Cap. 8 - Virtual Memory

8.1 - Estrutura de Controle e Hardware

8.2 - Sub-Sistema de Gerenciamento de Memória

8.3 - Gerenciamento de Memória no UNIX

8.4 - Gerenciamento de Memória no Windows NT

... Cap. 8 - Virtual Memory

✠ William STALLINGS; **Operating Systems: Internals and Design Principles**, New Jerssey, Prentice-Hall, 1998, ISBN: 0-13-887407-7

✠ Eleri CARDOZO; Maurício MAGALHÃES; Luís F. FAINA; **Sistemas Operacionais**, Dep. de Eng. de Computação e Automação Industrial, Fac. de Engenharia Elétrica e de Computação, UNICAMP, 1996.

8.1 - Estrutura de Controle e Hardware

✰ Ao compararmos os processos de paginação e segmentação com o particionamento fixo e dinâmico, observamos os princípios básico para o gerenciamento de memória:

- todas as referências dentro de um processo são endereços lógicos dinamicamente traduzidos para endereços físicos em tempo de execução;
- um processo pode ser particionado em páginas ou segmentos que não necessitam ser alocados de forma contígua na memória durante a execução.

✰ Com os 02 princípios presentes, então não é necessário que todas as páginas ou todos os segmentos de um processo estejam na memória durante a execução, entretanto:

- ... como todo este procedimento pode ser posto em prática?

... 8.1 - Estrutura de Controle e Hardware

✰ Há 02 implicações que conduzem a melhora de performance no sistema:

- mais processos podem ser mantidos na memória principal, posto que apenas algumas páginas/segmentos de cada processo são carregados;
- neste cenário, um processo pode ser maior que toda a memória principal, não obstante, o mesmo poderá ser carregado para a memória a fim de ser executado.

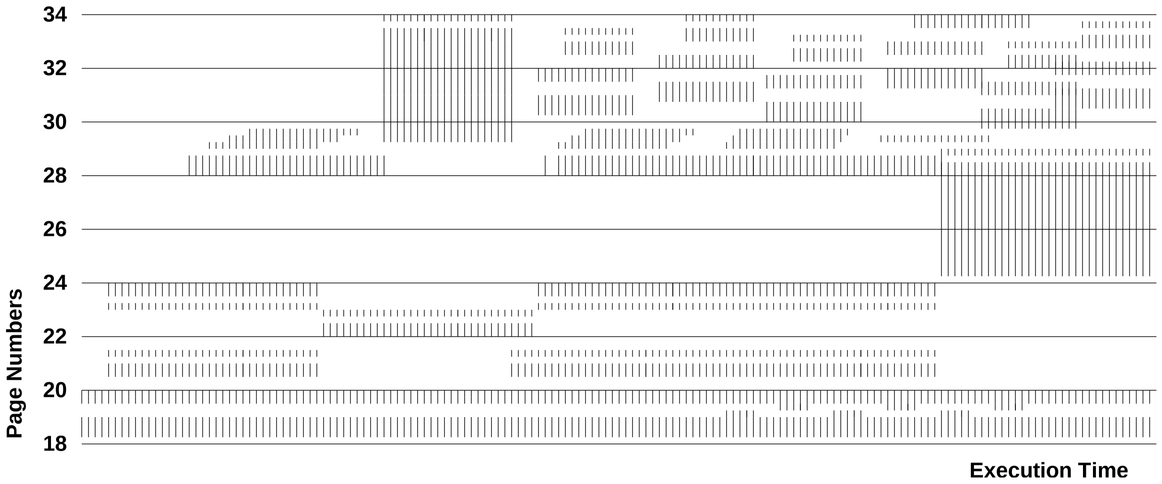
✰ Pelo fato do processo executar na memória principal, esta memória é referenciada como **memória real**, enquanto que os programadores ou usuários trabalham com uma memória maior – **memória virtual**.

