UNIVALI

Universidade do Vale do Itajaí
Escola Politécnica

Memória Principal

1

Gerência de Memória

- Conceitos
- Alocação de memória contínua
- Paginação
- Estrutura da tabela de páginas
- Swapping

2

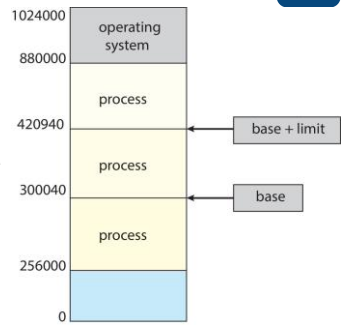
Conceitos

- O programa deve ser trazido (do disco) para a memória e deve ser transformado em um processo para que ele seja executado
  - A memória principal e os registradores são apenas o armazenamento que a CPU pode acessar diretamente
- A unidade de memória só vê um fluxo de:
  - endereços + solicitações de leitura, ou
  - endereço + dados e solicitações de gravação
- O acesso de registrador é feito em um clock da CPU (ou menos)
- A memória principal pode levar muitos ciclos, causando uma parada
- O cache fica entre a memória principal e os registradores da CPU
  - Proteção da memória necessária para garantir o funcionamento correto

3

Proteção

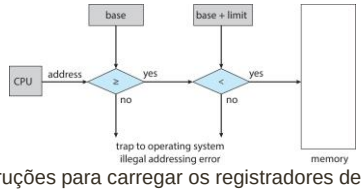
- Necessidade de proteger para que um processo só possa acessar os endereços em seu espaço de endereço
- Podemos fornecer essa proteção usando um par de registradores de base e limite para definir o espaço de endereço lógico de um processo



4

Proteção de endereço de hardware

- A CPU deve verificar cada acesso à memória gerado no modo de usuário para ter certeza de que está entre base e limite para esse usuário



- As instruções para carregar os registradores de base e limite são privilegiadas (só o SO e em Assembly)

5

Linkagem de endereço

- Programas em disco, prontos para serem trazidos para a memória para executar formam uma fila de entrada
  - Se sem configuração, devem ser carregado no endereço 0000
- Ruim ter o processo do usuário sendo carregado para o endereço físico 0000
  - Qual o motive de não pode ser?
- Endereços representados de diferentes maneiras em diferentes fases da vida de um programa
- Endereços de código-fonte geralmente simbólicos
  - Endereços de código compilados se associam a endereços relocáveis
    - Ex: "14 bytes a partir do início deste módulo"
  - Linker ou loader vinculará endereços relocáveis a endereços absolutos
    - Ex: 74014
- Cada ligação mapeia um espaço de endereço para outro

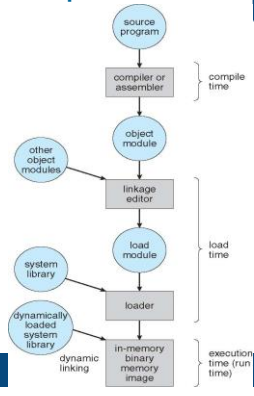
6

Linkagem de instruções e dados à memória

- A vinculação de instruções e dados a endereços de memória pode ocorrer em três estágios diferentes
  - Tempo de compilação: Se o local da memória conhecido a priori, o código absoluto pode ser gerado; deve recompilar o código se o local de início for alterado
  - Tempo de carregamento: Deve gerar código relocável se o local da memória não for conhecido em tempo de compilação
  - Tempo de execução: Vinculação atrasada até o tempo de execução se o processo puder ser movido durante sua execução de um segmento de memória para outro
    - Precisa de suporte de hardware para mapas de endereços (por exemplo, registradores de base e limite)

7

Processamento em várias etapas de um programa de usuário



8

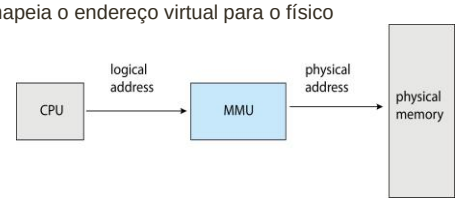
Espaço de endereço lógico vs. físico

- O conceito de um espaço de endereço lógico que está vinculado a um espaço de endereço físico separado é fundamental para o gerenciamento adequado da memória
  - Endereço lógico – gerado pela CPU; também conhecido como endereço virtual
  - Endereço físico – endereço visto pela unidade de memória
    - Os endereços lógicos e físicos são os mesmos em esquemas de vinculação de endereço em tempo de compilação e tempo de carregamento; os endereços lógicos (virtuais) e físicos diferem no esquema de vinculação de endereço em tempo de execução
- Espaço de endereço lógico é o conjunto de todos os endereços lógicos gerados por um programa
- Espaço de endereço físico é o conjunto de todos os endereços físicos gerados por um programa

