Universidade Federal
de Mato Grosso
Campus Rondonópolis

Sistemas Operacionais

Aula 11 - Gerenciamento de Memória (2)

Prof. Cleyton Slaviero

Recapitulando...

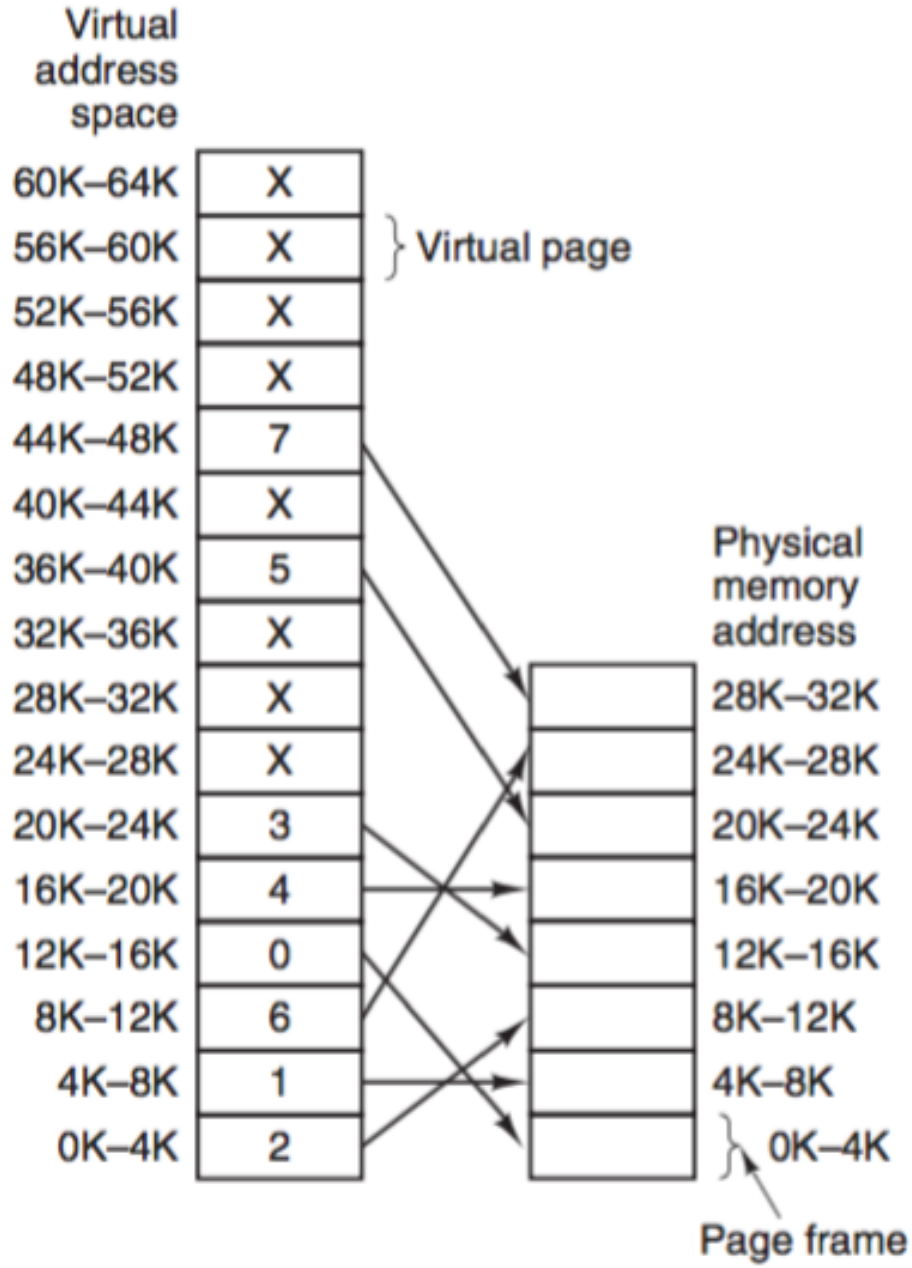
- Gerenciando memória
  - Manter o rastro, alocar e desalocar memória

- Swapping
  - Trocar processos na memória (swap in/out)
  - Problemas (tamanho – da memória e processos, alocação e desalocação,
  - Soluções (algoritmos *-fit, listas, mapa de bits)
- Memória virtual
  - Dividir a memória em partes (páginas); apenas algumas vão para a MPrincipal
  - MMU gerencia entrada e saída

Universidade Federal
de Mato Grosso
Campus Rondonópolis

Memória virtual

- Memória Virtual pode ser implementada quebrando-se o espaço de endereçamento virtual em páginas
- E mapeando cada página a alguma moldura de página na memória física
- Ou mantendo-a temporariamente não mapeada



Universidade Federal de Mato Grosso
Campus Rondonópolis

Paginação

- Memória Principal e Memória Secundária são organizadas em páginas de mesmo tamanho;
- Página é a unidade básica para transferência de informação;

- E como se dá o mapeamento?

Paginação

- Memória Principal e Memória Secundária são organizadas em páginas de mesmo tamanho;
- Página é a unidade básica para transferência de informação;

- E como se dá o mapeamento?
  - **Tabela de páginas**

Paginação – Tabela de páginas

- Tabela de páginas
  - Responsável por armazenar informações sobre as páginas virtuais
    - Pode ser visto como uma função:
      - argumento de entrada : número da página virtual;
      - argumento de saída (resultado) : número da página real (ou moldura de página page frame);
  - Cada processo tem sua própria tabela
    - Cada um tem seu próprio espaço de endereçamento
    - Cada um acha que começa em uma mesma posição

Universidade Federal de Mato Grosso
Campus Rondonópolis

Paginação – Tabela de páginas

- E como buscar um endereço?
  - Busca seqüencial? Binária?
    - Qualquer que seja a alternativa, é lenta
- Que fazer?

Universidade Federal
de Mato Grosso
Campus Rondonópolis

Paginação – Tabela de páginas

- E como buscar um endereço?
  - Busca seqüencial? Binária?
    - Qualquer que seja a alternativa, é lenta
- Que fazer?
  - Ideal: usar parte do endereço virtual como índice na tabela, onde está o endereço base da framecorrespondente na memória

Paginação – Funcionamento da Tabela de Páginas na MMU

- Usar parte do endereço virtual como índice na tabela, onde está o endereço base da frame correspondente na memória
- Como fazer isso?
  - Use tamanhos de página que sejam potências de 2
- 4K = 4096 = 2^{12}
  - 0k a 4k → 000000000000000 a 001000000000000
  - 4K a 8K → 001000000000000 a 010000000000000
  - 8K a 12K → 010000000000000 a 011000000000000
  - 12K a 16K → 011000000000000 a 100000000000000

Universidade Federal de Mato Grosso
Campus Rondonópolis

Paginação – Funcionamento da Tabela de Páginas na MMU

- E como buscar um endereço?
- E endereços dentro da página?
- Use os bits além do limite da página (em preto, no exemplo anterior)
  - Ex: 8196
    - Múltiplo de 4k mais próximo → 2 = 8192

    8196 = 8192 + 4

    (8192) 001000000000000 +

    (4) 000000000000100 =

    (8196) 001000000000100

  - Parte em vermelho: endereço base da página
  - Parte em preto: deslocamento (offset)

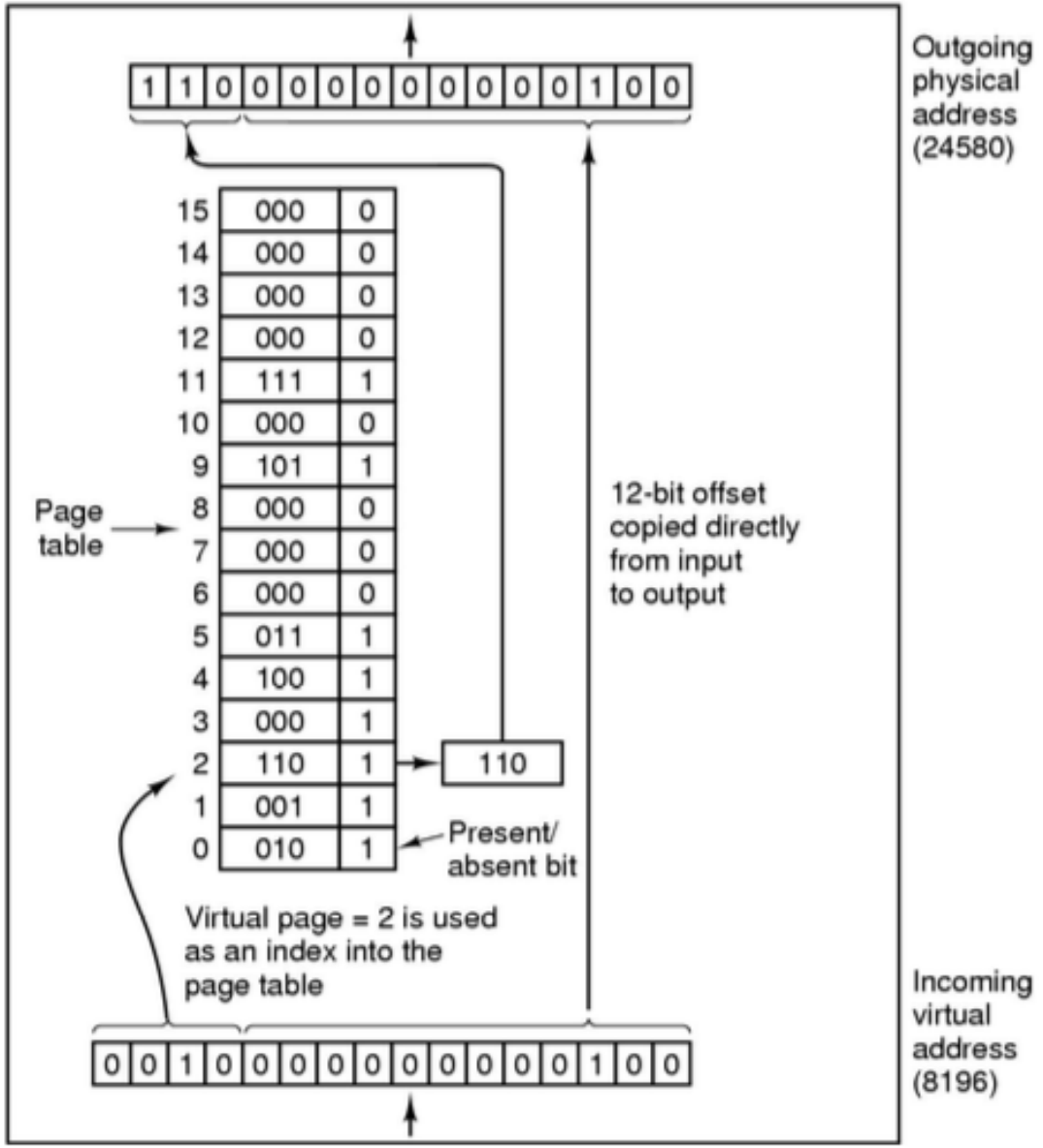
Universidade Federal de Mato Grosso
Campus Rondonópolis

