máquina de 32 bits de espaço de endereçamento, especialmente o VAX-11/780 (1978)
 - Promoveu a oportunidade do Unix expandir seus serviços de memória virtual
- 3BSD foi a primeira versão do UNIX a suportar paginação sob demanda
- A partir de 1980, a maioria dos UNIXs suportam memória virtual paginada



Paginação – Conceitos Gerais (2)

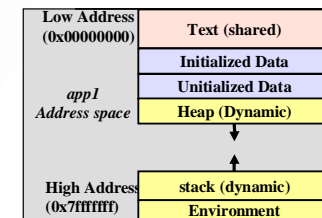
- Requisitos funcionais para o sub-sistema de memória
 - Gerência do espaço de endereçamento
 - O kernel aloca espaço de end. para um novo processo durante o *fork*, e desaloca quando o processo *exits*
 - No caso de um *exec*, o kernel libera o antigo espaço de end. alocando um novo p/ o novo programa
 - São comuns operações p/ mudar o tamanho das regiões de pilha, dados e adicionar novas regiões (e.g. memória compartilhada)
 - Tradução de endereços
 - Usando-se a MMU (*address translation maps*)
 - Na ocorrência de *page faults*, o *fault handler* do kernel deve tratá-la (ex: trazendo a nova página p/ memória)
 - Gerência da memória física
 - O sistema a utiliza como uma "cache"
 - O kernel deve garantir uso otimizado e consistência

Paginação – Conceitos Gerais (3)

- Requisitos funcionais para o sub-sistema de memória (cont.)
 - Proteção de memória
 - O kernel implementa proteção usando os mecanismos de hardware disponíveis.
 - Se ele detecta uma tentativa de acesso a uma localização ilegal, ele notifica o processo através de um sinal (SIGSEGV)
 - Memória compartilhada
 - Exemplos:
 - processos executando um mesmo programa podem compartilhar uma mesma região de texto;
 - bibliotecas compartilhadas
 - **Em algumas implementações**, após um *fork*, pai e filho compartilham região de dados enquanto não é feita nenhuma alteração na mesma
 - Aumenta o desempenho da gerência de memória paginada
 - Monitoramento da carga do sistema
 - A carga depende do número de processos, do tamanho dos mesmos, e do padrão de acesso à memória

Gerência do Espaço de Endereçamento Virtual (1)

- O espaço de endereçamento de um processo organiza-se em regiões de memória
 - área de texto (código do programa)
 - área de dados, inicializados ou não (variáveis alocadas estaticamente)
 - pilha (variáveis automáticas ou locais, passagem de parâmetros, salvar e restaurar endereços de retorno)
 - Além disso: *heap*, *shared memory*, *shared libraries*
- Cada região tem uma tabela de páginas própria



Gerência do Espaço de Endereçamento Virtual (2)

- A páginas se diferem!
 - Proteção
 - Páginas da área de texto são em geral *read-only*, enquanto pilha, dados e *heap* são *read-write*
 - Inicialização
 - Código e dados inicializados: lidos do arquivo executável
 - Dados ã inicializados: *zero filled*
 - u area e kernel stack: têm suas páginas copiadas do pai c/ alguns elementos alterados
 - *Shared pages* podem conter tanto dados quanto código
 - Dados podem ser compartilhados apenas até serem modificados

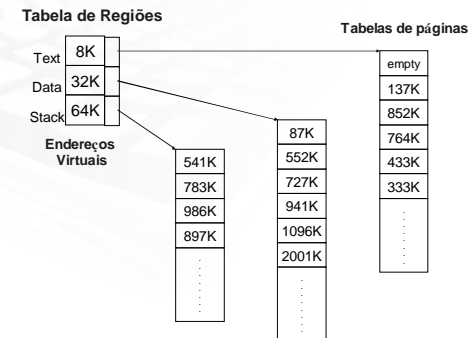
Algoritmos de Substituição

- A maioria dos UNIX utiliza política global de substituição
 - É garantido um número mínimo de páginas residentes por processo
- São usadas variantes do LRU
- Quando liberar páginas da memória?
 - Na hora em que ocorre um *page fault*: solução ineficiente que degrada o desempenho do sistema.
 - Em geral, os kernels implementam um esquema onde as páginas são periodicamente liberadas e colocadas em um pool de páginas livres "free page list" (molduras)
 - *Prepaging*: as páginas do *working set* são trazidas para memória antes que o processo seja escalonado

Requisitos de Hardware

- O subsistema de gerência de memória se apóia no hardware para a execução de várias tarefas
 - MMU
- Tradução de endereços
 - Tabelas de páginas
 - Translation Lookaside Buffer (TLB)
 - O hardware dita o formato dessas estruturas, mas o S.O. é responsável pelo set up e a manutenção das mesmas
- Tabelas de páginas
 - uma tabela de páginas para endereços de kernel
 - uma ou mais tabelas de páginas para cada processo
 - Em geral cada região/área tem uma tabela de páginas própria

Tabela de Páginas por Região (1)



Lprm Laboratório de Pesquisa em Redes e Multimídia

Tabela de Páginas por Região (2)

O que ocorre após um fork()?

