

## Gerência de Memória - Algoritmos de substituição de páginas

Quando ocorre um Page Fault, o S.O. deve escolher que página remover para abrir espaço em memória.

### Algoritmo NRU – Not Recently Used

É um algoritmo de substituição da página “não usada recentemente”

Na maioria dos computadores com memória virtual, as entradas nas tabelas de páginas têm 2 bits de status:

Reference bit (R)

Modified bit (M)

#### Algoritmo

- Qdo o processo é iniciado, os bits R e M das páginas são zerados;
- Bits são sempre alterados quando a página é referenciada/modificada;
- Periodicamente o bit R é zerado (por exemplo, a cada tique de clock);
- Quando acontece um Page fault, o S.O. inspeciona todas as páginas que encontram-se na memória e as separa em quatro categorias

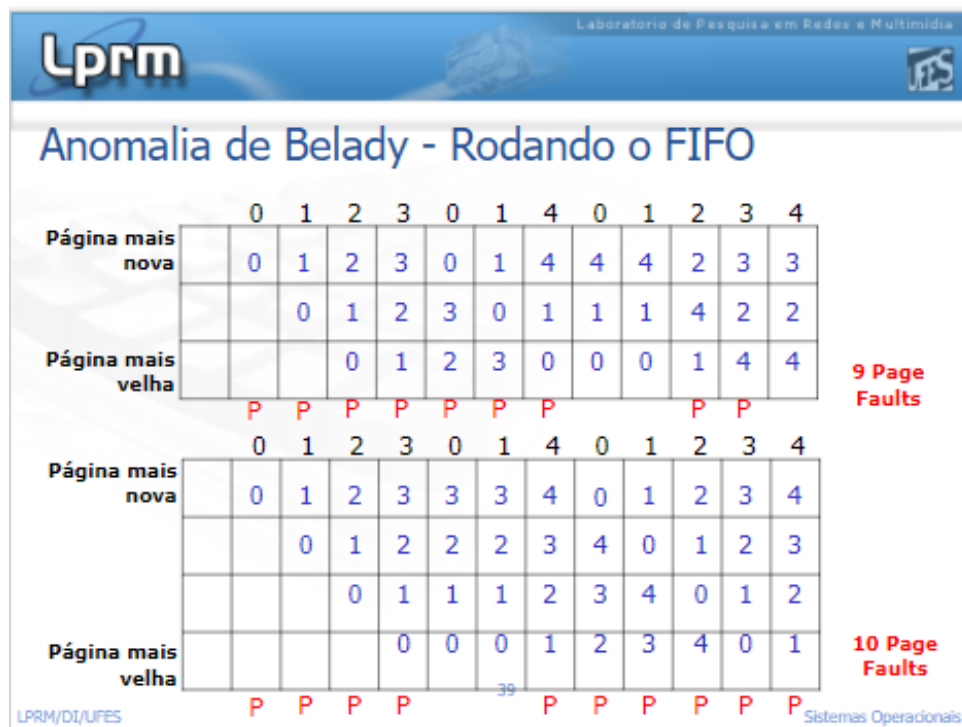
- Classe 0: *Not referenced, not modified* (R=0 , M=0)
- Classe 1: *Not referenced, modified* (R=0 , M=1)
- Classe 2: *referenced, not modified* (R=1 , M=0)
- Classe 3: *referenced, modified* (R=1 , M=1)

- O S.O. remove uma das páginas (aleatoriamente) da classe mais baixa não vazia.

### Algoritmo FIFO

- Mantém-se uma lista encadeada de páginas ordenada pela chegada das páginas à memória;
- Quando ocorre um Page Fault, a página no início da lista (que é a mais antiga) é a escolhida para a troca (A página mais antiga pode ser também uma página usada muito frequentemente, e isso é ruim para o algoritmo)