Paginação – Funcionamento da Tabela de Páginas na MMU

- E como buscar um endereço?
  - Parte em vermelho:
    - Endereço base da página (montado zerando-se a parte em preto)
    - Varia, na hora de mapear páginas a suas respectivas molduras
  - Parte em preto:
    - Deslocamento (offset)
    - Não varia → um endereço que estava n bytes acima da base da página estará os mesmos n bytes acima da base da moldura

Paginação

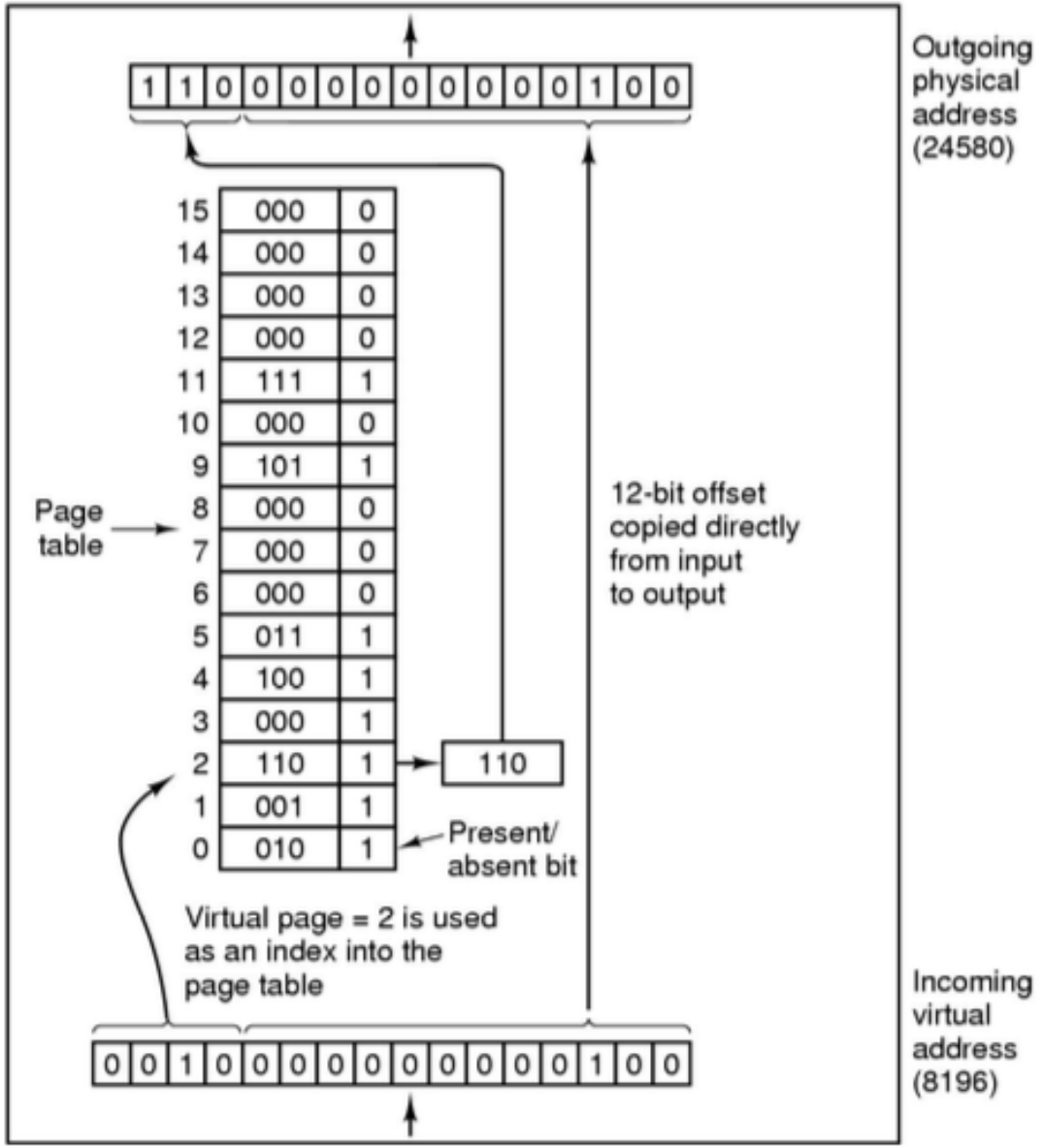
- Endereço 8196
- MMU com 16 páginas de 4KB
  - Endereço virtual de 16bits
  - Tabela tem 16 entradas (0000 a 1111)
  - Hardware com 8 frames
    - Endereço físico de 15 bits



Universidade Federal de Mato Grosso
Campus Rondonópolis

Paginação

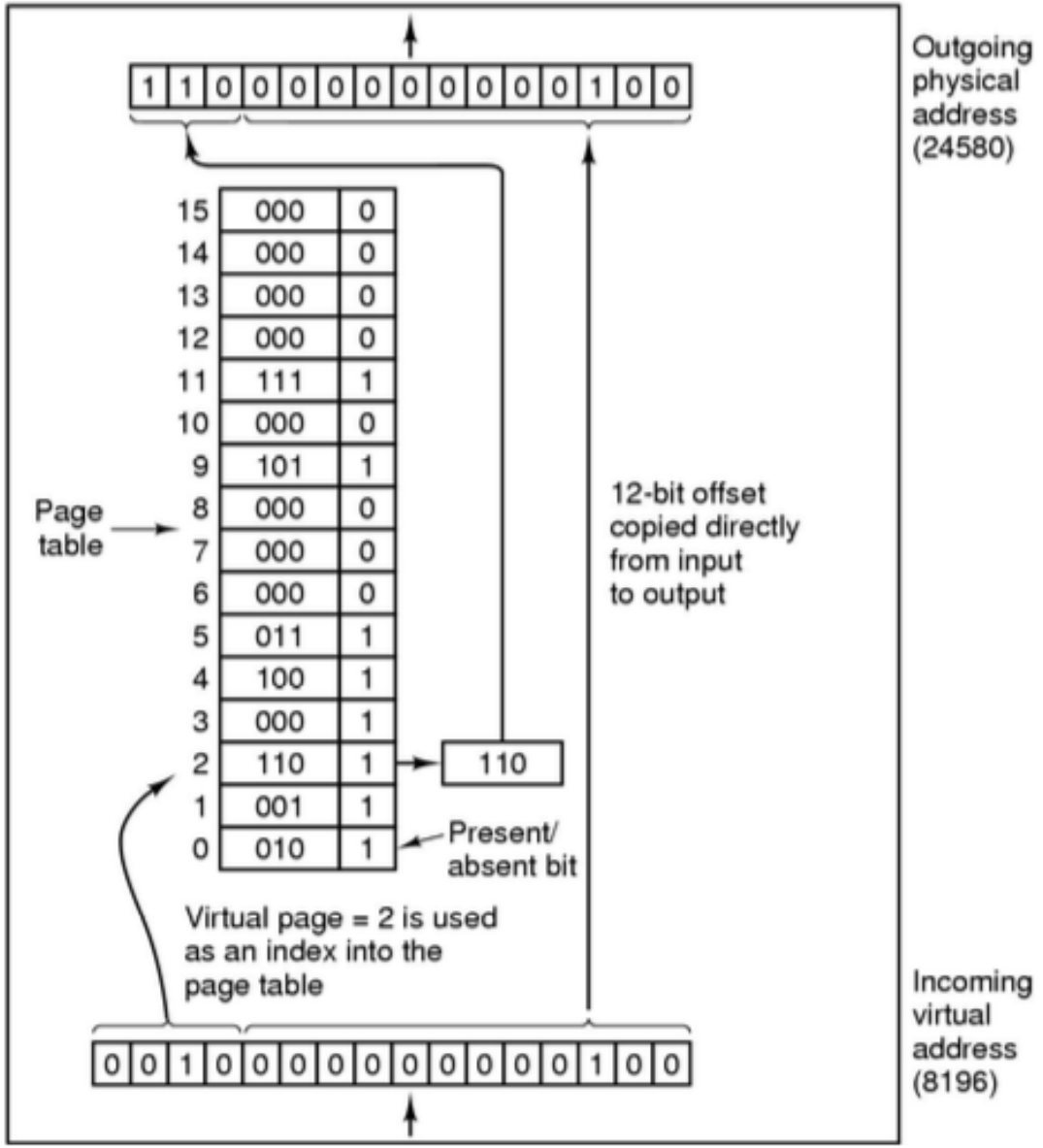
- Montando o endereço físico
  - O nº da página física (110) é copiado para os três bits mais significativos do endereço de saída (real), juntamente com o deslocamento (sem alteração)