- ... além de proporcionar uma multiprogramação efetiva, libera os usuários das preocupações desnecessárias acerca da memória principal.

8.1.1 - Memória Virtual e Princípio da Localidade

✰ Quanto a pergunta se esta abordagem é prática, experiências com inúmeros sistemas operacionais tem demonstrado sem dúvida que memória virtual funciona;

- ... muitas das justificativas são baseada na crença do princípio da localidade que estabelece que as referências as instruções e dados tendem a ser agrupadas.



... 8.1.1 - Memória Virtual e Princípio da Localidade

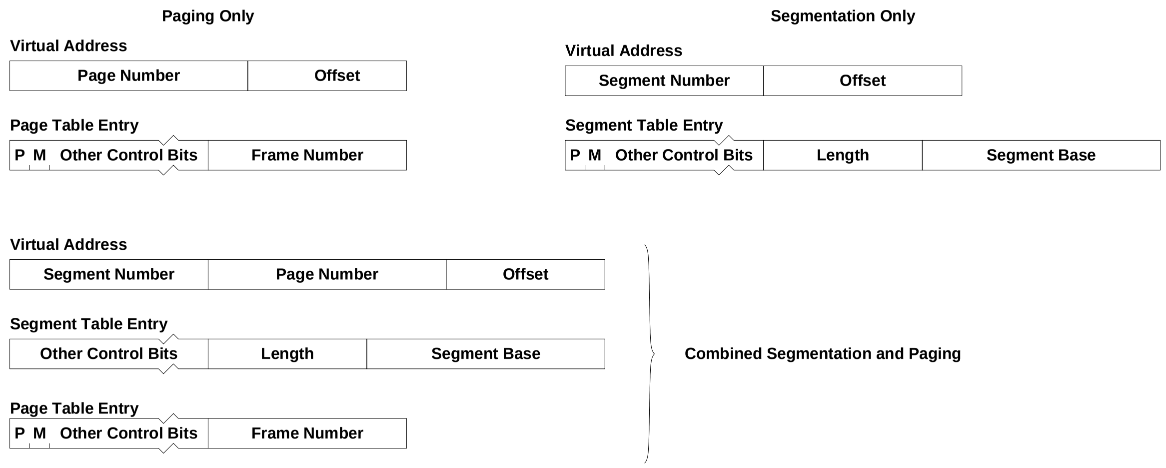
✰ Para que este esquema seja prático e efetivo, faz-se necessário suporte de *hardware* e *software* do Sistema Operacional para o esquema de paginação e/ou segmentação.

Simple Paging	Simple Segmentation	Virtual Memory Paging	Virtual Memory Segmentation
Main memory partitioned into small fixed−size chunks called frames.	Main memory not partitioned.	Main memory partitioned into small fixed−seze chunks called frames.	Main memory not partitioned.
Program broken into pages by the compiler or memory management system.	Program segments specified by the programmer to the compiler (i.e., the decision ismade by the programmer)	Program broken into pages by the compiler or memory management system.	Program segments specified by the programmer to the compiler (i.e., the decision is made by the programmer).
Internal fragmentation within frames.	No internatl fragmentation.	Internal fragmentation within frames.	No internal fragmentation.
No external fragmentation.	External fragmentation.	No external fragmentation.	External fragmentation.
Operating system must maintain a page table for each process showing which frame each page is in.	Operating system must maintain a segment table for each process showing the load address and length of each segment.	Operating system must maintain a page table for each process showing which frame each page is in.	Operating system must maintain a seg−ment table for each process showing the load address and length of each segment.
Operating system must maintain a free frame list.	Operating system must maintain a list of free holes in main memory.	Operating system must maintain a free frame list.	Operating system must maintain a list of free holes in main memory.
Processor uses page number, offset to calculate absolute address.	Processor uses segment number, offset to calculate absolute address.	Processor uses page number, offset to calculat4e absolute address.	Processor uses segment number, offset to calculate absolute address.
All the pages of a process must be in main memory for process to run, unless overlays are used.	All the segments of a process must be in main memory for process to run, unless overlays are used.	Not all pages of a process need be in main memory frames for the process to run. Pages may be read in as need.	Not all segments of a process need be in main memory frames for the process to run. Segments may be read in as need.
		Reading a page into main memory may require writing a page out to disk.	Reading a segment into main memory may require writing one or more segments out to disk.