9

Memory-Management Unit (MMU)

- Dispositivo de hardware que, em tempo de execução, mapeia o endereço virtual para o físico



- Muitos métodos possíveis

10

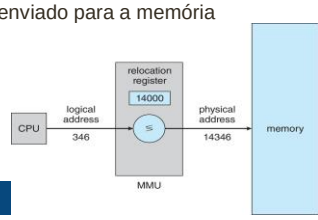
Memory-Management Unit (MMU)

- Considere um esquema simples. que é uma generalização do esquema de registrador de base.
- O registrador base agora chamado de registrador de offset
- O valor no registrador de offset é adicionado a cada endereço gerado por um processo de usuário no momento em que é enviado para a memória
- O programa de usuário lida com endereços lógicos; ele nunca vê os endereços físicos reais
  - A linkagem em tempo de execução ocorre quando é feita referência ao local na memória
  - Endereço lógico associado a endereços físicos

11

Memory-Management Unit (MMU)

- Considere um esquema simples. que é uma generalização do esquema de registrador de base.
- O registrador base agora chamado de registro de ofsset
- O valor no registrador de offset é adicionado a cada endereço gerado por um processo de usuário no momento em que é enviado para a memória



12

Carregamento dinâmico

- Todo o programa precisa estar na memória para ser executado
- A rotina não é carregada até que seja chamada
- Melhor utilização do espaço de memória
  - A rotina não utilizada nunca é carregada
- Todas as rotinas mantidas em disco em formato de carga realocável
- Útil quando grandes quantidades de código são necessárias para lidar com casos que ocorrem com pouca frequência
- Nenhum suporte especial do sistema operacional é necessário
  - Implementado através do design do programa
  - O sistema operacional pode ajudar fornecendo bibliotecas para implementar o carregamento dinâmico

13

Linkagem dinâmica

- Vinculação estática – bibliotecas do sistema e código de programa combinados pelo carregador na imagem do programa binário
- Vinculação dinâmica – vinculação adiada até o momento da execução
  - Pequeno pedaço de código, *stub* ou esboço, usado para localizar a rotina apropriada da biblioteca residente na memória
- O Stub substitui-se pelo endereço da rotina e executa a rotina
- O SO verifica se a rotina está no endereço de memória dos processos
  - Se não estiver no espaço de endereço, adicione ao espaço de endereço
- A vinculação dinâmica é particularmente útil para bibliotecas
- Sistema também conhecido como bibliotecas compartilhadas
- Considere a aplicabilidade para corrigir bibliotecas do sistema
  - O controle de versão pode ser necessário

14

Alocação contígua

- A memória principal deve suportar os processos do sistema operacional e do usuário
- Recurso limitado, deve alocar de forma eficiente
- A alocação contígua é um dos primeiros métodos
- Memória principal geralmente em duas partições:
  - Sistema operacional geralmente mantido em pouca memória com vetor de interrupção
  - Processos do usuário, em seguida, mantidos em alta memória
  - Cada processo contido em uma única seção contígua da memória

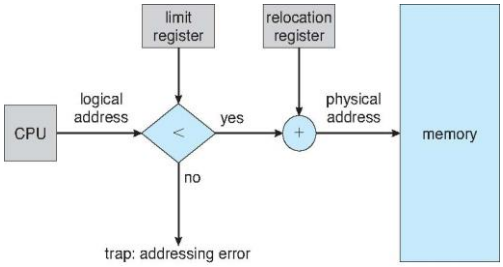
15

Alocação contígua

- Registradores de realocação usados para proteger os processos do usuário uns dos outros e contra a alteração do código e dos dados do sistema operacional
  - O registrador base contém o valor do menor endereço físico
  - O registrador de limite contém um intervalo de endereços lógicos – cada endereço lógico deve ser menor que o registrador de limite
  - MMU mapeia o endereço lógico dinamicamente
  - Pode então permitir ações como o código do kernel sendo transitório e o kernel mudando de tamanho

