Aula 16 – Troca de Página e Implementação de Paginação

Norton Trevisan Roman
Clodoaldo Aparecido de Moraes Lima

7 de novembro de 2014

FIFO – First-in First-out

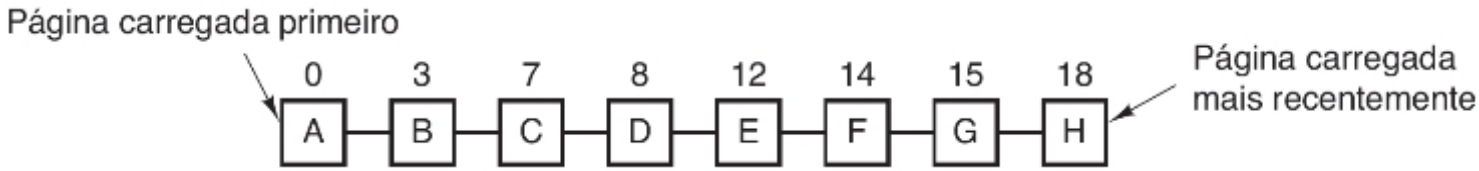
- SO mantém uma fila das páginas atualmente na memória
  - A página no início da fila é a mais antiga e a página no final é a mais nova
  - Quando ocorre um page fault
    - A página do início (mais velha) é removida
    - A nova é inserida ao final da fila
- Simples, mas pode ser ineficiente, pois uma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

Segunda Chance

- FIFO + bit R
  - Evita que se jogue fora página intensamente usada
- Inspeciona o bit R da página mais antiga
  - Se for 0, ela é velha (está no início da fila) e não foi usada recentemente $\rightarrow$ 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

Segunda Chance

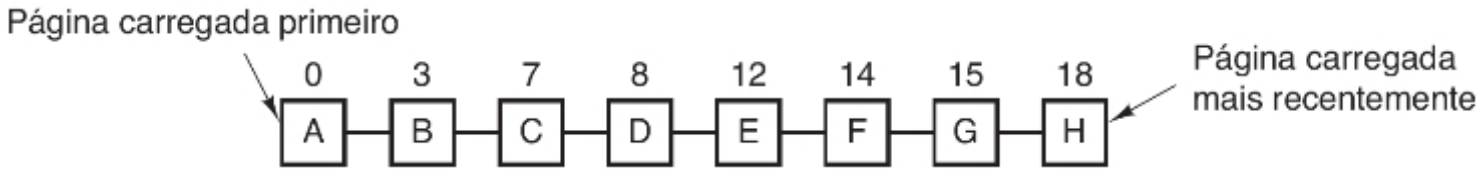
- Ex:



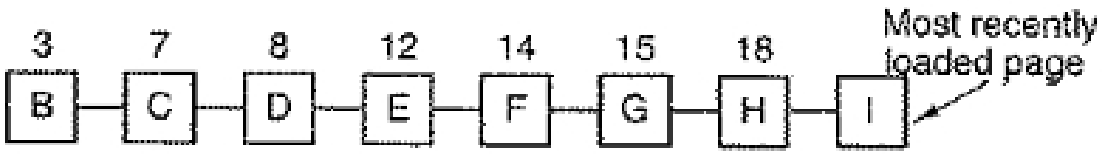
  - Ocorre page fault no tempo 20 e $R_A = 0$

Segunda Chance

- Ex:



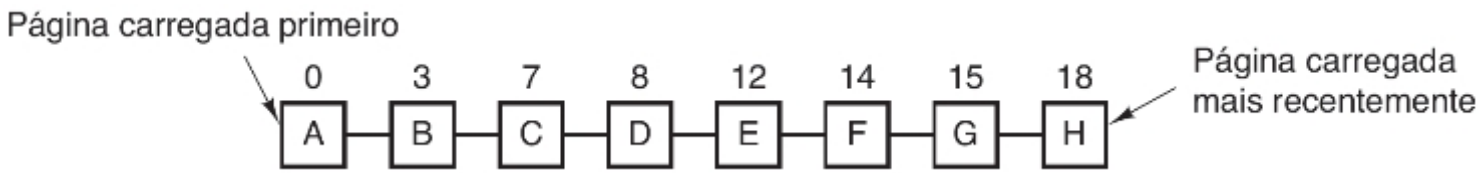
  - Ocorre page fault no tempo 20 e $R_A = 0$



  - A é removido, e o novo elemento é inserido ao final

Segunda Chance

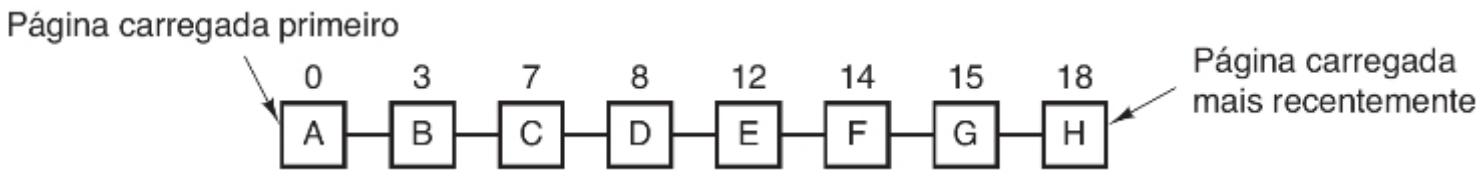
- Ex:



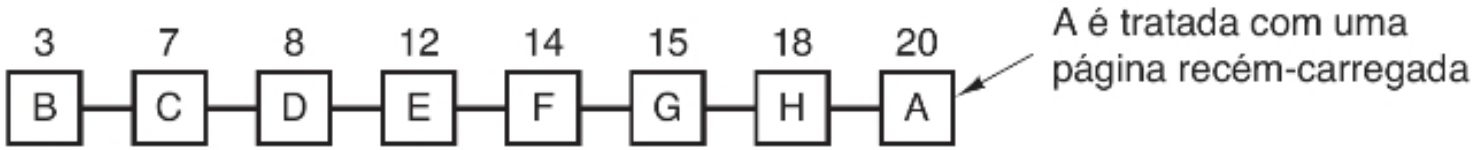
  - Ocorre page fault no tempo 20 e $R_A = 1$

Segunda Chance

- Ex:



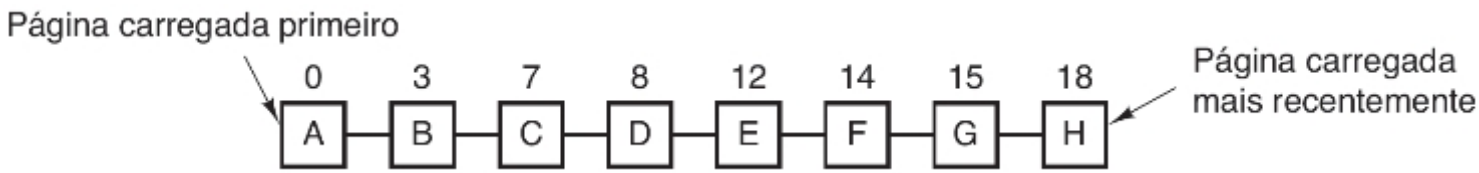
  - Ocorre page fault no tempo 20 e $R_A = 1$