Universidade Federal de Mato Grosso
Campus Rondonópolis

Paginação

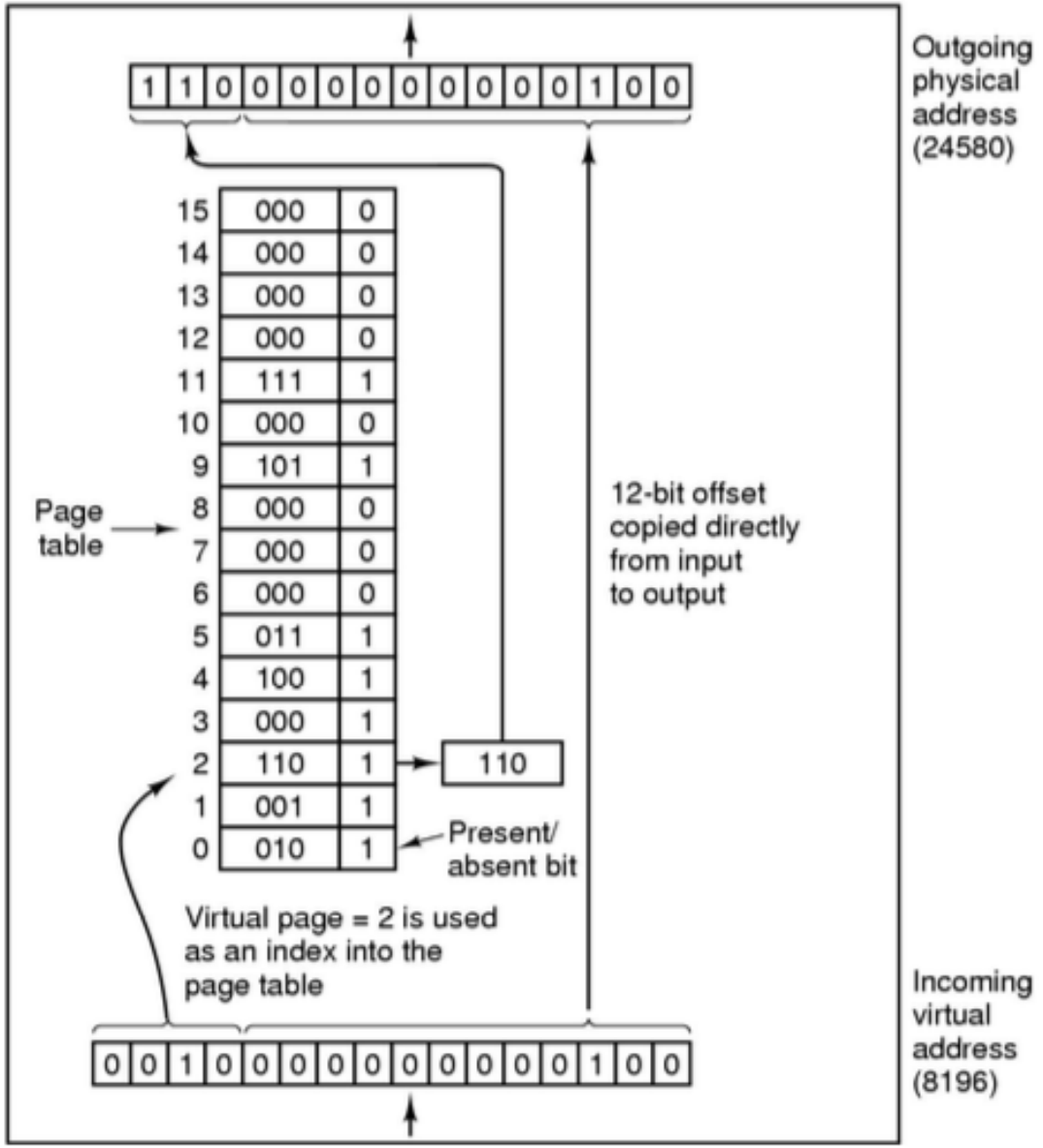
- Montando o endereço físico
  - O registrador de saída envia então esse endereço à memória, via barramento



Universidade Federal de Mato Grosso
Campus Rondonópolis

Paginação

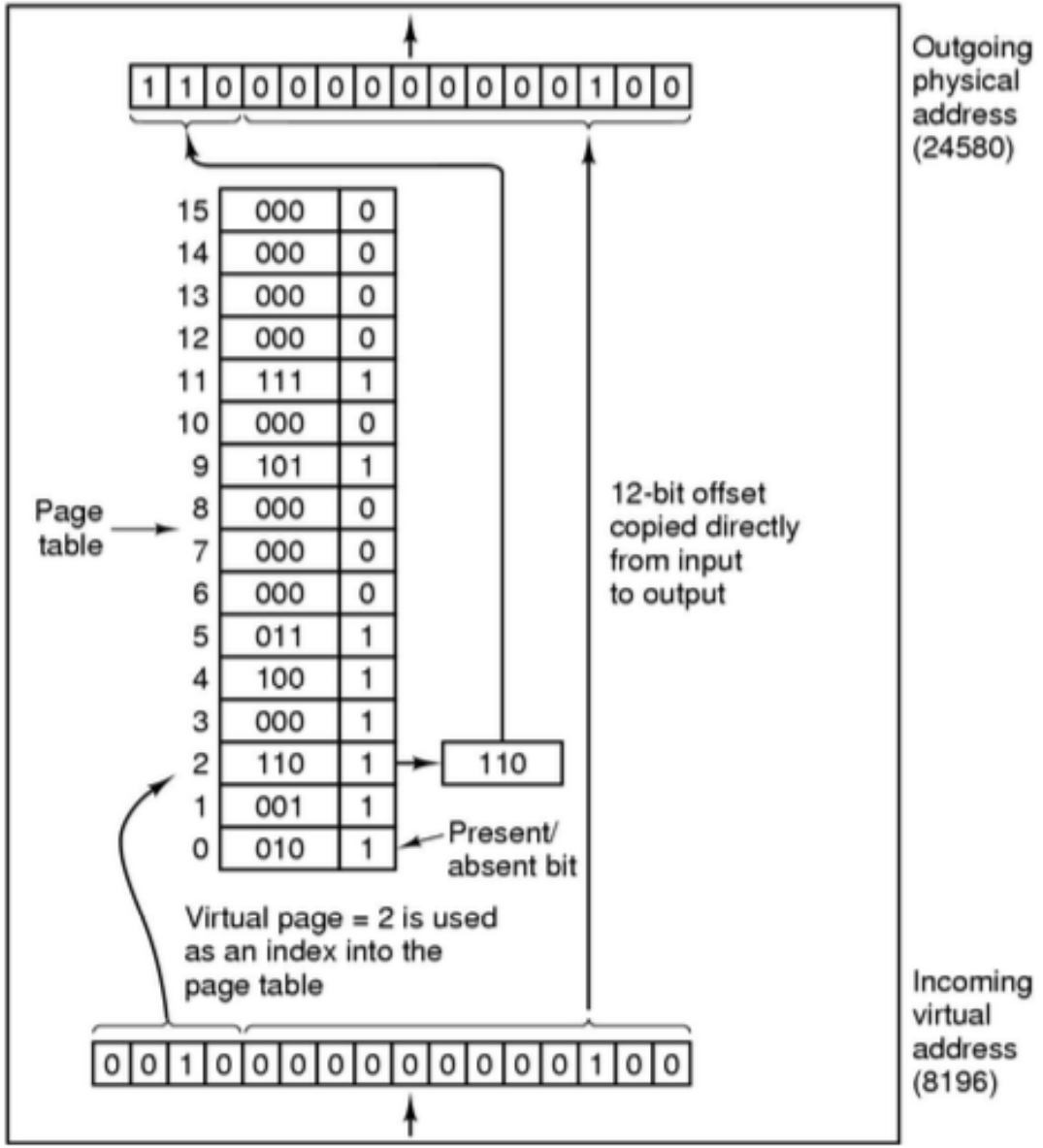
- Usamos os 4 bits mais altos do endereço virtual como índice na tabela
  - Se página não estiver na RAM
  - (Bit presente/ausente = 0)
  - Executa uma trap → Desvia ao S.O