16

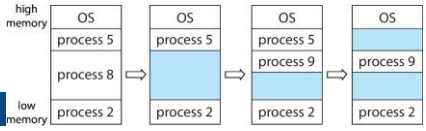
Suporte de hardware para realocação e registrador de limite



17

Partição variável

- Alocação de várias partições
  - Grau de multiprogramação limitado pelo número de partições
  - Tamanhos de partição variável para eficiência (dimensionados de acordo com as necessidades de um determinado processo)
  - Buraco – bloco de memória disponível; buracos de vários tamanhos estão espalhados por toda a memória
  - Quando um processo chega, ele recebe memória de um parte grande o suficiente para acomodá-lo
  - Processo de saída libera sua partição e partições livres adjacentes combinadas
  - O sistema operacional mantém informações sobre:
    a) partições alocadas b) partições livres (buracos)



18

Problema de armazenamento com alocação dinâmica

- Como satisfazer um pedido de tamanho n de uma lista de buracos (memória) livres?
  - First-fit: Aloque o primeiro local que é grande o suficiente
  - Best-fit: Aloque o menor local que seja grande o suficiente; deve pesquisar a lista inteira, a menos que ordenado por tamanho
    - Resulta em um o menor orifício restante
  - Worst-fit: Alocar o maior espaço; também deve pesquisar a lista inteira
    - Produz o maior espaço não alocado
- First-fit e best-fit são melhores do que worst-fit em termos de velocidade e utilização do armazenamento

19

Fragmentação

- Fragmentação externa – existe espaço total de memória para satisfazer uma solicitação, mas não é contíguo
- Fragmentação interna – a memória alocada pode ser um pouco maior do que a memória solicitada; essa diferença de tamanho é a memória interna a uma partição, mas não está sendo usada
- A primeira análise de ajuste revela que, dados os N blocos alocados, 0,5 N blocos perdidos para a fragmentação
  - 1/3 pode ser inutilizável -> regra de 50%

20

Fragmentação

- Reduzir a fragmentação externa por compactação
  - Embaralhe o conteúdo da memória para colocar toda a memória livre junta em um grande bloco
  - A compactação só é possível se a realocação for dinâmica e for feita no momento da execução
  - Problema de E/S
  - Operação trava na memória enquanto ela está envolvida na E/S
    - Fazer E/S somente em buffers do sistema operacional
- Agora considere que o armazenamento de backup tem os mesmos problemas de fragmentação

21

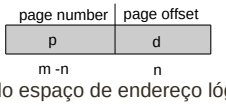
Paginação

- O espaço de endereço físico de um processo pode ser não contíguo; o processo é alocado na memória física sempre que há disponibilidade
  - Evita a fragmentação externa
  - Evita o problema de blocos de memória de tamanho variável
- Divida a memória física em blocos de tamanho fixo chamados quadros (frames)
  - Tamanho é potência de 2, entre 512 bytes e 16 Mbytes
- Divida a memória lógica em blocos do mesmo tamanho chamados páginas
- Acompanhe todos os quadros livres
- Para executar um programa de tamanho N páginas, precisa encontrar N quadros livres e carregar o programa
- Configurar uma tabela de páginas para converter endereços lógicos em físicos
- O armazenamento de backup também é dividido em páginas
- Ainda tem fragmentação interna

22

Esquema de Tradução de Endereço

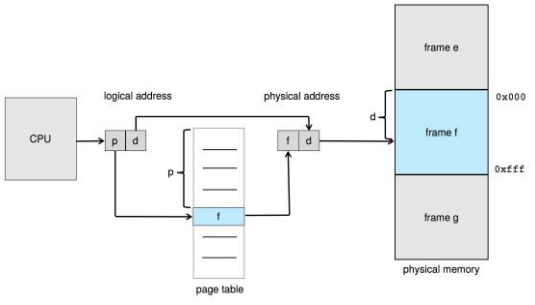
- O endereço gerado pela CPU é dividido em:
  - Número de página (p) – usado como um índice em uma tabela de página que contém o endereço base de cada página na memória física
  - Deslocamento (offset) de página (d) – combinado com o endereço base para definir o endereço de memória física que é enviado para a unidade de memória



