4.4 - Algoritmos de Troca de Páginas

- quando uma falta de página ocorre, o sistema operacional tem que escolher uma página para remover da memória a fim de dar lugar à que será trazida do disco;

- ... se a página a ser removida foi modificada enquanto estava na memória, ela deve ser reescrita no disco para manter a cópia em disco atualizada;

- ... se, todavia, a página não tiver sido alterada, a cópia do disco já está atualizada, não sendo necessária sua reescrita e assim a página a ser lida é simplesmente superposta à página que foi retirada.

- apesar de ser possível escolher uma página aleatória para dar lugar à página em demanda, o desempenho do sistema é melhorado se for escolhida uma página que tenha sido pouco referenciada/usada;

- ... se uma página muito usada é removida, ela provavelmente terá de ser trazida de volta em breve, resultando um esforço adicional e, portanto, algoritmos eficientes de troca de página visam minimizar este esforço.

4.4.1 - Troca Ótima de Página

- **idéia** - rotular cada página com o número de instruções que serão executadas antes que a própria páginas seja inicialmente referenciada;

- ... o algoritmo da página ótima simplesmente diz que a página com o maior rótulo deve ser removida, adiando-se o máximo possível a próxima falta de página.

- **problema do algoritmo** - ele não é realizável, pois no momento da falta de página, o sistema operacional não tem como saber quando cada página será referenciada (problema similar ocorre no algoritmo da menor tarefa primeiro);

- ... no máximo podemos executar um programa num simulador e, mantendo uma lista de todas as páginas referenciadas, implementar o algoritmo na segunda execução (usando as informações coletadas na primeira execução).

4.4.2 - Troca de Página Não Recentemente Usada

- **idéia** - considerando 02 bits, R e M indicando operação de leitura/escrita e de escrita na página, podemos construir um algoritmo de paginação utilizando esses bits;
- ... o **bit de referência** é ativado pelo *hardware* em qualquer leitura ou escrita na página, enquanto o **bit de modificação** é ativado pelo *hardware* quando uma página é escrita, p. ex., um byte é alterado e/ou adicionado à página;
- ... quando um processo é iniciado, ambos os bits de página para todas as páginas são declarados 0 pelo sist. oper. e, periodicamente, o bit R é zerado para distingüir páginas que não foram referenciadas recentemente daquelas que tenham sido;
- ... quando uma falta de página ocorre, o sist. oper. examina todas as páginas e as classifica em 4 categorias baseado nos valores correntes de seus bits R e M:

Classe 0: não referenciada, não modificada	Classe 1: não referenciada, modificada
Classe 2: referenciada, não modificada	Classe 3: referenciada, modificada

... 4.4.2 - Troca de Página Não Recentemente Usada

- Ainda que as páginas na Classe 1 pareçam impossíveis de existir, elas ocorrem quando as páginas da Classe 3 têm seu bit R zerado pela interrupção do relógio;
- ... interrupções de relógio não zeram o bit M porque esta informação é necessária para determinar se uma página terá que ser reescrita no disco ou não.
- O algoritimo *Não Recentemente Usada (Not Recently Used, NRU)*, remove uma página aleatória da classe não vazia de numeração mais baixa;
- ... implícito neste algoritmo é que é melhor remover uma página modificada que não foi referenciada pelo menos no último *tick* de relógio (tipicamente 20 mseg), que uma página não modificada mas muito usada;
- ... as características principais do NRU é que ele é fácil de entender, eficiente de se implementar, e gera um desempenho que, enquanto certamente não ótimo, é geralmente tido como adequado.

4.4.3 - Troca de Página FIFO

- **idéia** - de modo análogo ao NRU, o sistema operacional mantém uma lista de todas as páginas correntes na memória, sendo a página da cabeça da lista a mais velha e a página do fim a mais recente;
- ... em uma falta de página, a página da cabeça é removida e a nova página é acrescentada no final da lista ocupando, então, a posição da mais recente.
- ... para evitar o problema da retirada de uma página muito usada, uma variação do algoritmo é examinar os bits R e M da página mais velha;
- ... se a página pertencer a Classe 0, ela é retirada, senão a próxima página mais velha é examinada, e assim por diante;
- ... se páginas da Classe 0 não estão presentes na memória, então o algoritmo é repetido procurando por páginas da Classe 1, 2 e 3.

... 4.4.3 - Troca de Página FIFO

- outra variação do FIFO é o algoritmo *Segunda Chance* - que primeiro examinar a página mais velha como uma vítima potencial;
- ... se seu bit R é 0, a página é trocada imediatamente, senão o bit R é zerado e a página é colocada no fim da lista, como se tivesse acabado de chegar à memória;
- ... a pesquisa pode então prosseguir.
- **Idéia do Segunda Chance** - procurar por uma página velha que não tem sido referenciada no *tick* de relógio anterior. Se todas as páginas tiverem sido referenciadas, o algoritmo degenera-se e torna-se simplesmente um FIFO.
- ... uma pequena variação do *Segunda Chance* é guardar todas as páginas numa lista circular, ou seja, em vez de colocar as páginas no fim da lista, o ponteiro da cabeça da lista é simplesmente avançado uma página, produzindo o mesmo efeito.