8.1.2 - Memória Virtual e Paginação

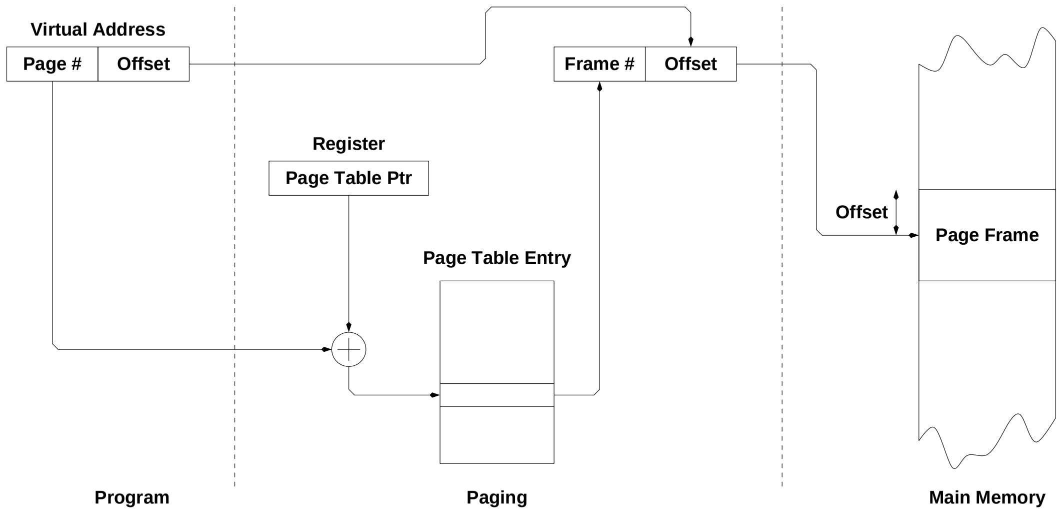
✰ O termo Memória Virtual é usualmente associado a sistemas que empregam a paginação, entretanto, memória virtual baseada em segmentação também é empregada;

- ... na forma mais simples de paginação, cada processo tem sua própria tabela de páginas e quando todas as suas páginas estão carregadas, a tabela de página para um processo é criada e carregada para a memória principal.



... 8.1.2 - Memória Virtual e Paginação

✰ O processo de leitura de uma palavra da memória principal envolve a tradução de endereço virtual, consistindo o número da página e *offset*, para endereço físico, consistindo do número do *frame* e *offset*.



... 8.1.2 - Memória Virtual e Paginação

✰ Na maioria dos sistemas, há uma tabela de página por processo e, claramente, a quantidade de memória devotada para tabela de páginas pode ser inaceitável;