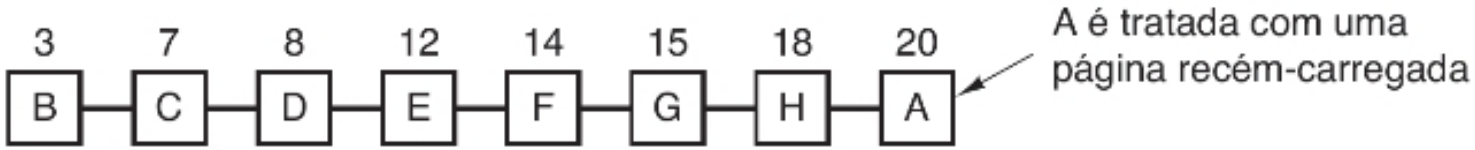
    - $R_A = 0$ e $\text{tempo}_A = 20$ agora

Segunda Chance

- Ex:



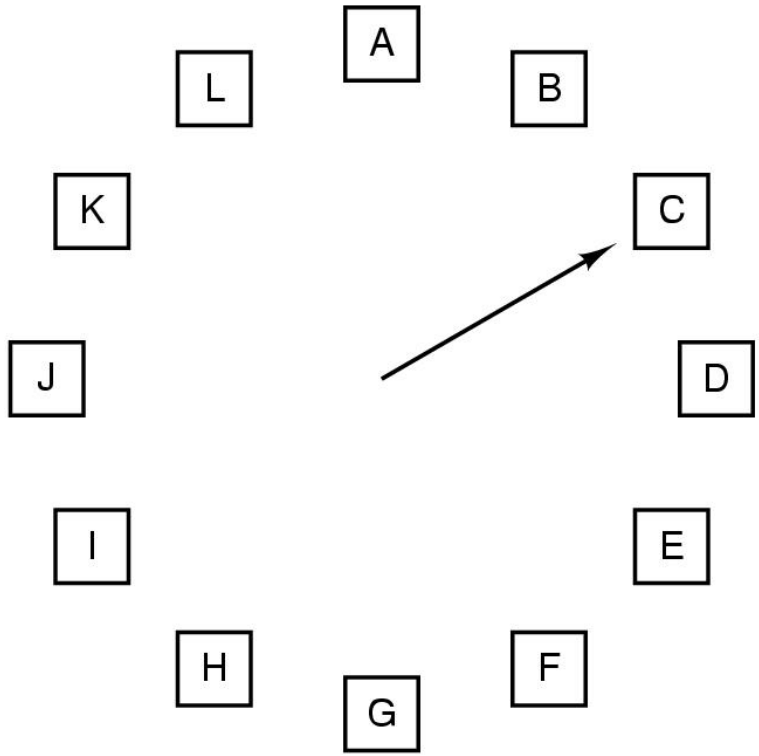
  - Ocorre page fault no tempo 20 e $R_A = 1$



    - $R_A = 0$ e $tempo_A = 20$ agora
  - Repete a operação com B
    - Se $R_B = 0$, troca
    - Senão, passa ao final da fila, com $R_B = 0$, e verifica-se C

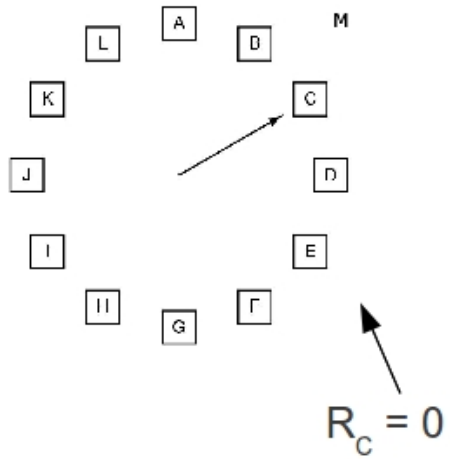
Algoritmo do Relógio

- Melhoria ao Segunda Chance:
  - Lista circular com ponteiro (cabeça) apontando para a página mais antiga, na forma de um relógio
  - Simplesmente uma implementação diferente do segunda chance