- Para determinado espaço de endereço lógico 2^m e tamanho de página 2^n

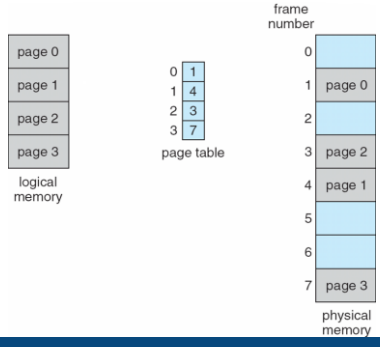
23

Hardware de paginação



24

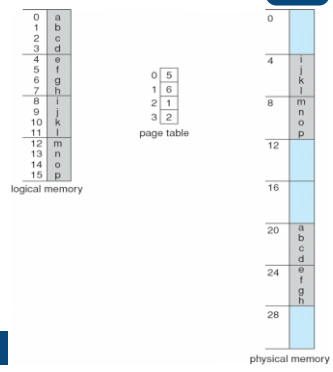
Modelo de Paginação das Memória Lógica e Física



25

Exemplo de paginação

- Endereço lógico: n = 2 e m = 4. Usando um tamanho de página de 4 bytes e uma memória física de 32 bytes (8 páginas)



26

Paginação – Calculando a fragmentação interna

- Tamanho da página = 2.048 bytes
- Tamanho do processo = 72.766 bytes
- 35 páginas + 1.086 bytes
- Fragmentação interna de 2.048 - 1.086 = 962 bytes
- Fragmentação no pior caso = 1 quadro – 1 byte
  - Nesse exemplo, 2.047 bytes
- Em média, fragmentação = 1/2 do tamanho do quadro
- Tamanhos de quadro estão pequenos o suficiente?
- Mas cada entrada de tabela de página usa memória para rastrear: não pode ser tão pequeno
- Os tamanhos das páginas crescem ao longo do tempo

27

Quadros (Frames) Livres

28

Implementação da tabela de páginas

- A tabela de páginas é mantida na memória principal
  - Page-table base register (PTBR) aponta para a tabela de páginas
  - Page-table length register (PTLR) indica o tamanho da tabela de páginas
- Nesse esquema, todo acesso a dados/instruções requer dois acessos à memória
  - Um para a tabela de páginas e outro para os dados/instrução
- O problema de dois acessos a memórias pode ser resolvido com o uso de um cache de hardware especial de pesquisa rápida chamado Translation Look-Aside Buffers (TLBs) (também chamada de memória associativa)

29

Translation Look-Aside Buffer

- Alguns TLBs armazenam identificadores de espaço de endereço (ASIDs) em cada entrada TLB - identifica exclusivamente cada processo para fornecer proteção de espaço de endereço para esse processo
  - Caso contrário, precisa liberar a cada troca de contexto
- TLBs tipicamente pequenas (64 a 1024 entradas)
- Em uma miss de TLB, o valor é carregado na TLB para acesso mais rápido na próxima vez
  - As políticas de substituição devem ser consideradas
  - Algumas entradas podem ser conectadas para acesso rápido permanente

30

Hardware

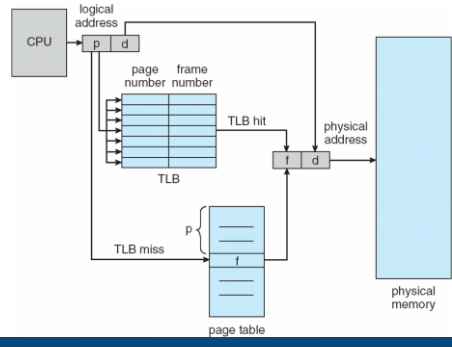
- Memória associativa – pesquisa paralela

Page #	Frame #

- Tradução de endereço (p, d)
  - Se p estiver no registrador associativo, obtenha o quadro # da memória física
  - Caso contrário, obtenha o quadro # da tabela de páginas na memória (dois acessos)

31

Hardware de paginação com TLB



32

Tempo de Acesso Efetivo

- Taxa de acertos – porcentagem de vezes que um número de página é encontrado na TLB
- Uma taxa de acerto de 80% significa que encontramos o número da página desejada na TLB 80% das vezes.
- Suponha que 10 nanossegundos para acessar a memória
  - Se encontrarmos a página desejada na TLB, um acesso à memória mapeada leva 10 ns
  - Caso contrário, precisamos de dois acessos à memória, por isso é 20 ns
- Tempo de Acesso Efetivo (EAT)

  EAT = 0,80 x 10 + 0,20 x 20 = 12 nanossegundos

  implicando 20% de lentidão no tempo de acesso
- Considere uma taxa de acerto mais realista de 99%