## Anomalia de Belady - Rodando o FIFO,

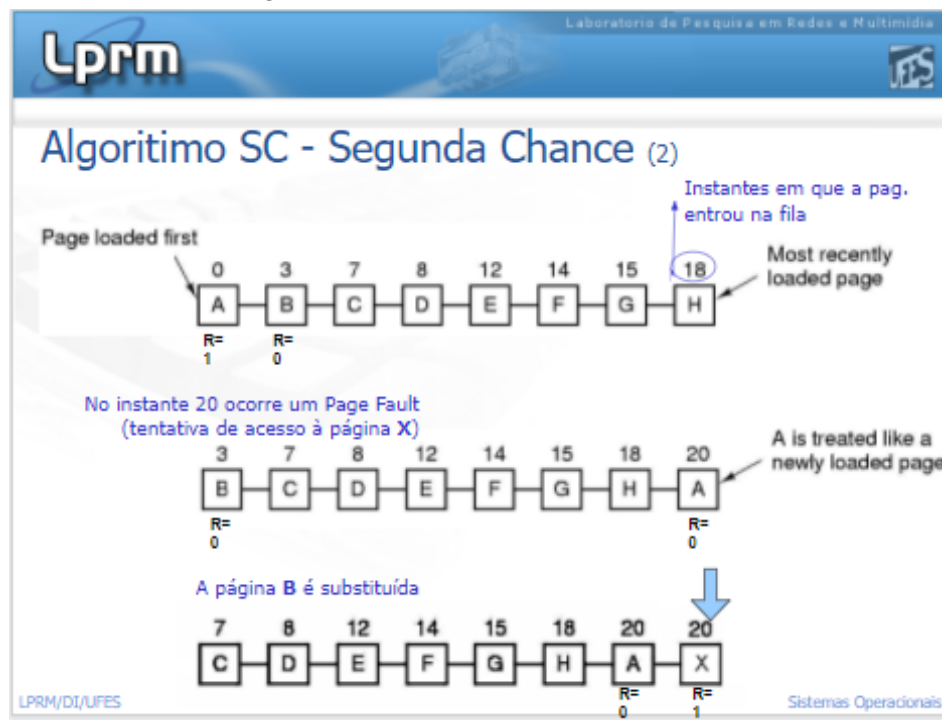


### Explicação:

- A fifo indica qual página irá sair da memória caso ocorra o page fault;
- Os quadrados representam a fila, e quando uma página (em preto) vem para a memória, ela entra na fila de controle;
- A ordem de saída é de baixo para cima, caso não haja espaço para adicionar uma página na fila de controle, ou seja, quem for o primeiro irá sair;
- Page fault, quando tenta acessar uma página (em preto) e ela não está na fila de controle e então é necessário substituir;
- Grande problema do fifo, ocorre quando tenta acessar uma página que foi acessada recentemente e a fila de controle para substituir as páginas não muda a ordem.
- Quando aumenta a quantidade de frames, o problema aumenta havendo mais page fault!

## Algoritmo SC - Segunda Chance

- Versão melhorada do FIFO
- Cada página tem um bit R (referenciada)
- Antes de remover a página mais antiga (cabeça da fila), seu bit R é verificado
  - Se  $R=0$ , a página é substituída (a página referenciada ocupará o seu lugar na memória)
  - Se  $R=1$ , a página vai para fim da fila, como se houvesse sido carregada agora e seu bit é setado para 0
  - Se todas as páginas tiverem seu bit  $R=1$ , haverá uma volta completa



### Explicação:

- Possui uma fila fifo, entretanto quando ocorre o page fault ele olha o bit "R";
- Caso o bit R for igual a 1, o sistema faz  $R = 0$  e joga a página para o final da fila;
- O algoritmo segue para a próxima página;
- Caso o bit R for igual 0, o sistema remove aquela página e adiciona outra no final da fila fazendo  $R=1$ ;

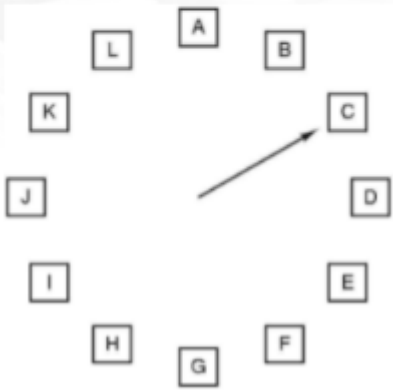
## Algoritmo do Relógio

- Visa melhorar o desempenho do algoritmo SC, diferenciando apenas na implementação da fila
- O ponteiro sempre aponta para a página mais antiga
- Na ocorrência de um Page fault
  - Se o bit R desta página for 0, ela é substituída, e o ponteiro “roda” uma casa
  - Se  $R=1$ , R é resetado e o ponteiro avança para a próxima página até encontrar uma página com  $R=0$

Lprm Laboratório de Pesquisa em Redes e Multimídia

### Algoritmo do Relógio

- Visa melhorar o desempenho do algoritmo SC, diferenciando apenas na implementação da fila



O diagrama ilustra o algoritmo do relógio em um formato circular. Doze caixas, rotas de A a L, estão dispostas em um círculo. Uma seta indica o ponteiro, que atualmente aponta para a caixa 'C'. As caixas são: A (topo), B (topo-direita), C (direita), D (baixo-direita), E (baixo), F (baixo-esquerda), G (esquerda), H (topo-esquerda), I (topo), J (topo-direita), K (esquerda) e L (topo).