Universidade Federal de Mato Grosso
Campus Rondonópolis

Paginação

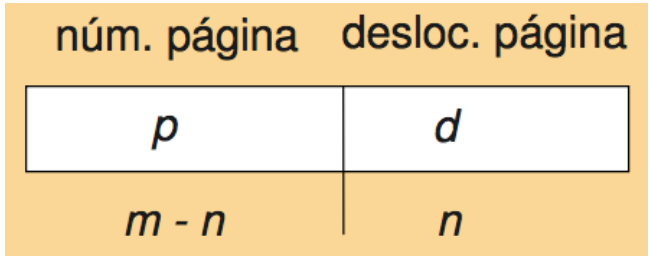
- Deslocamento
  - O quanto acima da base da página está
  - Ex: Endereço 8296
    - 1000000011101000
- 8296 = 8192 + 104:
- 010000000000000 + 000000001101000 = 010000001101000



Universidade Federal de Mato Grosso
Campus Rondonópolis

Esquema de tradução de Endereço

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



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

Universidade Federal de Mato Grosso
Campus Rondonópolis

Paginação

- Espaço de endereço lógico de um processo pode ser não contíguo
  - Processo recebe memória física sempre que houver memória disponível
- Implementação da paginação
  - Divida a memória física em blocos de tamanho fixo, denominados frames
  - Divida a memória lógica em blocos do mesmo tamanho, denominados páginas
  - Acompanhe todos os frames livres
  - Configure uma tabela de página para traduzir endereços lógicos para físicos

Paginação

- Problemas
  - Fragmentação interna;
  - Definição do tamanho das páginas;
    - Geralmente a MMU que define e não o SO;
    - Páginas maiores: leitura mais eficiente, tabela menor, mas maior fragmentação interna;
    - Páginas menores: leitura menos eficiente, tabela maior, mas menor fragmentação interna;
  - Problema
    - Geralmente queremos páginas enormes com acesso rápido

Implementação da Tabela de Páginas

- Entrada na Tabela de Páginas:
  - Depende muito do hardware
  - Em geral, 32 bits, divididos da seguinte maneira:
    - Page frame number:
      - Identifica a página real
      - Campo mais importante
    - Bit de Presente/ausente
    - Bit de proteção (tipo de acesso permitido - 1 ou 3 bits)
    - Bit de modificação (controla uso da página)
    - Bit de referência (marca se a página foi referenciada)
    - Bit de cache (permite desabilitar caching para a página)

Implementação da Tabela de Páginas

- A tabela de páginas pode ser armazenada de três diferentes maneiras:
  - Array de Registradores, se a memória for pequena;
    - Mantidos no hardware
  - Na própria memória RAM
    - A MMU gerencia utilizando um ou dois registradores
  - Em uma memória cache na MMU chamada Memória Associativa (Translation Lookaside Buffer – TLB)
    - Usada para melhorar o desempenho da tabela na RAM

Implementação da tabela de páginas

- Tabela de página mantida na memória principal
  - Registrador de base da tabela de página (PTBR- Page table base register)
    - Aponta para o início da tabela de página, indicando o endereço físico de memória onde a tabela está alocada
  - Registrador de tamanho da tabela de página (PTLR - Pagetable length register)
    - Existente apenas em alguns sistemas
    - Indica tamanho da tabela de página (número de entradas da tabela → número de páginas);

Tabela de páginas na RAM

- Nesse esquema, cada acesso de dado/instrução exige dois acessos à memória:
  - **Um** para a tabela de página + **um** para o dado/instrução
  - Cada acesso à memória, feito no programa, se transforma em 2
  - Contudo, muitos programas tendem a fazer um grande número de referências a um pequeno número de páginas
    - Apenas uma fração pequena das entradas na tabela são lidas com grande freqüência
  - O que fazer?

Tabela de páginas na RAM

- Solução
  - O problema dos dois acessos à memória pode ser solucionado pelo uso de um cache de hardware especial para pesquisa rápida
  - Chamado memória associativa ou Translation Lookaside Buffer (TLB)
  - Hardware especial para mapear endereços virtuais para endereços reais sem ter que passar pela tabela de páginas na memória principal

Estrutura de um TLB

- Válido/inválido
- Página virtual
- Bit Modificado
- Bit de proteção
- Frame da página

Valid	Virtual page	Modified	Protection	Page frame
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

Proteção de memória

- Implementada associando-se bits de proteção e de válido/inválido a cada moldura
- Bit de válido-inválido anexado a cada entrada na tabela de página:
  - “válido” indica que a entrada na TLB possui dados válidos
  - “inválido” indica que a entrada ainda não foi usada
    - Necessário porque a TLB tem tamanho fixo, então há que se saber o que está em uso e o que não está

Memória associativa (TLB)

- Com exceção do número da página virtual (virtual page), todos os demais também estão na tabela de páginas

Valid	Virtual page	Modified	Protection	Page frame
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

Memória associativa (TLB)

- Funcionamento
  - Quando um endereço virtual chega à MMU, o hardware verifica se sua página virtual está na TLB
  - Compara a todas as entradas simultaneamente (operação feita no hardware)

Valid	Virtual page	Modified	Protection	Page frame
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

Memória associativa (TLB)

- Funcionamento
  - Se estiver na TLB (hit), e os bits de proteção não forem violados, a moldura é obtida da TLB, sem passar pela tabela de páginas
  - Se violar algum dos bits de proteção é gerada uma falha de proteção (protection fault)
    - Desvio ao SO via TRAP

Valid	Virtual page	Modified	Protection	Page frame
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

Memória associativa (TLB)

- Funcionamento
  - Se a página virtual não estiver na TLB
  - A MMU busca-a na tabela de páginas
  - Remove uma das entradas da TLB, devolvendoa à tabela na memória
  - Coloca nessa entrada a página que acabou de buscar
  - Se essa página for usada novamente, estará na TLB

Valid	Virtual page	Modified	Protection	Page frame
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

Memória associativa (TLB)

- Pode ser implementada em:
  - Hardware
    - Maior rapidez
    - Ocupa espaço físico que poderia ser disponibilizado para outras funções (cache etc)
  - Software
    - Usa o software SO (mais lento); gera traps
    - As páginas contendo a tabela de páginas podem não estar na TLB, quando do processamento de uma page fault
      - Causará novas page faults durante o processo

Memória associativa

- Dois tipos de falhas em encontrar páginas:
- Soft miss
  - Quando a página referenciada não está na TLB, mas está na memória física
  - Basta atualizar a TLB
- Hard miss
  - A página em si não está na memória física (e nem na TLB, naturalmente)
  - Deve-se fazer um acesso ao disco para trazê-la à memória (e então à TLB)
  - Muito lento
- Page fault pequeno/simples vs. Page fault maior/complexo
- Falha de segmentação

Tabelas de páginas

- TLBs aceleram a tradução entre endereços virtuais e reais
- Mas esse é apenas um problema → como traduzir rapidamente
- Resta ainda como lidar com grandes espaços de endereços virtuais → Como organizar a tabela de páginas?
  - Paginação hierárquica (multinível)
  - Tabelas de página com hash
  - Tabelas de página invertidas

Paginação multinível

- Quebre o espaço de endereço lógico em múltiplas tabelas de página
  - Uma técnica simples é uma tabela de página em dois níveis
- A idéia é evitar manter na memória todas as tabelas de página o tempo todo
  - Apenas as necessárias devem estar na memória
  - Ex: Podemos dividir o programa em:
    - Uma tabela para o código (4MB da base da memória)
    - Uma para os dados (4MB seguintes)
    - Uma para a pilha (4MB do topo)
    - Há um buraco gigante vazio, na tabela (usou apenas 3 páginas)
- Mesmo que a tabela inteira seja enorme, para esse programa precisaríamos apenas manter 4 (a principal, e essas outras 3)
- As demais (possivelmente não usadas), não estariam na memória.

Universidade Federal de Mato Grosso
Campus Rondonópolis

Tabela de página em hash

- Comuns em espaços de endereço > 32 bits
- O número de página virtual é usado para cálculo do valor de hash
- Cada entrada na tabela contém uma lista ligada de elementos, consistindo de:
  - O número da página virtual
  - O valor da frame
  - Um ponteiro para o próximo elemento da lista

Universidade Federal de Mato Grosso
Campus Rondonópolis

Tabela de página invertida

- Quanto mais bits, pior para multi-nível
  - Solução : tabela de página invertida
- Possui uma entrada por frame na memória física
  - Em vez de uma entrada por página no espaço virtual
  - A entrada registra que página virtual esta localizada na moldura
    - Com informações sobre o processo que possui essa página
- Poupam muito espaço quando o espaço virtual é muito maior que o físico

Universidade Federal de Mato Grosso
Campus Rondonópolis

Alocação de páginas

- Quantas páginas reais serão alocadas a um processo?
- Duas estratégias
  - **1) Alocação fixa ou estática:** cada processo tem um número máximo de páginas reais, definido quando o processo é criado;
  - O limite pode ser igual para todos os processos;
  - Vantagem: simplicidade;
  - Desvantagens: (i) número muito pequeno de páginas reais pode causar muita paginação; (ii) número muito grande de páginas reais causa desperdício de memória principal;

Alocação de páginas

- Quantas páginas reais serão alocadas a um processo?
- Duas estratégias
  - **2) Alocação variável ou dinâmica:** número máximo de páginas reais alocadas ao processo varia durante sua execução;
  - Vantagem: (i) processos com elevada taxa de paginação podem ter seu limite de páginas reais ampliado; (ii) processos com baixa taxa de paginação podem ter seu limite de páginas reais reduzido;
  - Desvantagem: monitoramento constante;