  EAT = 0,99 x 10 + 0,01 x 20 = 10,1 ns

  implicando apenas 1% de desaceleração no tempo de acesso

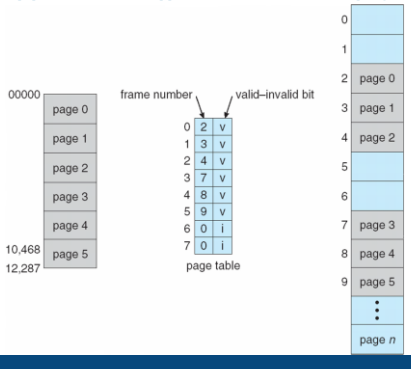
33

Proteção de memória

- Proteção de memória implementada associando o bit de proteção a cada quadro para indicar se o acesso somente leitura ou leitura/gravação é permitido
  - Também pode adicionar mais bits para indicar somente execução da página e assim por diante
- Bit válido-inválido anexado a cada entrada na tabela de páginas:
  - "válido" indica que a página associada está no espaço de endereço lógico do processo e, portanto, é uma página válida
  - "inválido" indica que a página não está no espaço de endereçamento lógico do processo
  - Ou usar page-table length register (PTLR)
- Quaisquer violações resultam em uma interrupção do tipo trap para o kernel

34

Bit válido (v) ou inválido (i) em uma tabela de página



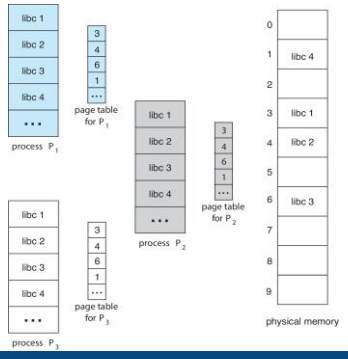
35

Páginas Compartilhadas

- Código compartilhado
  - Uma cópia de código somente leitura compartilhada entre processos (ou seja, editores de texto, compiladores)
  - Semelhante a vários threads compartilhando o mesmo espaço de processo
  - Também útil para comunicação entre processos se o compartilhamento de páginas de leitura/gravação for permitido
- Código privado e dados
  - Cada processo mantém uma cópia separada do código e dos dados
  - As páginas para o código privado e os dados podem aparecer em qualquer lugar no espaço de endereço lógico

36

Exemplo de páginas compartilhadas



37

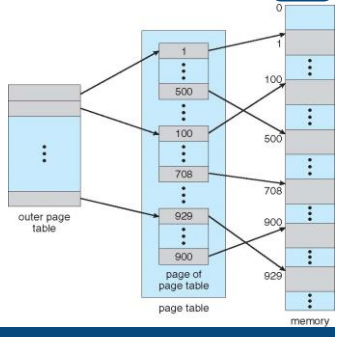
Estrutura da tabela de páginas

- As estruturas de memória para paginação podem ficar enormes usando métodos diretos
  - Considere um espaço de endereço lógico de 32 bits como em computadores modernos
  - Tamanho da página de 4 KB (2^{12})
  - A tabela de páginas teria 1 milhão de entradas ($2^{12}/2^{12}$)
  - Se cada entrada tiver 4 bytes -> cada processo precisa de 4 MB de espaço de endereço físico apenas para a tabela de páginas
    - Não quero alocar isso continuamente na memória principal
  - Uma solução simples é dividir a tabela de páginas em unidades menores
    - Paginação Hierárquica
    - Tabelas de página com hash
    - Tabelas de páginas invertidas

38

Tabelas de Páginas Hierárquicas

- Divida o espaço de endereço lógico em várias tabelas de páginas
- Uma técnica simples é uma tabela de página de dois níveis
- Em seguida, paginamos a tabela de páginas



39

Exemplo de Paginação em Dois Níveis

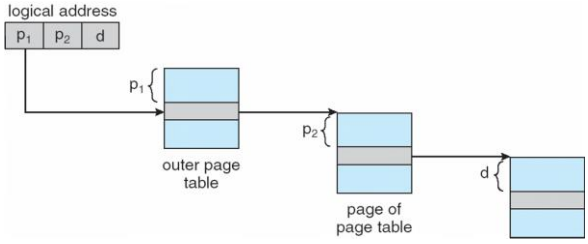
- Um endereço lógico (em uma máquina de 32 bits com tamanho de página de 1K) é dividido em:
  - um número de página consistindo de 22 bits
  - um deslocamento de página consistindo de 10 bits
- Como a tabela de páginas é paginada, o número da página é dividido em:
  - um número de página de 10 bits
  - um deslocamento de página de 12 bits
- Assim, um endereço lógico é o seguinte:

page number		page offset
p_1	p_2	d
10	10	12

- onde p_1 é um índice na tabela de página externa e p_2 é o deslocamento dentro da página da tabela de página interna
  - Conhecida como tabela de página mapeada para frente