- O ponteiro sempre aponta para a página mais antiga
- Na ocorrência de um Page fault
  - Se o bit R desta página for 0, ela é substituída, e o ponteiro “roda” uma casa
  - Se  $R=1$ , R é resetado e o ponteiro avança para a próxima página até encontrar uma página com  $R=0$

LPRM/DI/UFES 8 Sistemas Operacionais

### Explicação:

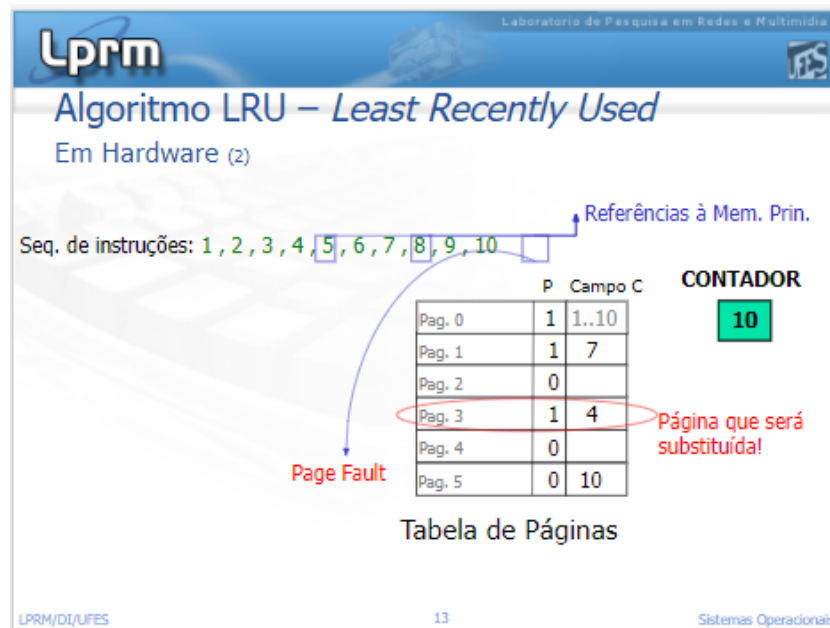
- Segue o mesmo esquema que o SC de segunda chance;
- Caso queira acessar uma página, o ponteiro irá apontar para o espaço da memória, caso for  $R=1$ , ele irá setar  $R=0$  e roda o ponteiro para o próximo espaço da memória;
- Se ocorrer de todos os bits R forem igual a 1, torna a ser igual o algoritmo SC de segunda chance.

### Algoritmo LRU – Least Recently Used

- É a personificação da localidade temporal
- Assume que as páginas usadas recentemente voltarão a ser usadas em breve;
- Substitui páginas que estão há mais tempo sem uso;
- Uma página acessada mais recentemente, tem mais chances de ser acessada novamente

Implementações (em hardware):

- Usar um contador C de 64 bits incrementado a cada instrução;
- Cada entrada da tabela de páginas deve ter um campo extra para armazenar o valor do contador
- A cada referência à memória o valor corrente de C é armazenado na entrada da tabela de páginas na posição correspondente à página referenciada
- Quando ocorre um Page Fault, a tabela de páginas é examinada, a entrada cujo campo C é de menor valor é a escolhida
- Substitui página com o menor valor no campo do contador (maior idade)



## LRU usando matrizes

- HW especial que mantém uma matriz  $n \times n$ , onde  $n$  é o número de molduras
- Inicialmente todos os bits da matriz são 0
- Sempre que a moldura  $k$  é referenciada, o hardware seta todos os bits da linha  $k$  para 1, e depois zera todos os bits da coluna  $k$  para zero
- Deste modo, a qualquer instante a linha com o menor valor binário é a menos recentemente usada.

Lprm Laboratório de Pesquisa em Redes e Multimídia

### Algoritmo LRU – *Least Recently Used*

Em Hardware (4)

■ LRU usando matrizes (cont.)