Busca de página

- Paginação simples
  - Todas as páginas virtuais do processo são carregadas para a memória principal;
  - Assim, sempre todas as páginas são válidas;
- Paginação por demanda (Demand Paging):
  - Processos começam com nenhuma página na memória
  - Assim que a CPU tenta executar a primeira instrução, gera um page fault
  - O S.O. Traz a página que falta à memória

Busca de página

- Paginação por demanda (Demand Paging):
  - Apenas as páginas efetivamente acessadas pelo processo são carregadas na memória principal;
  - Quais páginas virtuais foram carregadas?
    - Indicado pelo bit de residência
  - Página inválida:
    - MMU gera uma interrupção de proteção e aciona o sistema operacional
    - Se a página está fora do espaço de endereçamento do processo, o processo é abortado;
    - Se a página ainda não foi carregada na memória principal, ocorre uma falta de página (page fault);

Busca de página

- Paginação por demanda – falta de página
  - Processo é suspenso e seu descritor é inserido em uma fila especial
  - Fila dos processos esperando uma página virtual;
  - Uma página real livre deve ser alocada;
  - A página virtual acessada deve ser localizada no disco;
  - Operação de leitura de disco
    - Indicando o endereço da página virtual no disco e o endereço da página real alocada

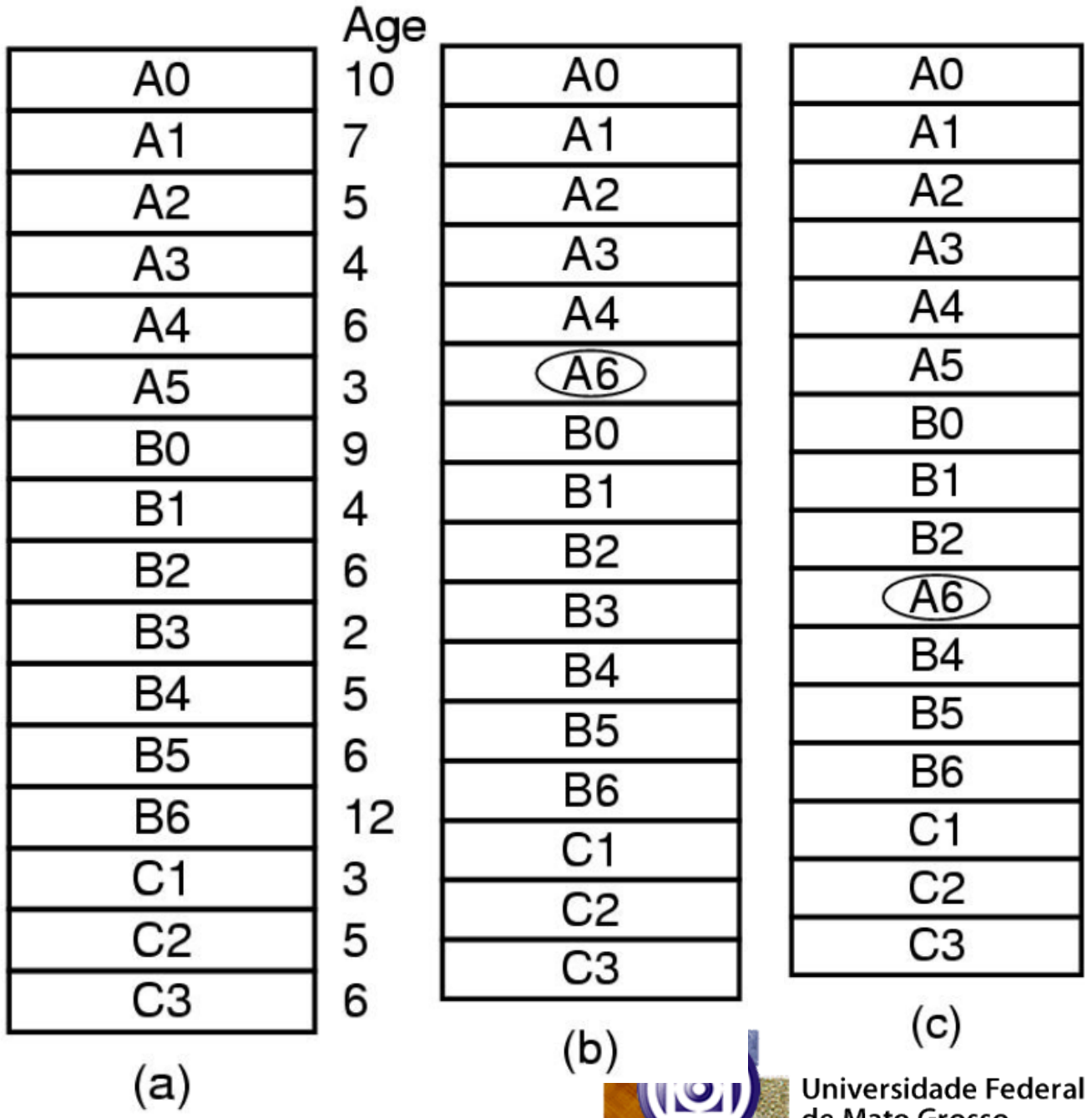
Universidade Federal de Mato Grosso
Campus Rondonópolis

Busca de página

- Paginação por demanda – Falta de Página:
  - Após a leitura do disco
    - Tabela de páginas do processo é corrigida para indicar que a página virtual agora está válida e está na página real alocada;
    - Processo executado pelo paginador:
      - Carrega páginas especificas de um processo do disco para a memória principal;
    - O descritor do processo é retirado da fila especial e colocado na fila do processador;

Troca de páginas

- Política de Substituição Local/Global
  - Local: Páginas dos próprios processos são utilizadas na troca;
    - Dificuldade: definir quantas páginas cada processo pode utilizar
  - Global: páginas de todos os processos utilizados na troca
    - Processos sem fração fixa na memória
    - Problema: falta de página para processo com menor prioridade



Universidade Federal de Mato Grosso
Campus Rondonópolis

Troca de páginas

- Quando ocore uma page fault...
  - Se todas as frames estiverem ocupadas, o S.O. Escolhe uma página para eliminar – página vítima
    - A escolha pode ser aleatória
    - Será melhor escolher uma página que não seja muito usada
  - Se essa página foi modificada enquanto estava na memória, deve ser reescrita no disco
  - Se não tiver sido modificada, pode apenas ser sobrescrita

Algoritmos de troca de páginas

- Ótimo
- NRU (Not Recently Used)
- FIFO (First In, First Out)
- Segunda Chance
- Relógio (Clock)
- LRU (Least Recently Used)
- Working Set (WS)
- WSClock

Universidade Federal
de Mato Grosso
Campus Rondonópolis

Algoritmo ótimo

- Cada página é marcada com o número de instruções que serão executadas antes que a página seja referenciada
  - Retira da memória a página que tem menos chance de ser referenciada (maior número de instruções faltantes)
- Praticamente impossível de se saber;
- Impraticável;
- Usado em simulações para comparação com outros algoritmos;

Algoritmo Not Recently Used Page Replacement (NRU)

- Para auxiliar o S.O. a coletar estatísticas de página de uso:
  - 02 bits associados a cada página → R(eferenciada) e M(odificada)
    - Classe 0 (00) → não referenciada, não modificada;
    - Classe 1 (01) → não referenciada, modificada;
    - Classe 2 (10) → referenciada, não modificada;
    - Classe 3 (11) → referenciada, modificada
- Referenciada → lida ou escrita
- Modificada → escrita

Algoritmo Not Recently Used Page Replacement (NRU)

- Para auxiliar o S.O. a coletar estatísticas de página de uso:
  - R e M são atualizados a cada referência à memória;
  - Armazenados em cada entrada da tabela de página
  - Seu valor é determinado pelo hardware
- Quando um processo é iniciado, ambos R e M são 0 para todas suas páginas
  - Periodicamente, o bit R é limpo para diferenciar as páginas que não foram referenciadas recentemente;
    - A cada interrupção de relógio;

Algoritmo Not Recently Used Page Replacement (NRU)

- O bit M não é limpo, pois o S.O. precisa saber se deve escrever a página no disco
- Quando ocorre uma page fault:
  - Remove uma página aleatoriamente, escolhendo dentre
    - as classes mais inferiores → bits 00, 01, 10, 11
- Vantagens:
  - Fácil de entender, eficiente para implementar e fornece bom desempenho;

Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo First in, First out Page Replacement (FIFO)

- SO mantém uma fila das páginas correntes na memória;
  - A página no início da fila é a mais antiga e a página no final é a mais nova;
  - Quando ocorre um page fault
    - A página do início é removida
    - A nova é inserida ao final da fila
- Simples, mas pode ser ineficiente, pois uma página que está em uso constante pode ser retirada;
- Pouco utilizado;