40

Conhecida como tabela de página mapeada para frente



41

Espaço de endereço lógico de 64 bits

- Mesmo o esquema de paginação de dois níveis não é suficiente
  - Se o tamanho da página for 4 KB (2^{12})
  - Então a tabela de páginas tem 2^{52} entradas
  - Se for um esquema de dois níveis, as tabelas de páginas internas podem ter 2^{10} entradas de 4 bytes
  - Ex:

outer page	inner page	offset
p_1	p_2	d
42	10	12

  - Tabela de página externa tem 2^{42} entradas ou 2^{44} bytes
  - Uma solução é adicionar uma segunda tabela de página externa
  - Mas no exemplo a seguir, a tabela da segunda página externa ainda tem 2^{34} bytes de tamanho
    - E possivelmente 4 acessos à memória para chegar a um local de memória física

42

Esquema de Paginação de Três Níveis

outer page	inner page	offset
p_1	p_2	d
42	10	12

2nd outer page	outer page	inner page	offset
p_1	p_2	p_3	d
32	10	10	12

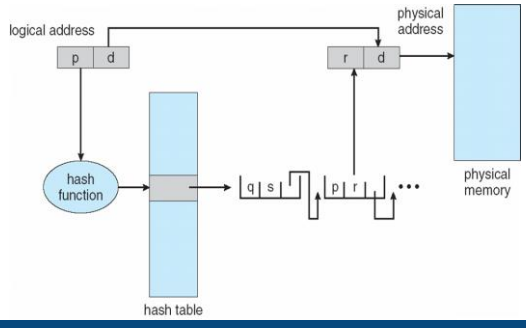
43

Tabelas de página com hash

- Comum em espaços de endereços > 32 bits
- O número da página virtual é hash em uma tabela de páginas
  - Esta tabela de página contém uma cadeia de hashing de elementos para o mesmo local
- Cada elemento contém (1) o número da página virtual (2) o valor do quadro da página mapeada (3) um ponteiro para o próximo elemento
- Números de páginas virtuais são comparados nesta cadeia em busca de uma correspondência
  - Se for encontrada uma correspondência, o quadro físico correspondente é extraído
- A variação para endereços de 64 bits são tabelas de páginas agrupadas
  - Semelhante ao hash, mas cada entrada refere-se a várias páginas (como 16) em vez de 1
  - Especialmente útil para espaços de endereço esparsos (onde as referências de memória são não contíguas e dispersas)

44

Tabela de página com hash



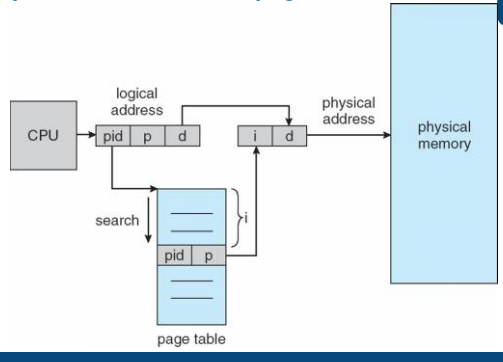
45

Tabela de página invertida

- Em vez de cada processo ter uma tabela de páginas e rastrear todas as páginas lógicas possíveis, rastreie todas as páginas físicas
- Uma entrada para cada página real da memória
- A entrada consiste no endereço virtual da página armazenada naquele local de memória real, com informações sobre o processo que possui essa página
- Diminui a memória necessária para armazenar cada tabela de página, mas aumenta o tempo necessário para pesquisar a tabela quando ocorre uma referência de página
- Use a tabela de hash para limitar a pesquisa a uma — ou no máximo algumas — entradas da tabela de páginas
  - TLB pode acelerar o acesso
- Mas como implementar a memória compartilhada?
  - Um mapeamento de um endereço virtual para o endereço físico compartilhado