	Página na moldura 0				Página na moldura 1				Página na moldura 2				Página na moldura 3				Página na moldura 2			
	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
0	0	1	1	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	1	1	1	0	0	1	1	0	0	0	1	0	0	0
2	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	0	1	1	0	1
3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	1	0

	Página na moldura 1				Página na moldura 0				Página na moldura 3				Página na moldura 2				Página na moldura 3			
	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
0	0	0	0	0	0	1	1	1	0	1	1	0	0	1	0	0	0	1	0	0
1	1	0	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	1	0	0	0	1	0	0	0	0	1	1	0	1	1	1	0	0
3	1	0	0	0	0	0	0	0	1	1	1	0	1	1	0	0	1	1	1	0

LPRM/DI/UFES Sistemas Operacionais

## Algoritmo LRU – Least Recently Used (Software)

### Algoritmo Aging

- Após cada período/TICK, desloca o contador de 1 bit p/ a direita
- Soma R ao bit mais significativo do contador
- Feitas as somas, os bits R de cada página/frame são “zerados

	Bits R para páginas 0-5 em t0	Bits R para páginas 0-5 em t1	Bits R para páginas 0-5 em t2	Bits R para páginas 0-5 em t3	Bits R para páginas 0-5 em t4
<b>Página</b>	101011	110010	110101	100010	011000
0	1000000	1100000	1110000	1111000	0111000
1	0000000	1000000	1100000	0110000	1011000
2	1000000	0100000	0010000	0010000	1000100
3	0000000	0000000	1000000	0100000	0010000
4	1000000	1100000	0110000	1011000	0101100
5	1000000	0100000	1010000	0101000	0010100

### Explicação:

- Separa os bits em subgrupos
- Coloca os bits mais significativos nesse grupo deslocando todos os zeros para a direita e para o bit mais significativo, transfere para o início dos blocos
- Antes do clock resetar, o SO grava na memória utilizando o contador
- Os bits R das páginas são zerados
- O processo é repetido quando há um novo ciclo, e faz o shift para a direita, adicionando 1 e 0.

### Exemplo Prático:

Uma memória física consiste de 4 frames, atualmente contendo as páginas 0 a 3 de um processo (nessa ordem, isto é, frame 0 contém página 0, assim por diante). Este sistema roda o algoritmo de substituição de páginas Aging (com registradores de 5 bits). A tabela abaixo mostra a sequência de páginas referenciadas durante cada período d. Note que entre cada período há um clock tick. Determine qual página será substituída se um page fault ocorrer logo após o último período indicado na tabela (isto é, após o período no.7). Em caso de empate nos valores dos registradores, será escolhido o Frame com menor valor numérico.

Período (d)	0	1	2	3	4	5	6	7
Referências	3	1, 2	0	0, 3	1	2	2	1

Sequência de passos:			
Após período 0:	Após período 1:	Após período 2:	Após período 3:
frames: 0 1 2 3 Bits R: 0 0 0 1	frames: 0 1 2 3 Bits R: 0 1 1 0	frames: 0 1 2 3 Bits R: 1 0 0 0	frames: 0 1 2 3 Bits R: 1 0 0 1
Registradores: Frame 0: 0 0 0 0 0 Frame 1: 0 0 0 0 0 Frame 2: 0 0 0 0 0 Frame 3: 1 0 0 0 0	Registradores: Frame 0: 0 0 0 0 0 Frame 1: 1 0 0 0 0 Frame 2: 1 0 0 0 0 Frame 3: 0 1 0 0 0	Registradores: Frame 0: 1 0 0 0 0 Frame 1: 0 1 0 0 0 Frame 2: 0 1 0 0 0 Frame 3: 0 0 1 0 0	Registradores: Frame 0: 1 1 0 0 0 Frame 1: 0 0 1 0 0 Frame 2: 0 0 1 0 0 Frame 3: 1 0 0 1 0
Após período 4:	Após período 5:	Após período 6:	Após período 7:
frames: 0 1 2 3 Bits R: 0 1 0 0	frames: 0 1 2 3 Bits R: 0 0 1 0	frames: 0 1 2 3 Bits R: 0 0 1 0	frames: 0 1 2 3 Bits R: 0 1 0 0
Registradores: Frame 0: 0 1 1 0 0 Frame 1: 1 0 0 1 0 Frame 2: 0 0 0 1 0 Frame 3: 0 1 0 0 1	Registradores: Frame 0: 0 0 1 1 0 Frame 1: 0 1 0 0 1 Frame 2: 1 0 0 0 1 Frame 3: 0 0 1 0 0	Registradores: Frame 0: 0 0 0 1 1 Frame 1: 0 0 1 0 0 Frame 2: 1 1 0 0 0 Frame 3: 0 0 0 1 0	Registradores: Frame 0: 0 0 0 0 1 Frame 1: 1 0 0 1 0 Frame 2: 0 1 1 0 0 Frame 3: 0 0 0 0 1

## Working Set

- O conjunto de páginas que um processo está atualmente usando é denominado Working Set (espaço de trabalho);
- Se vários processos tiverem menos páginas em memória que o seu espaço de trabalho, o sistema pode entrar em colapso (trashing!!!);
- Trashing = O CPU passa mais tempo bloqueando e desbloqueando processos para poder trazer as páginas para a memória RAM do que executando os processos, a máquina não anda.
- Quanto mais frames alocamos para um processo, a tendência é que ocorram menos Page Faults neste processo....