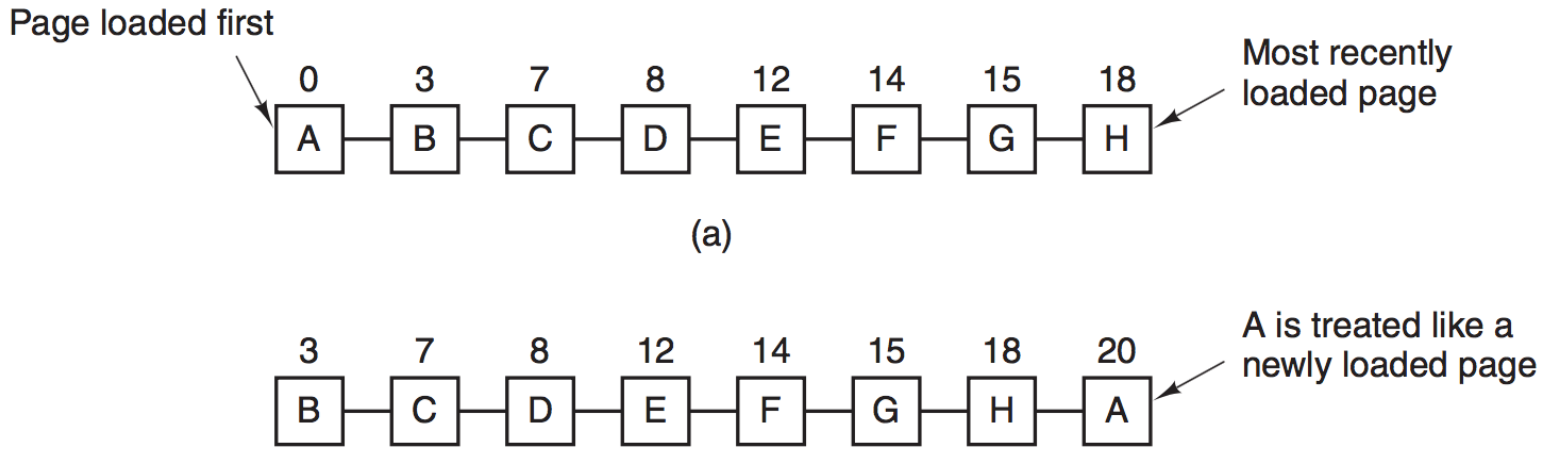
Algoritmo da Segunda Chance

- FIFO + bit R;
- Inspeciona o bit R da página mais velha
  - Se for 0, ela é velha e não foi usada recentemente → é trocada
  - Se for 1, o bit é feito 0
    - A página é colocada no final da fila
    - Seu tempo de carga é modificado, fazendo parecer que recém chegou na memória (recebe uma segunda chance)
    - A busca continua
- Busca por uma página antiga que não foi referenciada no clock mais recente

Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo da Segunda Chance

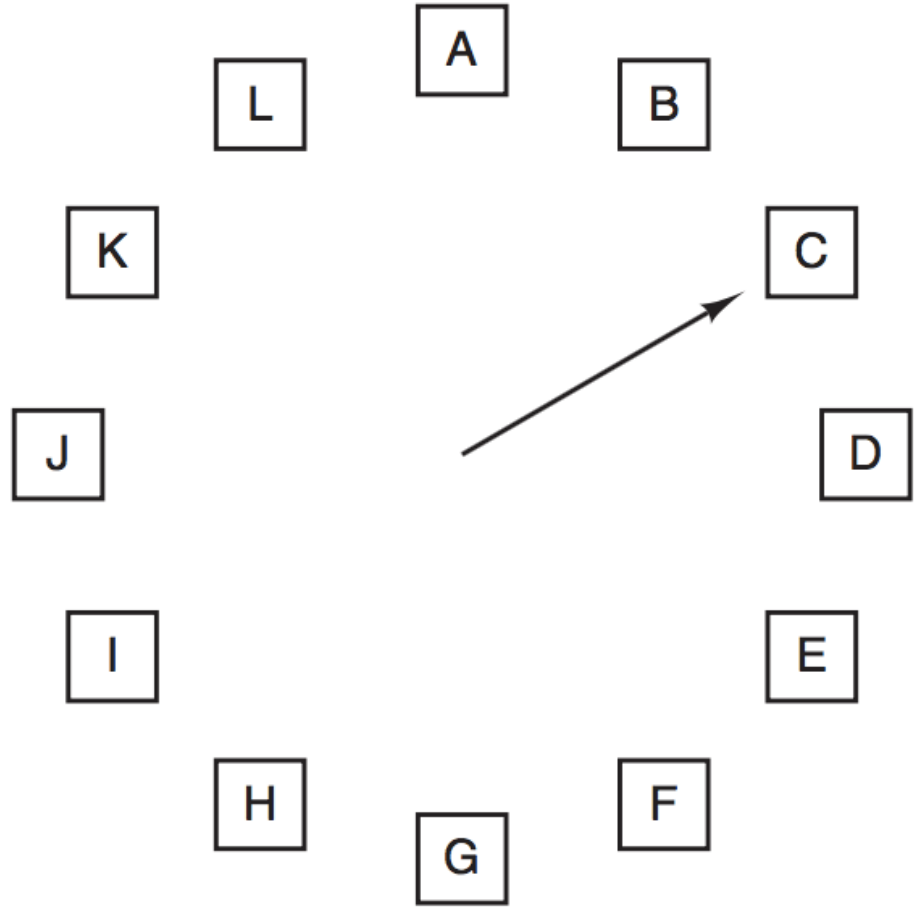
- Exemplo
  - Ocorre uma page fault no tempo 20
  - Se Ra = 0
    - R é abandonado ou levado para o disco
  - Se Ra = 1,
    - Vai para o final da lista e seu tempo de carregamento é setado para o atual (20)
    - Ra = 0



Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo do Relógio

- Algoritmo do Relógio
- Melhoria ao Segunda Chance:
  - Lista circular com ponteiro apontando para a página mais antiga, na forma de um relógio
  - A cabeça aponta para a página mais antiga
- Quando ocorre uma page fault...
  - Inspeciona-se a cabeça
  - Se R = 1, limpa e procura pelo próximo
  - Se R = 0
    - Susbstitui-se a página pela nova página
    - Avança-se uma posição



Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo Least Recently Used Page Replacement (LRU)

- Ideia
  - Páginas que foram muito usadas nas últimas instruções serão provavelmente usadas novamente nas próximas
  - Troca a página que permaneceu em desuso pelo maior tempo
- Alto Custo
  - Deve-se manter lista encadeada com todas as páginas que estão na memória, com as mais recentemente utilizadas no início e as menos utilizadas no final;
  - A lista deve ser atualizada a cada referência da memória

Algoritmo Least Recently Used Page Replacement (LRU)

- Pode ser implementado tanto por hardware quanto por software:
- Hardware: MMU deve suportar a implementação LRU;
  - Contador em hardware (64 bits), incrementado automaticamente após cada instrução
  - Tabela de páginas armazena o valor desse contador (C) em cada entrada
  - Após cada referência à memória, o valor atual de C é armazenado na entrada correspondente (página) na tabela
  - Em um *page fault*, o S.O. examina todas as entradas na tabela, para encontrar a com menor C

Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo Least Recently Used Page Replacement (LRU)

- Pode ser implementado tanto por hardware quanto por software:
- Software (1/2) – NFU (Not Frequently Used):
  - Para cada página existe um contador, iniciado com zero
  - A cada interrupção do clock, o SO varre todas as páginas da memória
  - Para cada página, adiciona o bit R (residência) ao contador Em um page fault, escolhe a página com o menor contador
  - **Problema**: Como esse algoritmo não se esquece de nada, páginas frequentemente acessadas em uma porção pequena do código, mas que não mais serão acessadas, não serão candidatas

Universidade Federal de Mato Grosso
Campus Rondonópolis

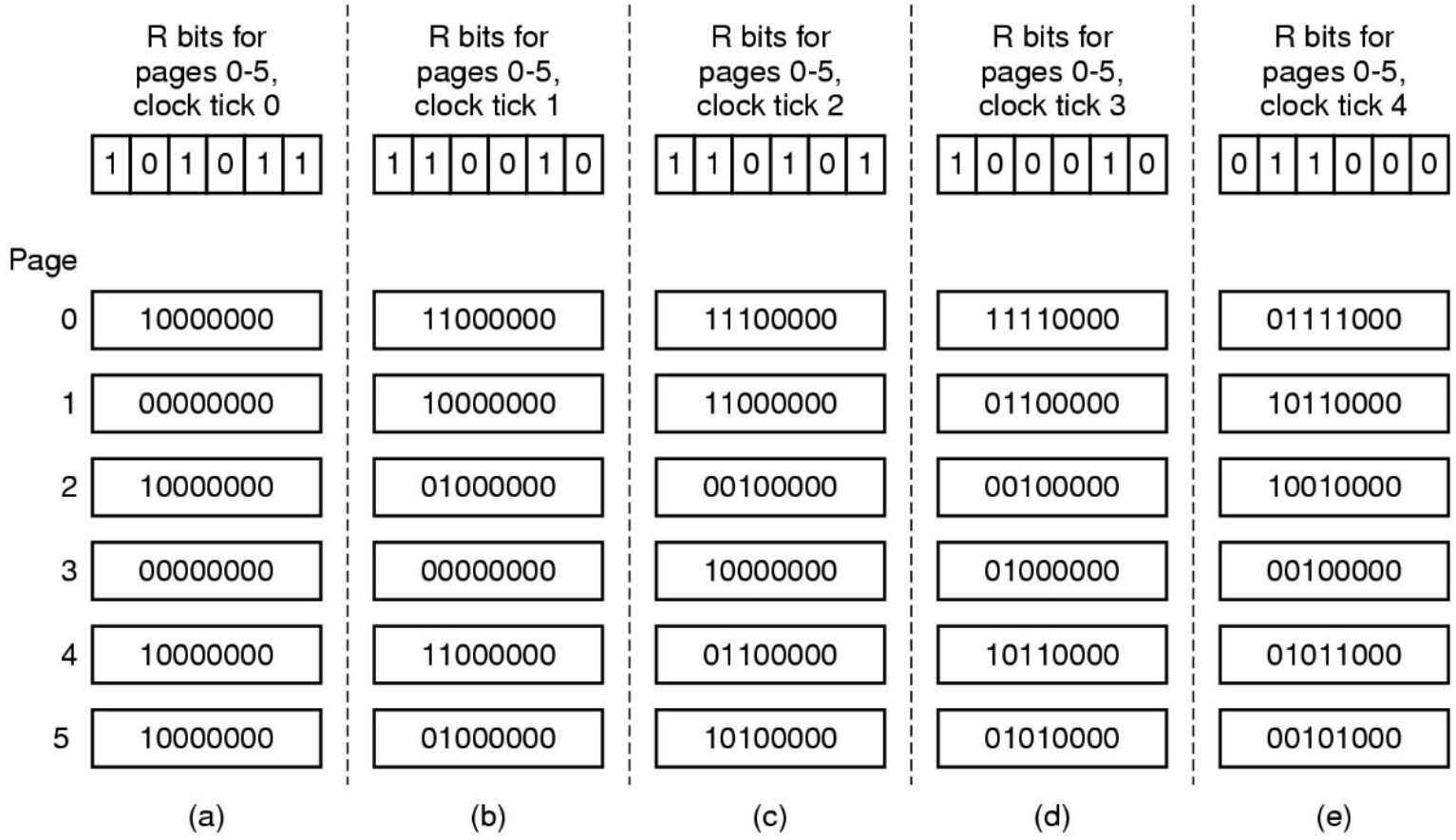
Algoritmo Least Recently Used Page Replacement (LRU)