46

Arquitetura de tabela de página invertida



47

Swapping

- Um processo pode ser trocado temporariamente da memória para um armazenamento secundário e, em seguida, trazido de volta à memória para execução contínua
  - O espaço total da memória física dos processos pode exceder a memória física
- Armazenamento de apoio – disco rápido grande o suficiente para acomodar cópias de todas as imagens de memória para todos os usuários; deve fornecer acesso direto a essas imagens de memória
- Roll out, roll in – variante de troca usada para algoritmos de escalonamento baseados em prioridade; o processo de prioridade mais baixa é trocado para que o processo de prioridade mais alta possa ser carregado e executado
- A maior parte do tempo de troca é o tempo de transferência; o tempo total de transferência é diretamente proporcional à quantidade de memória trocada
- O sistema mantém uma fila pronta de processos prontos para execução que possuem imagens de memória em disco

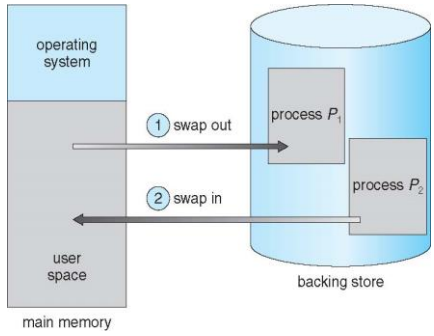
48

Swapping

- O processo trocado precisa trocar de volta para os mesmos endereços físicos?
- Depende do método de ligação de endereço
  - Além disso, considere E/S pendentes de/para o espaço de memória do processo
- Versões modificadas de troca são encontradas em muitos sistemas (ex: UNIX, Linux e Windows)
  - Swapping normalmente desativado
  - Iniciado se mais do que a quantidade limite de memória alocada
  - Desativado novamente quando a demanda de memória for reduzida abaixo do limite

49

Visão esquemática de Swapping



50

Tempo de troca de contexto, incluindo Swapping

- Se os próximos processos a serem colocados na CPU não estiverem na memória, é necessário trocar um processo e trocar no processo de destino
- O tempo de troca de contexto pode ser muito alto
- Troca de processo de 100 MB para disco rígido com taxa de transferência de 50 MB/s
  - Tempo de troca de 2000 ms
  - Mais troca de processo do mesmo tamanho
  - Tempo total do componente de troca de troca de contexto de 4000ms (4 segundos)
- Pode reduzir se reduzir o tamanho da memória trocada - sabendo quanta memória realmente está sendo usada
  - Chamadas do sistema para informar o SO sobre o uso da memória via request_memory() e release_memory()

51

Tempo de troca de contexto e Swapping

- Outras restrições também na troca
  - E/S pendente - não é possível fazer swapping, pois a E/S ocorreria no processo errado
  - Ou sempre transfira E/S para o espaço do kernel e depois para o dispositivo de E/S
    - Conhecido como buffer duplo, adiciona sobrecarga
- Troca padrão não usada em sistemas operacionais modernos
  - Mas versão modificada comum
    - Troque apenas quando a memória livre estiver extremamente baixa

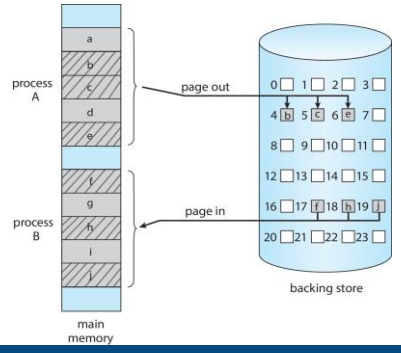
52

Swapping em sistemas móveis

- Normalmente não suportado
  - baseado em memória flash
    - Pequena quantidade de espaço
    - Número limitado de ciclos de gravação
    - Baixa taxa de transferência entre a memória flash e a CPU na plataforma móvel
- Em vez disso, use outros métodos para liberar memória se estiver baixa
  - O iOS pede que os aplicativos abandonem voluntariamente a memória alocada
    - Dados somente leitura descartados e recarregados do flash, se necessário
    - A falha na liberação pode resultar em rescisão
  - O Android encerra os aplicativos se houver pouca memória livre, mas primeiro grava o estado do aplicativo em flash para reinicialização rápida
  - Ambos os sistemas operacionais suportam paginação conforme discutido

53

Swapping com paginação



54

Universidade do Vale do Itajaí
Escola Politécnica

Fim

55