## Working Set Clock (para resolver o problema de trashing)

- A troca de páginas só são avaliadas as páginas presentes em uma lista circular
- Cada entrada dessa lista possui os bits R e M, além de um timestamp (tempo da última referência)

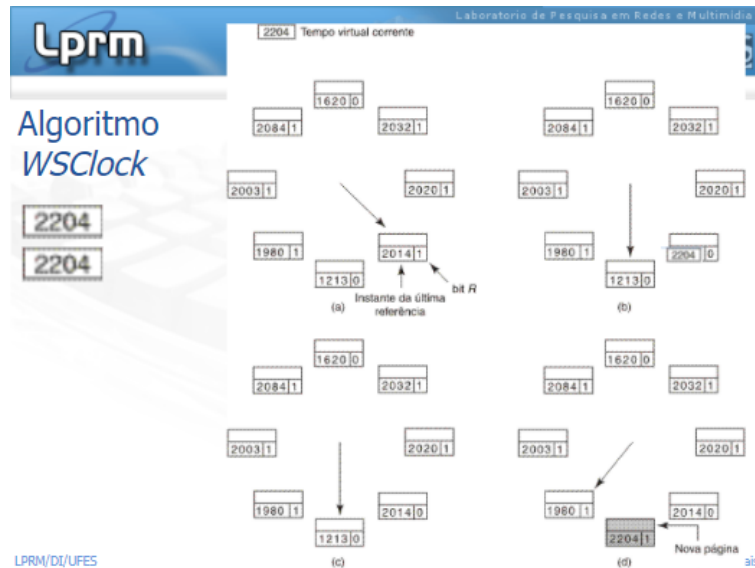


- À medida que as páginas são carregadas em memória, elas são inseridas na lista circular
- O algoritmo é executado quando precisa-se “liberar” molduras
 

Quando ocorrem page faults no processo em questão ou mesmo em outros processos (política Global de alocação de molduras)

Troca-se a primeira página (eventualmente, libera-se mais páginas) a partir da posição do ponteiro na lista que tenha  $R=0$  e cuja idade supera  $\tau$

Na verdade, verifica-se se a pag. está limpa (i.e. se ela não foi modificada). Caso ela tenha  $M=1$ , é escalonada uma escrita dessa pag. no disco e pula-se p/ a próxima página da lista circular.



### Exemplo prático

Para as questões a seguir, considere a seguinte "reference string"

1, 2, 3, 4, 2, 1, 2, 1, 4, 5, 4, 1, 5, 3, 5

FIFO		
Mem.	Fila	Acesso
_ _ _ _		1 (PF)
1 _ _ _	1	2 (PF)
1-2- _	1, 2	3 (PF)
1-2-3	1, 2, 3	4 (PF)
4-2-3	2, 3, 4	2
4-2-3	2, 3, 4	1 (PF)
4-1-3	3, 4, 1	2 (PF)
4-1-2	4, 1, 2	1
4-1-2	4, 1, 2	4
4-1-2	4, 1, 2	5 (PF)
5-1-2	1, 2, 5	4 (PF)
5-4-2	2, 5, 4	1 (PF)
5-4-1	5, 4, 1	5
5-4-1	5, 4, 1	3 (PF)
3-4-1	4, 1, 3	5 (PF)

LRU		
Mem.	Fila	Acesso
_ _ _ _		1 (PF)
1 _ _ _	1	2 (PF)
1-2- _	1, 2	3 (PF)
1-2-3	1, 2, 3	4 (PF)
4-2-3	2, 3, 4	2
4-2-3	3, 4, 2	1 (PF)
4-2-1	4, 2, 1	2
4-2-1	4, 1, 2	1
4-2-1	4, 2, 1	4
4-2-1	2, 1, 4	5 (PF)
4-5-1	1, 4, 5	4
4-5-1	1, 5, 4	1
4-5-1	5, 4, 1	5
4-5-1	4, 1, 5	3 (PF)
3-5-1	1, 5, 3	5

No caso de FIFO,

- Segue ordem da fila para retirada e entrada.

No caso da LRU

- Segue a ordem da página acessada recentemente (menor para o maior)
- Olhar o acesso antes de retirar da memória