4.4.4 - Troca de Página Menos Recentemente Usada

- Uma boa aproximação para o algoritmo ótimo é considerar que as páginas muito usadas nas últimas instruções, provavelmente o serão nas próximas instruções;
- ... igualmente, páginas que não têm sido usadas por um longo tempo provavelmente continuarão não sendo usadas (referenciadas);
- ... esta observação sugere um algoritmo realizável, ou seja, quando ocorre uma falta de página, retira-se a página que não tem sido usada por um tempo longo;
- ... esta estratégia é chamada *Least Recently Used* ou LRU.

... 4.4.4 - Troca de Página Menos Recentemente Usada

✯ Embora o algoritmo LRU seja teoricamente realizável, seu custo é alto, pois é necessário manter uma lista ligada de todas as páginas em memória, com a página mais recentemente usada no início e a menos recentemente usada no final;

✘ ... a dificuldade é que a lista deve ser atualizada para toda referência feita à memória;

✘ ... adicionalmente, encontrar a página na lista, removê-la de sua posição corrente, e movê-la para o início representa um esfôrço não desprezível.

✍ por ser proibitivo manipular uma lista ligada a toda instrução, mesmo em *hardware*, há 02 maneiras de implementar o LRU utilizando um *hardware* especial:

❶ equipar o *hardware* com um contador C de 64 bits que é automaticamente incrementado após a execução de cada instrução;
❷ para uma máquina com n *page frames*, o algoritmo LRU deve manter uma matriz de nxn bits, inicialmente todos zeros.

... 4.4.4 - Troca de Página Menos Recentemente Usada

❶ após cada referência de memória, o corrente valor de C é armazenado na entrada da tabela de páginas para a página referenciada;

- ... quando ocorre uma falta de página, o sistema operacional examina todos os contadores na tabela de páginas para achar o menor deles;

- ... a página correspondente é a menos recentemente usada.

❷ quando o *page frame* k é referenciado, o *hardware* primeiro ativa todos os bits da linha k para 1, atribuindo a todos os bits da coluna k o valor 0;

- ... em algum instante, a linha na qual o valor binário é o menor é a menos recentemente usada; a linha na qual o valor é o próximo superior é a segunda menos recentemente usada, e assim por diante.

... 4.4.4 - Troca de Página Menos Recentemente Usada

- Embora os algoritmos apresentados sejam realizáveis, eles são dependentes de *hardware* especial, e são de pouco uso para o projetista de sistema operacional construindo um sistema para uma máquina que não dispõe deste *hardware*.

- ... uma solução que possa ser implementada em *software* faz-se necessária e uma possibilidade é o algoritmo chamado de *Not Frequently Used* ou NFU.

- **idéia** - usar um contador em *software* associado a cada página, inicialmente zero.

- ... em cada *tick* de relógio, o sistema operacional pesquisa todas as páginas na memória e para cada página, o bit R, que é 0 ou 1, é adicionado ao contador;

- ... em suma, os contadores são uma tentativa de guardar a frequência com que cada página tem sido referenciada e, assim, quando uma falta de página ocorre, a página com o menor contador é escolhida para substituição.

... 4.4.4 - Troca de Página Menos Recentemente Usada

✘ **principal problema:** o algoritmo nunca esquece referências anteriores, ou seja, páginas muito (e não mais) refenciadas no começo da execução de um programa permanecem com um contador alto até o final da execução;

- ... felizmente, uma pequena modificação no NFU faz com que este seja capaz de simular LRU muito bem (o Algoritmo modificado é denominado *Aging*);

- NFU modificado = Algoritmo *Aging*:

❶ os contadores são cada um deslocados 1 bit para a direita, ou seja, são divididos por 2 antes do bit R ser incrementado;
❷ o bit R é incrementado no bit mais a esquerda.

- ... quando ocorre ume falta de página, a página de menor contador é removida.

... 4.4.4 - Troca de Página Menos Recentemente Usada

- ... é óbvio que a página que não tenha sido referenciada por, digamos, 04 *ticks* de relógio terá 04 zeros significativos em seu contador, tendo assim um valor mais baixo que o contador de uma página que tenha sido referenciada nos 04 últimos *ticks*.

✍ Uma diferença entre LRU e *Aging* é que, no último os contadores têm um número finito de bits (tipicamente 8). Portanto, não podemos classificar as páginas segundo referências anteriores à capacidade do contador.