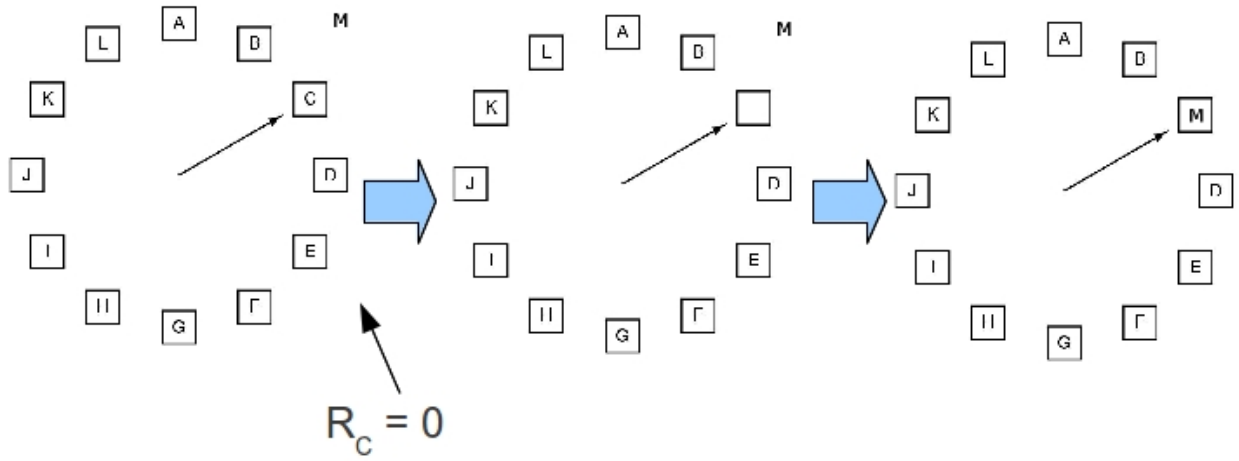
Algoritmo do Relógio

- Quando ocorre um page fault:
  - Inspeciona-se a cabeça da lista



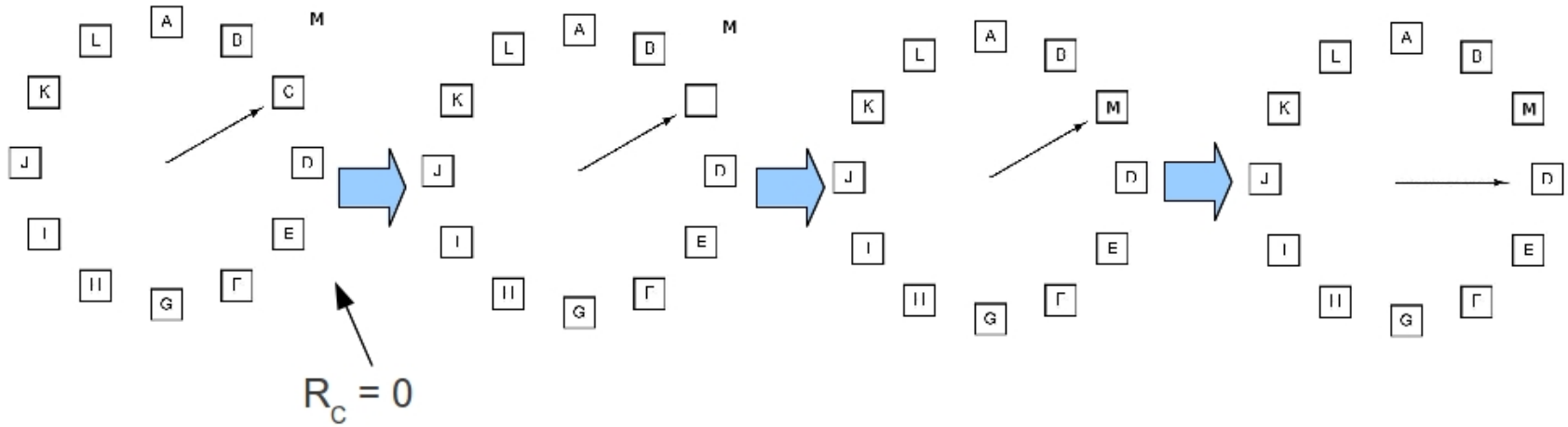
Algoritmo do Relógio

- Quando ocorre um page fault:
  - Inspeciona-se a cabeça da lista
  - Se R = 0:
    - Substitui-se a página da cabeça pela nova página



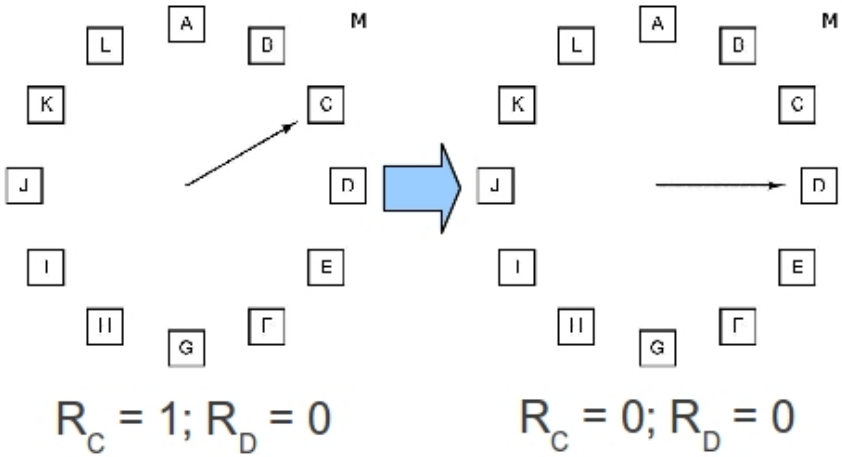
Algoritmo do Relógio

- Quando ocorre um page fault:
  - Inspeciona-se a cabeça da lista
  - Se R = 0:
    - Substitui-se a página da cabeça pela nova página
    - Avança-se a cabeça em uma posição



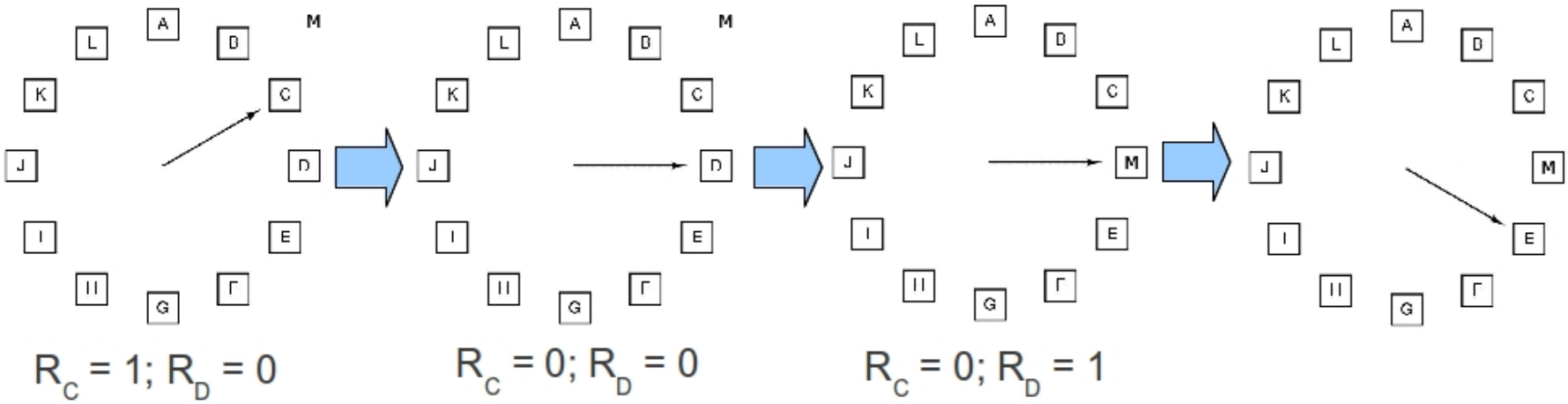
Algoritmo do Relógio

- Quando ocorre um page fault:
  - Se R = 1:
    - R é feito 0 e avança-se a cabeça em uma posição
    - Repete-se o processo até encontrar página com R = 0



Algoritmo do Relógio

- Quando ocorre um page fault:
  - Se R = 1:
    - R é feito 0 e avança-se a cabeça em uma posição
    - Repete-se o processo até encontrar página com R = 0
    - Age como no caso anterior (R = 0)



LRU – Least Recently Used

- 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 (tempo de última referência mais antigo)
- 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

LRU – Least Recently Used

- 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
    - Se política local: contador inicia com o processo
    - Se política global: contador inicia com início do sistema

LRU – Least Recently Used

- Hardware:
  - Após cada referência à memória, o valor atual de C é armazenado na entrada correspondente (página) na tabela
    - Tabela de páginas armazena o valor desse contador – C – em cada entrada
  - Em um page fault, o S.O. examina todas as entradas na tabela, para encontrar a com menor C

LRU – Hardware

- Alternativamente:
  - Se o computador tem n molduras, o hardware de LRU mantém uma matriz de n × n bits, inicialmente zero
  - Quando uma moldura k (ex, k=0) é referenciada, o hardware faz todos os bits da linha k serem 1, e os da coluna k serem 0

	0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0

	0	1	2	3
0	0	1	1	1
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0

LRU – Hardware (alternativo)

- A qualquer momento, a linha com o menor valor binário é a menos recentemente usada. A linha seguinte é a segunda menos recentemente usada, e assim por diante.
- Ex: referências às páginas 0, 1, 2, 3, 2