Prof.: Patrícia D. Costa LPRM/DI/UFES 13 Sistemas Operacionais 2008/1

Lprm Laboratório de Pesquisa em Redes e Multimídia

Estruturas de Dados para Gerência de Memória

Prof.: Patrícia D. Costa LPRM/DI/UFES 14 Sistemas Operacionais 2008/1

Lprm Laboratório de Pesquisa em Redes e Multimídia

Page Faults

- Faltas de Presença
 - A página não está presente na tabela de páginas (ñ há PTE)
- Faltas de Validação
 - PTE marcada como inválida (página ñ residente na memória)
- Faltas de Proteção
 - O acesso pretendido não está de acordo com a proteção definida para a página
- Na ocorrência de um *Page Fault*, a MMU gera uma interrupção, e o controle é transferido para o respectivo *handler* no kernel
 - O handler pode:
 - tentar resolver o problema (ex: trazendo a página para a memória)
 - notificar o processo enviando um sinal (SIGSEGV)
 - Poderá eventualmente ser tratado pelo processo, por exemplo, aumentando a região de dados.

Prof.: Patrícia D. Costa LPRM/DI/UFES 15 Sistemas Operacionais 2008/1

Lprm Laboratório de Pesquisa em Redes e Multimídia

Memória Virtual do 4.3 BSD

- Baseada no VAX-11 (32 bits c/ páginas de 512 bytes)
- Três estruturas de dados principais
 - *Core map* (tabela de frames da memória física)
 - *Page tables* (tabelas de páginas p/ tradução)
 - *Disk maps*
 - Mapeia páginas do espaço de endereçamento virtual em blocos de discos da área de swap
 - Localizados na u-area

Prof.: Patrícia D. Costa LPRM/DI/UFES 16 Sistemas Operacionais 2008/1

Lprm Laboratório de Pesquisa em Redes e Multimídia UFES

Memória Virtual do 4.3 BSD

Core map

- Paged pool** descrito como um array de estruturas **cmap**:
 - Identificação < owner(pid), type, virtual page number >
 - Free list pointers <next, prev> (mantido em LRU aproximado)
 - Text page cache <device, block number >
 - Synchronization flags

Profª. Patrícia D. Costa LPRM/DI/UFES 17 Sistemas Operacionais 2008/1

Lprm Laboratório de Pesquisa em Redes e Multimídia UFES

Memória Virtual do 4.3 BSD

Virtual Address Space

- P0 (região de programa): user text e data
- P1 (região de controle): *user stack, kernel stack, u area*
- S0 (região de sistema): kernel text e data
- Reserved: não suportada

Data e stack podem crescer "livremente"

Profª. Patrícia D. Costa LPRM/DI/UFES 18 Sistemas Operacionais 2008/1

Lprm Laboratório de Pesquisa em Redes e Multimídia UFES

Memória Virtual do 4.3 BSD

Tabelas de Páginas (1)

- O hardware do VAX suporta diretamente tabelas de páginas para tradução
- A *proc structure* possui uma descrição incluindo a localização e o tamanho de cada tabela de página do processo
- Existe uma única tabela de páginas de sistema
 - Ela encontra-se de forma contígua na memória física
- Cada processo possui 2 tabelas de páginas: para mapear P0 e P1
 - Elas encontram-se de forma contígua no espaço de endereçamento virtual do kernel
- Cada tabela de páginas é definida por um par de registradores
 - base address register
 - length register

Profª. Patrícia D. Costa LPRM/DI/UFES 19 Sistemas Operacionais 2008/1

Lprm Laboratório de Pesquisa em Redes e Multimídia UFES

Memória Virtual do 4.3 BSD

Tabelas de Páginas (2)