- Pode ser implementado tanto por hardware quanto por software:
- Software (2/2) – Aging:
  - Solução ao problema do algoritmo NFU
  - Além de saber quantas vezes a página foi referenciada, também controla quando ela foi referenciada
  - Primeiro, os contadores são deslocados à direita em um bit. Só então o bit R é adicionado, só que ao bit mais da esquerda
    - Também a cada interrupção do clock
  - Em um page fault, a página com o menor contador é removida

Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo Least Recently Used Page Replacement (LRU)

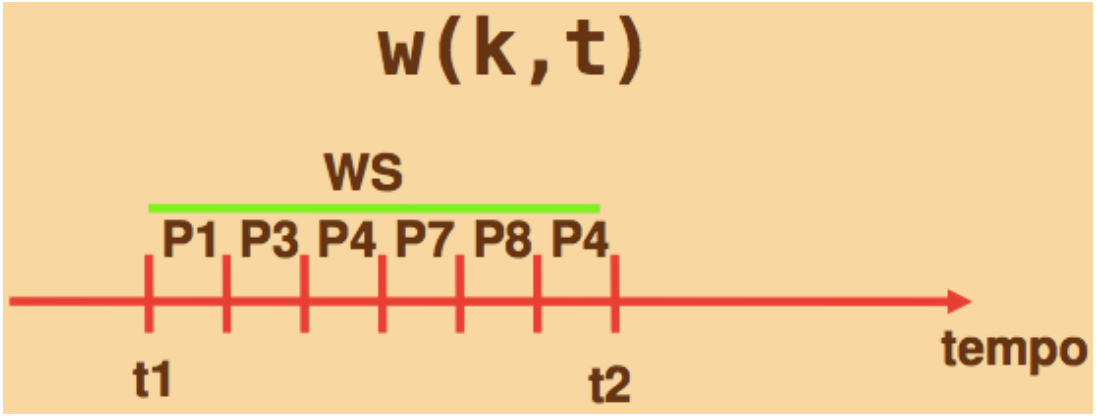
- Após 8 clocks uma página não referenciada tem seu contador zerado. Quanto mais tempo ficar sem ser referenciada, mais zeros a esquerda terá e menor será seu contador



Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo Working Set (WS):

- Paginação por demanda → páginas são carregadas na memória somente quando são necessárias;
- Prépaginação → Working set
  - Conjunto de páginas que um processo está efetivamente utilizando (referenciando) em um determinado tempo t;



Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo Working Set (WS):

- Objetivo principal: reduzir a falta de páginas
  - Um processo só é executado quando todas as páginas necessárias no tempo t estão carregadas na memória;
    - Até então, gerará page faults
  - A idéia é determinar o working set de cada processo e certificar-se de tê-lo na memória antes de rodar o processo – Modelo de Working set ou pré-paginação
- Working set w(k,t)
  - Conjunto consistindo, em um dado instante t, de todas as páginas usadas pelas k referências mais recentes à memória

Algoritmo Working Set (WS):

- O working set varia lentamente com o tempo
  - Podemos estimar o número de páginas necessárias quando o programa é trazido do disco com base em seu working set de quando foi interrompido.
  - Pré-paginação consiste em carregar essas páginas antes de rodar novamente o processo
- Implementação
  - O SO precisa manter registro de que páginas estão no working set.
  - Quando ocorrer um page fault, encontre uma página fora do working set e a remova, caso não haja mais nenhum frame livre

Algoritmo Working Set (WS):

- Implementação:
  - Contar as k referências mais recentes é custoso
  - Para simplificar → o working set pode ser visto como o conjunto de páginas que o processo referenciou durante os últimos t segundos de sua execução
    - Conta o tempo individual do processo, descontando escalonamento → seu tempo virtual corrente
  - Utiliza bit R e o tempo de relógio (tempo virtual) da última vez que a página foi referenciada;

Algoritmo Working Set (WS):

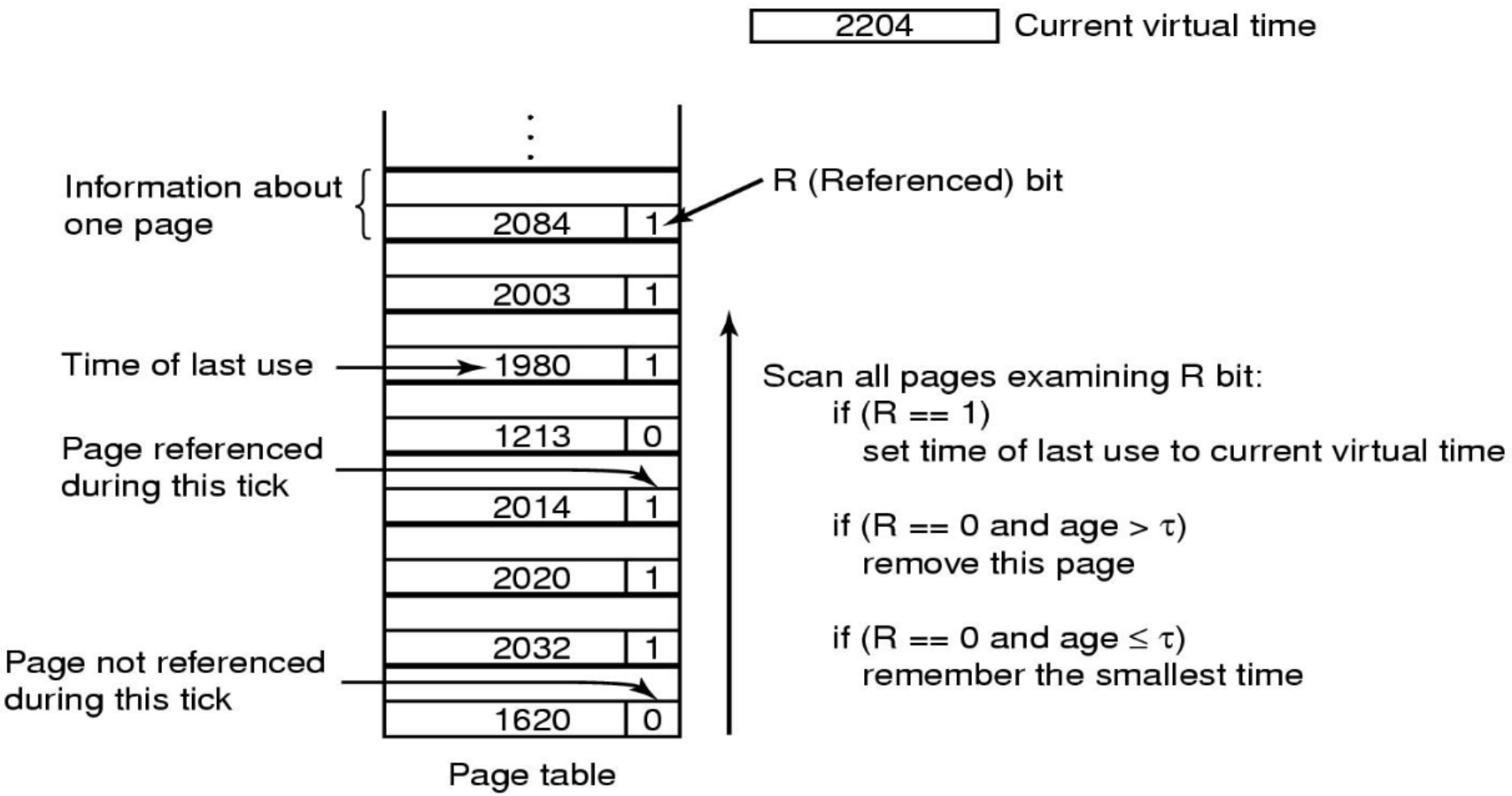
- Algoritmo:
  - Pressupostos:
    - O hardware define os bits R e M
    - Em cada ciclo do clock, o bit de referência é limpo
    - O tempo do working set se estende por vários ciclos do clock
  - Em cada page fault, a tabela de páginas inteira é buscada
    - À medida que cada entrada é processada, examine R
    - Se 1, escreva o tempo virtual corrente no campo Tempo do Último Uso (TLU), indicando que a página estava em uso no instante da page fault, ou seja, estava no working set → não é candidata