	0	1	2	3
0	0	1	1	1
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0

0	1	2	3
0	0	1	1
1	0	1	1
0	0	0	0
0	0	0	0

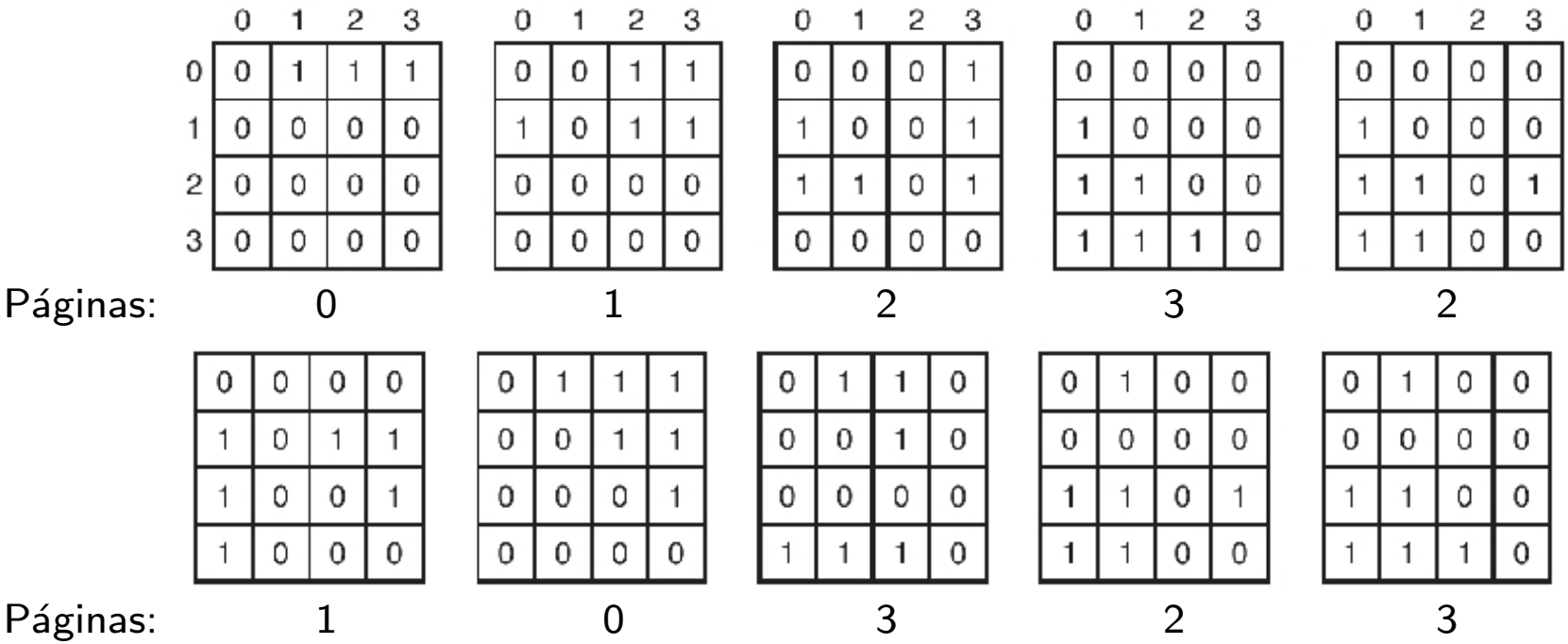
0	1	2	3
0	0	0	1
1	0	0	1
1	1	0	1
0	0	0	0

0	1	2	3
0	0	0	0
1	0	0	0
1	1	0	0
1	1	1	0

0	1	2	3
0	0	0	0
1	0	0	0
1	1	0	1
1	1	0	0

LRU – Hardware (alternativo)

- Ex:



NFU – Not frequently used

- Implementações em hardware da LRU são raríssimas $\rightarrow$ há que se simular em software
  - Simulação em Software – via NFU:
    - 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 (referência) ao contador correspondente
    - Conta, de fato, por quantos tiques do clock cada página foi referenciada

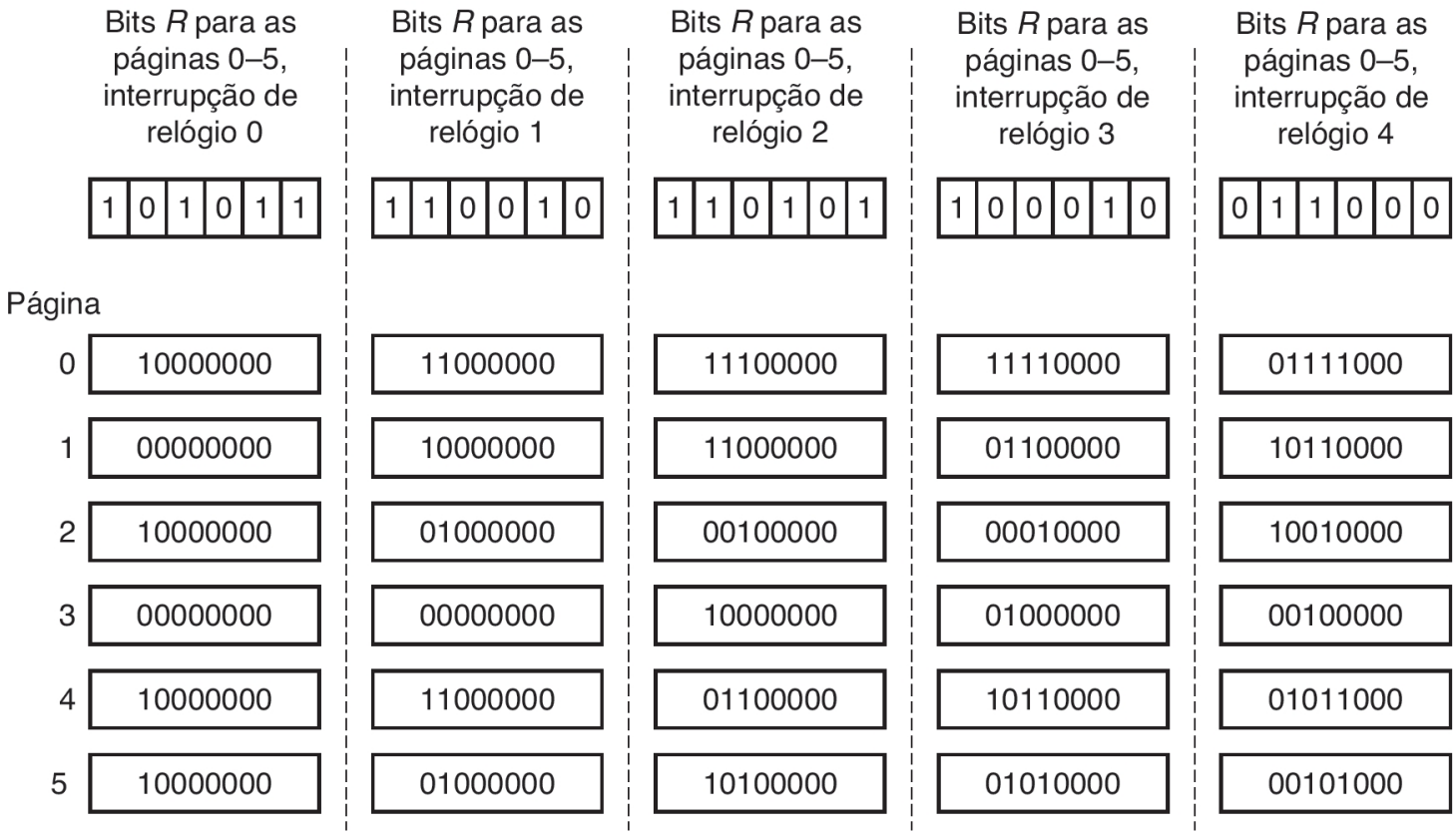
NFU – Not frequently used

- Simulação em Software – via NFU:
  - Em um page fault, escolhe a página com o menor contador
  - Problema:
    - Como esse algoritmo não se esquece de nada, páginas intensamente acessadas em uma porção pequena do código, mas que não mais serão acessadas, terão o contador alto e não serão candidatas a remoção