Profª. Patrícia D. Costa LPRM/DI/UFES 20 Sistemas Operacionais 2008/1

Memória Virtual do 4.3 BSD

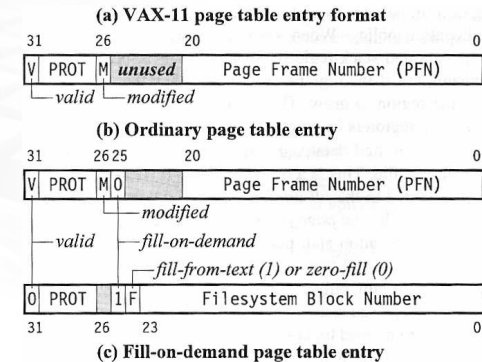
Tradução de Endereços

- A tradução de um endereço virtual dentro do espaço de endereçamento do processo envolve dois acessos à memória
 - O primeiro, para a tabela de páginas de sistema, para calcular o endereço físico da tabela de páginas do processo
 - O segundo, para a tabela de páginas (de uma das regiões) para calcular o endereço físico do elemento especificado
- Troca de contexto
 - Os registradores *base address* e *length* das tabelas de páginas de P0 e P1 são atualizados
 - TLB é dividida em duas seções
 - Uma para tradução de endereços de sistema
 - Outra (*per-process section*) para tradução de endereços de processo
 - Somente a *per-process section* é "descarregada" durante a troca de contexto

Memória Virtual do 4.3 BSD

Estados das Páginas

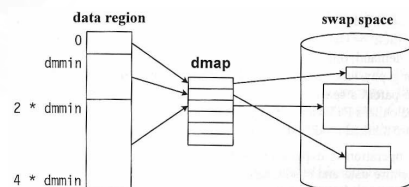
- Resident
- Fill-on-demand
 - Fill-from-text
 - Zero-fill
- Swapped out
- Estados codificados nas PTEs



Memória Virtual do 4.3 BSD

Área de Swap (dmap- disk map)

- Quando um processo é criado o kernel aloca espaço na área de swap para os dados e pilhas
- As páginas de *text* (e dados não modificados) teoricamente não precisam de ser "swapped out"
 - No BSD, elas também são colocadas em swap, para evitar a recomputação do *file system block number*
- Alocação de área de swap controlada em estruturas *dmap*
 - Cada região possui uma *dmap* de tamanho fixo



Memória Virtual do 4.3 BSD

Free Page List (1)

- *Page daemon*
 - Processo que mantém a lista de frames livres
 - Acorda periodicamente para checar o número de page frames livres (indicado pela variável *freemem*):
 - $minfree \leq freemem \leq maxfree$
 - Em geral mantém-se $freemem = 1/4$ da memória
 - Se for inferior, o page daemon transfere páginas do disco para a memória
 - O critério de substituição padrão é o NRU, implementado através do algoritmo "two handed clock" (similar ao Algoritmo do Relógio)
 - A tabela *cmap[]* é tratada como circular
 - São mantidos dois ponteiros que avançam juntos
 - O ponteiro da frente zera o *referenced bit*
 - O ponteiro de trás verifica o *reference bit*

Lprm Laboratório de Pesquisa em Redes e Multimídia

Memória Virtual do 4.3 BSD

Free Page List (2)

End of page list

Beginning of page list

hand

backhand

Core Map

Prof. Patrícia D. Costa LPRM/DI/UFES 25 Sistemas Operacionais 2008/1

Lprm Laboratório de Pesquisa em Redes e Multimídia

Discussão

- Tamanho de página reduzido (512 bytes).
- Quantidade reduzida de regiões
 - Normalmente necessita-se de 3 segmentos (código/dados/pilha), VAX/VMS tem 2.
 - Código e dados na mesma região
 - Pouco modular
- S.O. completamente amarrado ao hardware
- Não há suporte para memória ou biblioteca compartilhadas
- Sem suporte para "memory mapped files"

Prof. Patrícia D. Costa LPRM/DI/UFES 26 Sistemas Operacionais 2008/1

Lprm Laboratório de Pesquisa em Redes e Multimídia

Memory mapped files (1)

disk

page tables

A

B

buffer cache

pageable physical memory

Dois processos lendo uma mesma página de um arquivo

Prof. Patrícia D. Costa LPRM/DI/UFES 27 Sistemas Operacionais 2008/1

Lprm Laboratório de Pesquisa em Redes e Multimídia

Memory mapped files (2)

page tables

A

B

disk

pageable physical memory

Prof. Patrícia D. Costa LPRM/DI/UFES 28 Sistemas Operacionais 2008/1