Algoritmo Working Set (WS)

- Algoritmo
  - Em cada page fault, a tabela de páginas inteira é buscada
    - À medida que cada entrada é processada, examine R
    - Se R=0, a página não foi referenciada no ciclo atual, e pode ser uma candidata
      - Nesse caso, se sua idade for maior que o intervalo t do working set, ela não está nele, e pode ser removida
        - A busca continua atualizando as demais entradas
    - Se, contudo, a idade for menor que t, a página é poupada. Contudo, a página com maior idade é marcada
    - Se nenhum candidato for encontrado (todas as páginas estão no working set), substitua a página mais velha, dentre as com R=0

Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo Working Set (WS)



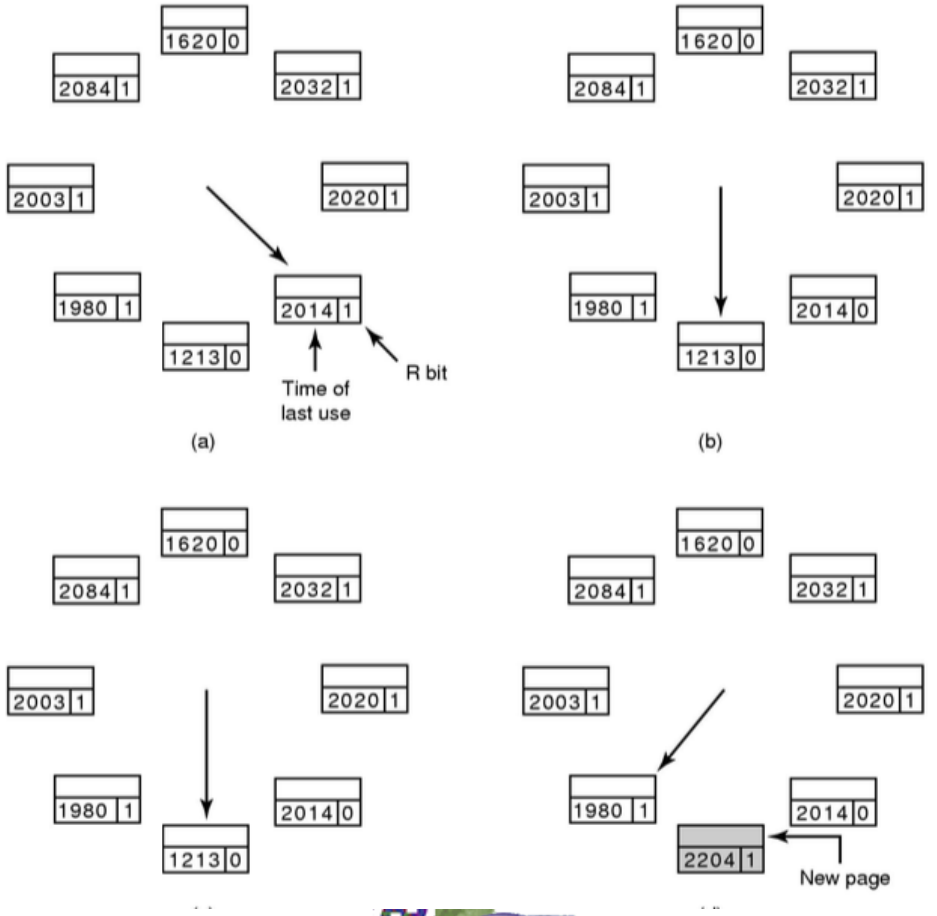
Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo WSClock

- Clock + Working Set
- Tenta evitar a verificação da tabela inteira
- Amplamente usado, devido à sua simplicidade e performance
- Utiliza lista circular de páginas
  - Inicialmente vazia
  - À medida que mais páginas são carregadas, entram na lista, formando um anel
  - Cada entrada contém o tempo de último uso, além dos bits R e M

Algoritmo WSClock

- Funcionamento
  - A cada page fault, a página da cabeça é examinada primeiro
- Se R=1
  - A página foi usada durante o ciclo de clock corrente → não é candidata a remoção
  - Faz R = 0 e avança a cabeça à próxima página, repetindo o algoritmo para esta página
- Se R=0
  - Se a idade for maior que o tamanho do working set t e a página estiver limpa (M=0) → não está no working set e uma cópia válida existe no disco
  - A página é substituída
  - A cabeça da lista avança



Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmo WSClock

- Funcionamento
- Se R =0
  - Se, contudo, a página estiver suja → não possui cópia válida no disco
    - Agenda uma escrita ao disco, evitando troca de processo
    - Avança a cabeça da lista, prosseguindo da página seguinte

Algoritmo WSClock

- Se a cabeça der uma volta completa na lista sem substituir:
  - E pelo menos uma escrita no disco foi agendada
  - A cabeça continua se movendo, em busca de uma página limpa
  - Em algum momento a escrita agendada será executada, marcando a página como limpa
- E nenhuma escrita foi agendada
  - Todas as páginas estão no working set
  - Na falta de informação adicional, substitua qualquer página limpa
  - Se nenhuma página limpa existir, escolha qualquer outra e a escreva no disco

Universidade Federal de Mato Grosso
Campus Rondonópolis

Algoritmos de troca de página

- Algoritmos de substituição local:
  - Working Set;
  - WSClock;
    - O conceito de working set se aplica somente a um único processo → não há working set para a máquina como um todo
- Algoritmos de substituição local/global:
  - Ótimo;
  - NRU;
  - FIFO;
  - Segunda Chance;
  - LRU;
  - Relógio

Questões de implementação

- Envolvimento do SO com a paginação, durante...
  - ...Criação de processos, execução, page fault, e terminação
- Sequência para lidar com page faults
- Backup das instruções, durante o trap do page fault
  - Referência da memória pode mudar
- Travamento da página na memória
  - No caso de I/O, por exemplo
- Aonde colocar a página quando retiradas da memória?
- Separação de política e mecanismo

Universidade Federal
de Mato Grosso
Campus Rondonópolis

Segmentação

- Até então a memória virtual discutida é unidimensional
- Para muitos problemas, ter mais de um espaço de endereçamento virtual é interessante
  - Exemplo: compilador e suas diferentes tabelas
- Uma forma de lidar com isso é fornecer espaços de endereçamentos independentes, chamados segmentos
  - Esses endereçamentos podem crescer e diminuir de acordo
- Um segmento é uma entidade lógica
  - Programação tem consciência do uso
- Segmentar facilita a troca de informações entre processos

Universidade Federal de Mato Grosso
Campus Rondonópolis

Segmentação

- Implementação
  - Na paginação, páginas tem tamanho fixo; segmentos não
- Ultimamente...
  - Pouco utilizada

Implementação da paginação

- Onde colocar as páginas no disco, quando retiradas da memória?
  - A solução mais simples é ter uma partição especial de swap
    - Solução do Unix e Linux
    - Não possui um sistema de arquivos normal
      - Quando o sistema inicia, a partição está vazia
        - Representada na memória como uma única entrada contendo sua origem e tamanho
      - À medida em que processos são iniciados, o SO reserva um pedaço da área de swap do tamanho do processo
        - Quando terminam, o espaço é liberado
      - A área de troca é gerenciada como uma lista de espaços disponíveis;

Universidade Federal de Mato Grosso
Campus Rondonópolis

Implementação da paginação

- Há algoritmos melhores, mas que não serão discutidos
- Associado a cada processo está o endereço no disco de sua área de swap
  - Mantido na tabela de processos
  - Cálculo do endereço para escrever uma página:
    - Adicione o endereço do início da página (seu valor no endereço virtual) ao início da área de swap associada ao processo

Implementação da paginação

- Onde colocar as páginas no disco, quando retiradas da memória?
  - Problema: antes de um processo iniciar, a área de swap deve ser inicializada
    - Possibilidade A – Assim que o processo é criado, ele é copiado todo para sua área de troca no disco, sendo carregado para memória quando necessário;
      - Alternativamente, podemos copiá-lo todo para a memória principal (espelhamento)
      - Problema: processos podem aumentar de tamanho após iniciarem (pilha e dados)
      - Solução: reservar áreas de troca diferentes para texto do programa, dados e pilha, permitindo que elas consistam de mais de um bloco no disco
      - Basta saber o endereço do início da área de swap do processo
        - As páginas são espelhadas no disco
      - Área de troca (swap) estática

Universidade Federal de Mato Grosso
Campus Rondonópolis

Implementação da paginação

- Problema: antes de um processo iniciar, a área de swap deve ser inicializada
  - Possibilidade B – Nada é alocado antecipadamente.
    - Espaço é alocado em disco quando a página for enviada para lá e desalocado quando volta para a memória
    - Assim, processo na memória RAM não fica “amarrado” a uma área específica;
    - Desvantagem: precisamos, na memória, de um endereço de disco para cada página.
      - Deve haver uma tabela em cada processo dizendo onde cada página está no disco (se estiver lá)
      - Antes, bastava saber onde o processo estava no disco
    - Além do endereço do início da área de swap do processo, temos que saber onde está a página dentro desse endereço (seu deslocamento)
    - Área de troca dinâmica