NFU – Aging

- Solução: Aging
  - Além de saber quantas vezes a página foi referenciada, também controla quando ela foi referenciada
  - Primeiro, os contadores são deslocados um bit à direita.
  - Só então o bit R é adicionado ao bit mais da esquerda (também a cada interrupção do clock)
  - Em um page fault, a página com o menor contador é removida
    - Páginas que não tenham sido referenciadas por mais tempo, terão mais zeros em seus bits mais significativos

NFU – Aging



Página	Bits *R* para as páginas 0–5, interrupção de relógio 0	Bits *R* para as páginas 0–5, interrupção de relógio 1	Bits *R* para as páginas 0–5, interrupção de relógio 2	Bits *R* para as páginas 0–5, interrupção de relógio 3	Bits *R* para as páginas 0–5, interrupção de relógio 4
	1 0 1 0 1 1	1 1 0 0 1 0	1 1 0 1 0 1	1 0 0 0 1 0	0 1 1 0 0 0
0	10000000	11000000	11100000	11110000	01111000
1	00000000	10000000	11000000	01100000	10110000
2	10000000	01000000	00100000	00010000	10010000
3	00000000	00000000	10000000	01000000	00100000
4	10000000	11000000	01100000	10110000	01011000
5	10000000	01000000	10100000	01010000	00101000

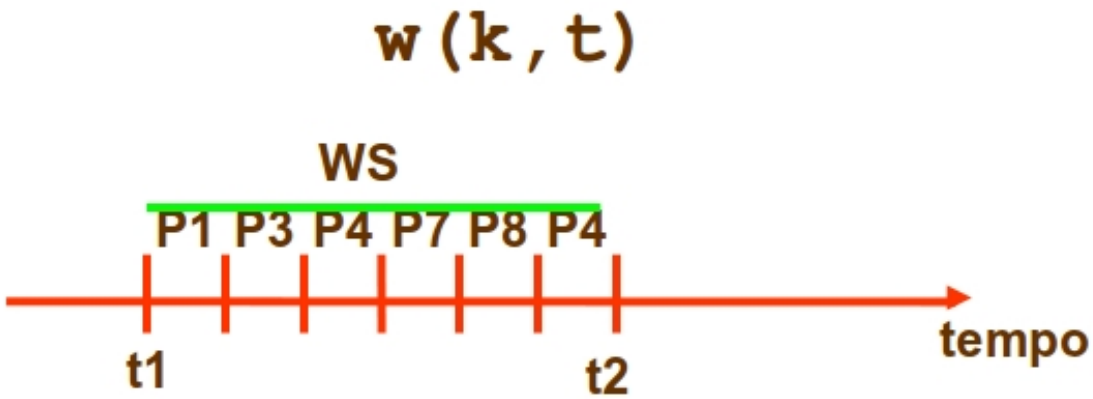
Note que após um máximo de 8 interrupções do clock uma página não referenciada tem seu contador zerado. Quanto mais tempo ficar sem ser referenciada, mais zeros à sua esquerda terá, e menor será seu contador

LRU – Linux

- Páginas agrupadas em duas listas: ativa e inativa
- Ativa
  - Inclui páginas que foram acessadas recentemente
- Inativa
  - Páginas que não foram acessadas por algum tempo
- Em caso de necessidade, uma página da lista de inativas é removida
  - Naturalmente, páginas trocam de lista, conforme são acessadas ou não

Working Set (WS)

- Propriedade apresentada pela maioria dos processos: Localidade de Referência
  - Durante qualquer fase de execução, o processo só vai referenciar uma fração relativamente pequena de suas páginas
- Conjunto de Trabalho (1968)
  - Conjunto de páginas que um processo está usando (referenciando) atualmente



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 ideia é determinar o working set de cada processo, em um determinado momento, e certificar-se de tê-lo na memória antes de rodar o processo – Modelo de Conjunto de Trabalho ou pré-paginação (1970)
- Working set w(k,τ)
  - Conjunto consistindo, em um dado instante τ, de todas as páginas usadas pelas k referências mais recentes à memória

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 da memória

Working Set (WS)

- Implementação:
  - Definir um valor para k
  - Contar as k referências mais recentes, contudo, é custoso $\rightarrow$ deve-se abandonar a ideia
  - Para simplificar, o working set pode ser visto como o conjunto de páginas que o processo referenciou durante os últimos τ segundos de sua execução
    - Conta o tempo individual (de CPU) do processo, descontando escalonamento $\rightarrow$ seu tempo virtual atual
  - Utiliza bit R e o tempo de relógio (tempo virtual) da última vez que a página foi referenciada

Working Set (WS) – Algoritmo

- Pressupostos:
  - O hardware define os bits R e M
  - Em cada interrupção do clock, o bit de referência é limpo
  - O tempo do working set se estende por várias interrupções do clock
  - O tempo virtual do último acesso de cada página é mantido em sua entrada na tabela de páginas
- Em cada page fault, a tabela de páginas inteira é buscada (para se encontrar candidata a remoção)
  - À medida em que cada entrada é processada, examine R

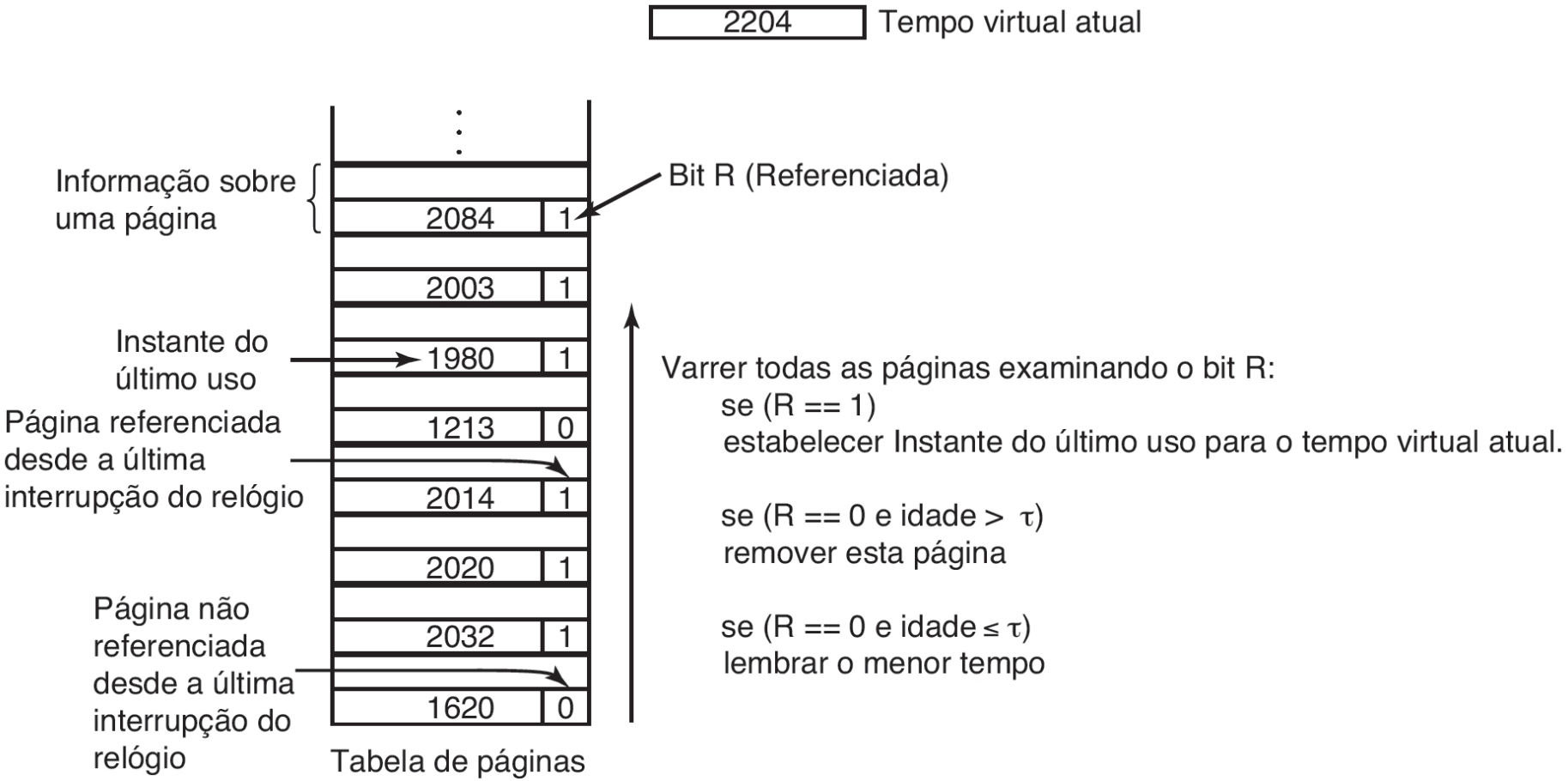
Working Set (WS) – Algoritmo

- Se R=1, copie o tempo virtual atual para o campo Tempo do Último Uso (TLU) na tabela de páginas, indicando que a página estava em uso no instante da page fault, ou seja, estava no working set
  - Não é candidata
- Se R=0, a página não foi referenciada no ciclo atual, e pode ser uma candidata.
  - Nesse caso, se sua idade (tempo virtual atual menos seu Tempo de Último Uso) for maior que o intervalo τ do working set, ela não está nele, e será removida
  - A nova página é então carregada aí
  - O procedimento continua atualizando as demais entradas

Working Set (WS) – Algoritmo

- Se, contudo, a idade for $\leq \tau$, a página é poupada. Ainda assim, a página com maior idade (menor Tempo de Último Uso) é marcada
- Se nenhum candidato for encontrado (todas as páginas estão no working set), substitua a página mais velha, dentre as com R=0
- Na pior das hipóteses, todas têm R=1
  - Nesse caso, escolhe-se uma aleatoriamente para remoção
  - Preferivelmente uma não modificada

Working Set (WS) – Algoritmo

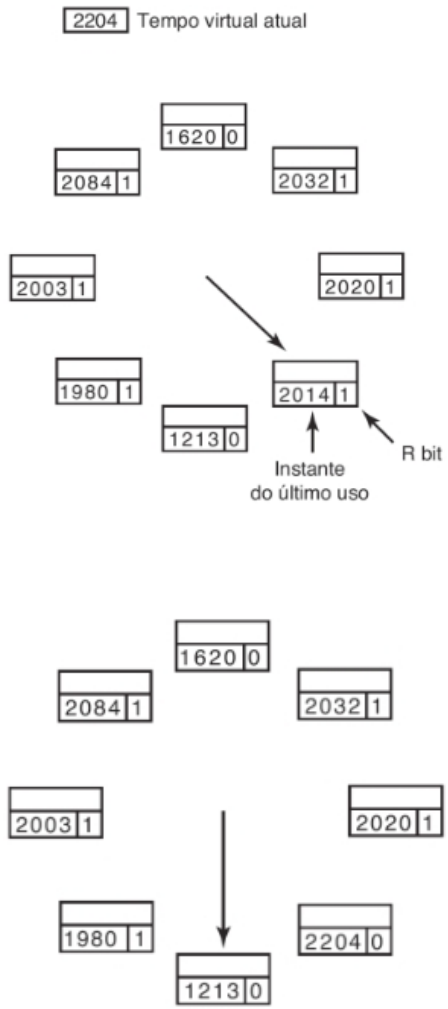


WSClock (1981)

- Clock + Working Set
- Amplamente usado, devido à sua simplicidade e desempenho
- Utiliza lista circular de molduras de páginas
  - Inicialmente vazia
  - À medida em que as páginas são carregadas, entram na lista, formando um anel
  - Cada entrada da lista contém o tempo de último uso, além dos bits R e M

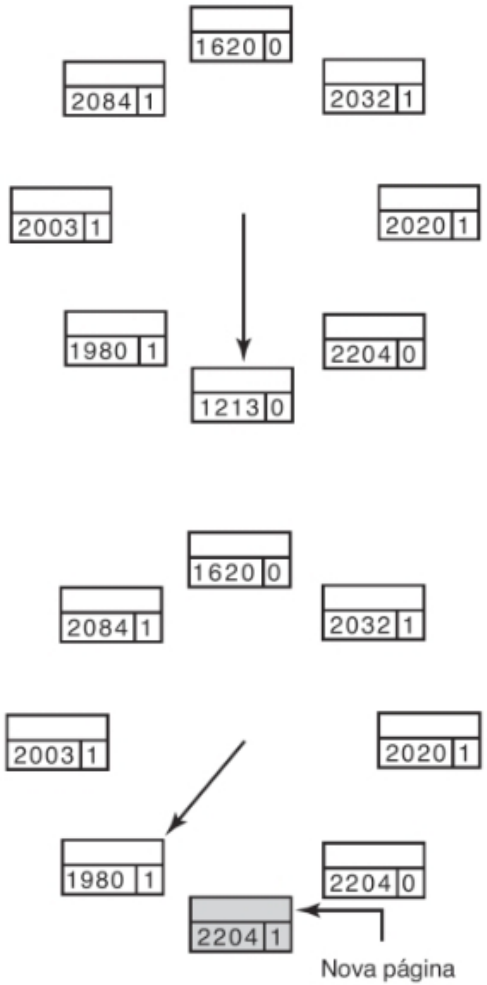
WSClock

- 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 $\rightarrow$ não é candidata a remoção
    - Copie o tempo virtual atual para o campo Tempo do Último Uso (TLU) na tabela de páginas
    - Faz R = 0 e avança a cabeça à próxima página, repetindo o algoritmo para esta página



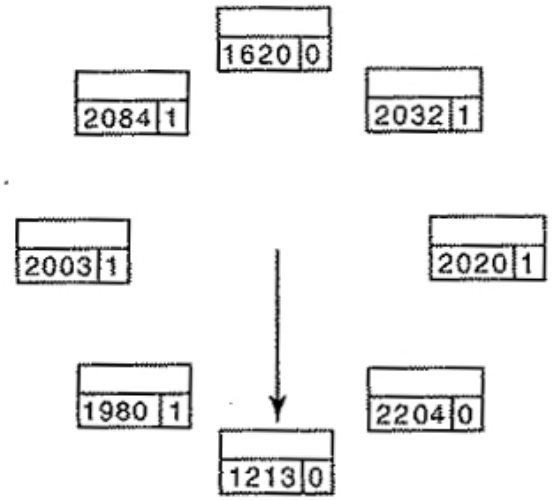
WSClock

- Funcionamento:
  - Se R=0
    - Se a idade da página for $\leq \tau$, ignore-a (está no working set)
    - Se a idade da página for maior que o intervalo τ e a página estiver limpa (M=0) $\rightarrow$ não está no working set e uma cópia válida existe no disco
    - A página é substituída
    - A cabeça da lista avança



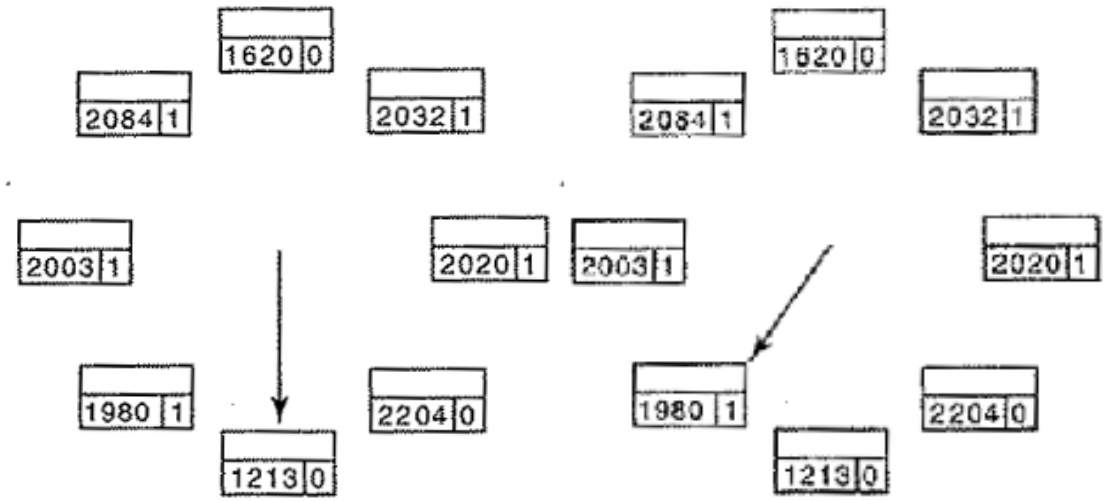
WSClock

- Funcionamento
  - Se R=0
    - Se, contudo, a idade da página for maior que o intervalo τ, mas a página estiver suja (M=1) $\rightarrow$ não possui cópia válida no disco
    - Agenda uma escrita ao disco (escalonada no driver de disco), evitando troca de processo
    - Avança a cabeça da lista, prosseguindo da página seguinte



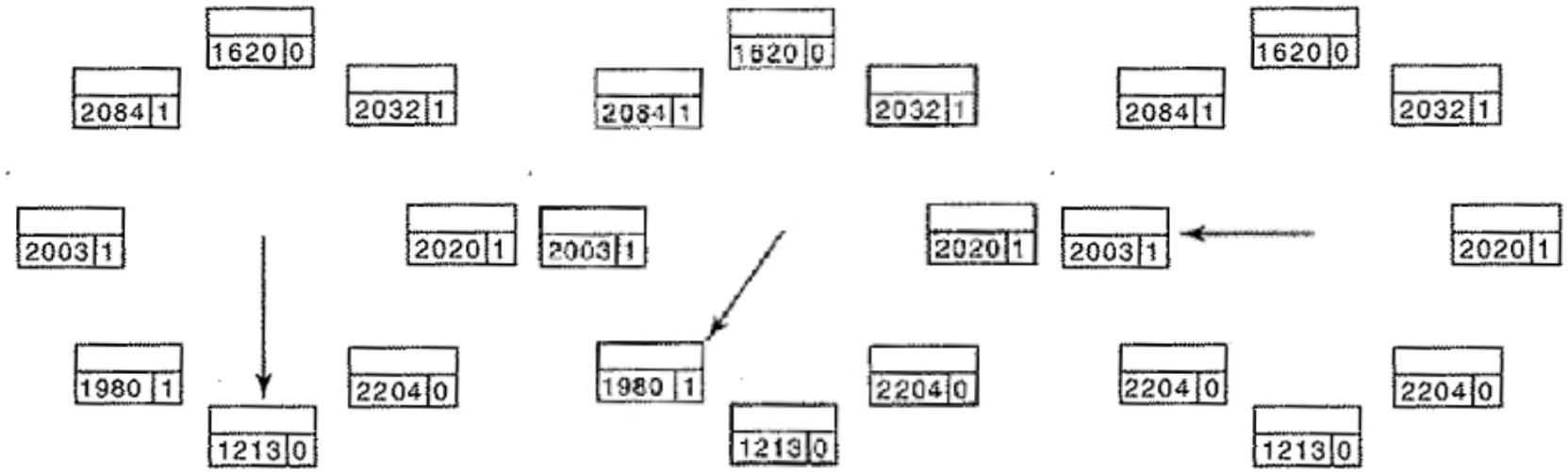
WSClock

- Funcionamento
  - Se R=0
    - Se, contudo, a idade da página for maior que o intervalo τ, mas a página estiver suja (M=1) $\rightarrow$ não possui cópia válida no disco
    - Agenda uma escrita ao disco (escalonada no driver de disco), evitando troca de processo
    - Avança a cabeça da lista, prosseguindo da página seguinte



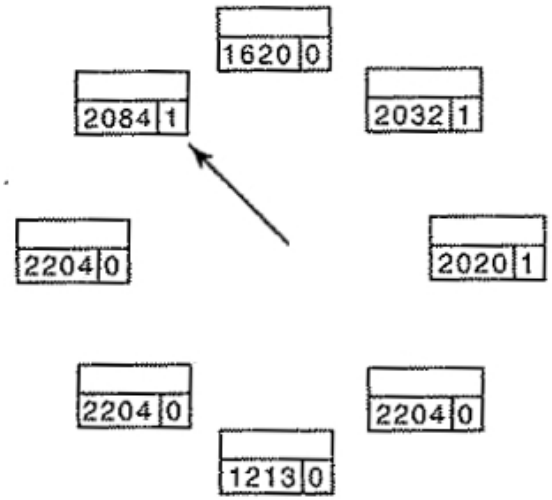
WSClock

- Funcionamento
  - Se R=0
    - Se, contudo, a idade da página for maior que o intervalo τ, mas a página estiver suja (M=1) $\rightarrow$ não possui cópia válida no disco
    - Agenda uma escrita ao disco (escalonada no driver de disco), evitando troca de processo
    - Avança a cabeça da lista, prosseguindo da página seguinte



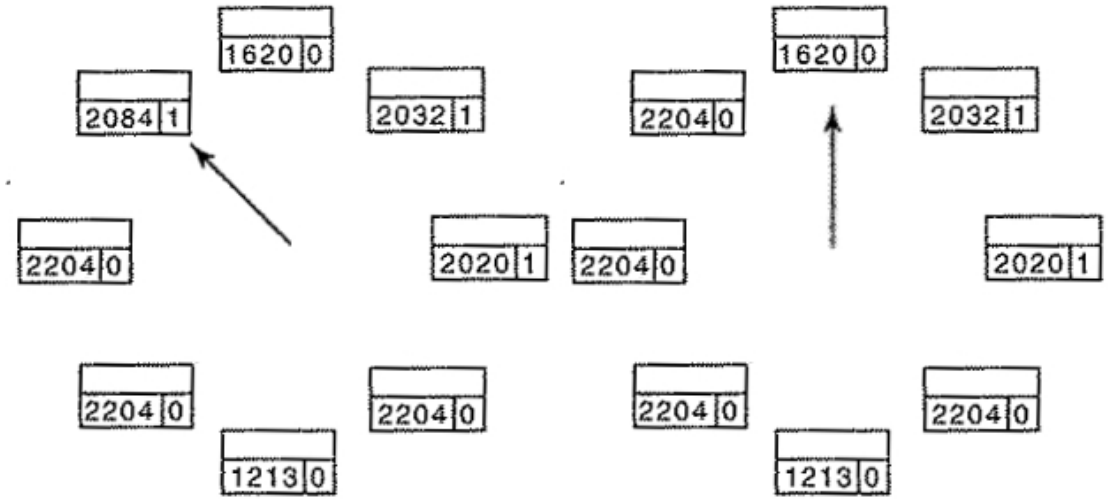
WSClock

- Funcionamento
  - Se R=0
    - Se, contudo, a idade da página for maior que o intervalo τ, mas a página estiver suja (M=1) $\rightarrow$ não possui cópia válida no disco
    - Agenda uma escrita ao disco (escalonada no driver de disco), evitando troca de processo
    - Avança a cabeça da lista, prosseguindo da página seguinte



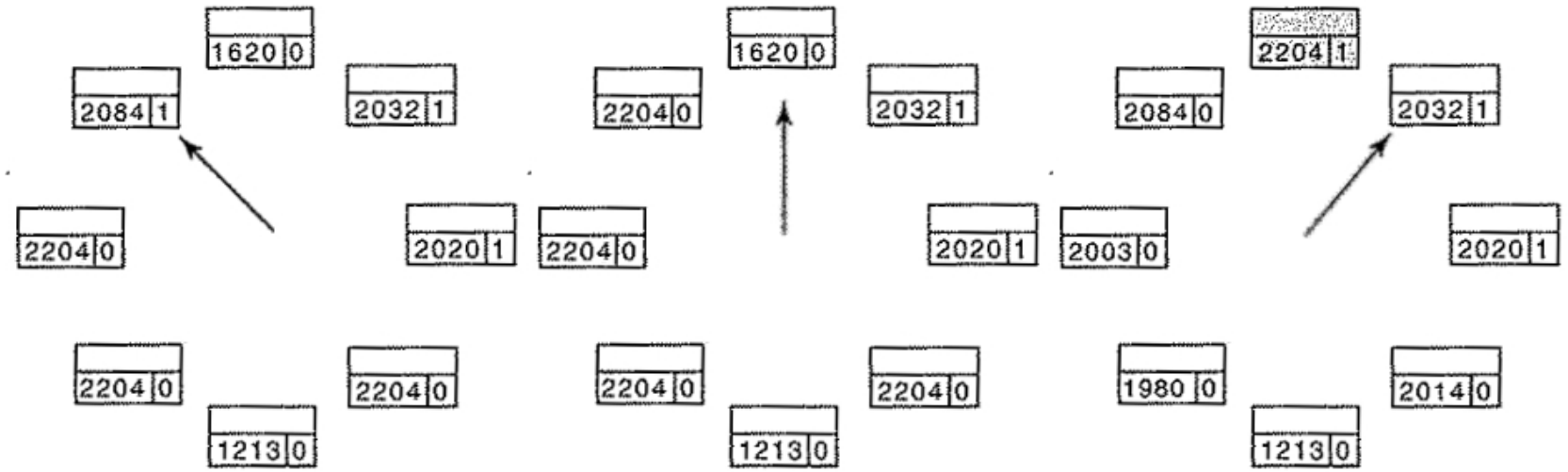
WSClock

- Funcionamento
  - Se R=0
    - Se, contudo, a idade da página for maior que o intervalo τ, mas a página estiver suja (M=1) $\rightarrow$ não possui cópia válida no disco
    - Agenda uma escrita ao disco (escalonada no driver de disco), evitando troca de processo
    - Avança a cabeça da lista, prosseguindo da página seguinte



WSClock

- Funcionamento
  - Se R=0
    - Se, contudo, a idade da página for maior que o intervalo τ, mas a página estiver suja (M=1) $\rightarrow$ não possui cópia válida no disco
    - Agenda uma escrita ao disco (escalonada no driver de disco), evitando troca de processo
    - Avança a cabeça da lista, prosseguindo da página seguinte



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
    - Alternativamente, pode-se escolher outro processo para rodar, enquanto aguarda
  - E nenhuma escrita foi agendada
    - Todas as páginas estão no working set
    - Na falta de informação adicional, substitua qualquer página limpa (M=0)
    - Se nenhuma página limpa existir, escreva a atual no disco

Algoritmos de Troca de Página

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

Algoritmos de Troca de Página

Algoritmo	Comentário
Ótimo	Não implementável, mas útil como um padrão de desempenho
NRU (não usada recentemente)	Aproximação muito rudimentar do LRU
FIFO (primeiro a entrar, primeiro a sair)	Pode descartar páginas importantes
Segunda chance	Algoritmo FIFO bastante melhorado
Relógio	Realista
LRU (usada menos recentemente)	Excelente algoritmo, porém difícil de ser implementado de maneira exata
NFU (não frequentemente usada)	Aproximação bastante rudimentar do LRU
Envelhecimento (*aging*)	Algoritmo eficiente que aproxima bem o LRU
Conjunto de trabalho	Implementação um tanto cara
WSClock	Algoritmo bom e eficiente

Melhores: Envelhecimento e WSClock

Referências Adicionais

- http://cs.winona.edu/Francioni/cs405/wsclock.html
- Bovet, D. P.; Cesati, M.: Understanding the Linux Kernel, 3 ed.