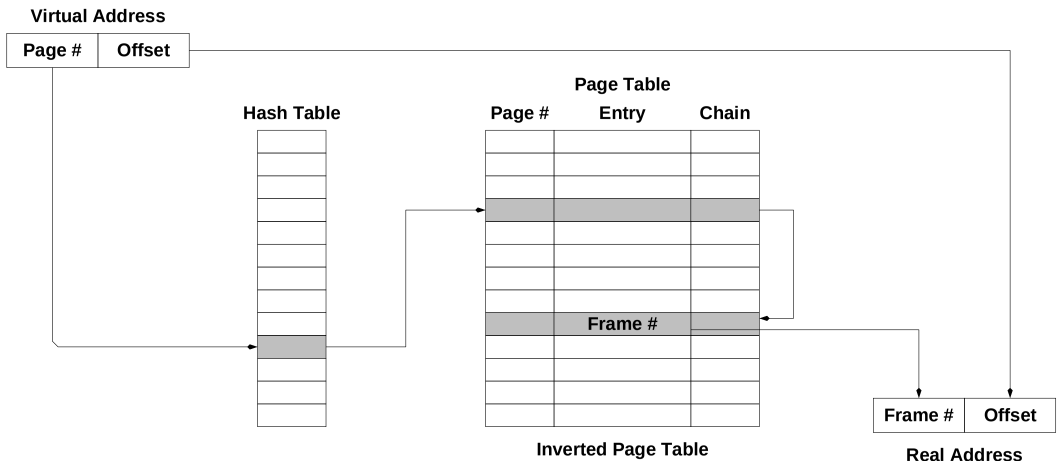
- ... para resolver este problema, muitos sistemas armazenam a tabela de páginas na memória secundária em vez de armazená-la na memória principal, o que por outro lado, significa que estão sujeitas às mesmas regras de paginação.

✰ **Alternativa:** uso de 02 níveis na organização da tabela de páginas, neste caso, há um diretório de páginas no qual cada entrada aponta para um tabela de páginas;

- ... assim se o comprimento do diretório é x e se o máximo comprimento da tabela é y, então um processo pode consistir de $x * y$ páginas.

... 8.1.2 - Memória Virtual e Paginação

✰ Um outra abordagem é a Tabela de Páginas Invertida (Power PC e IBM AS/400) onde o número da página é mapeado numa tabela $hash$ fornecendo um ponteiro para a tabela de páginas invertida, que contém entradas da tabela de página.



... 8.1.2 - Memória Virtual e Paginação

✰ A princípio, toda referência à memória virtual causa 02 acessos: um para buscar a entrada apropriada da tabela de páginas; e um para buscar o dado desejado;

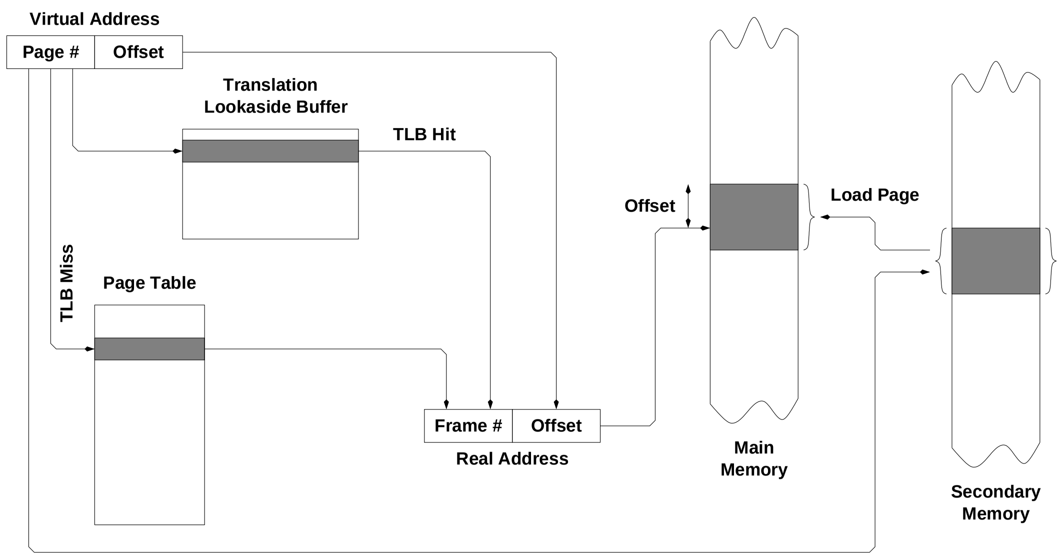
- ... o esquema direto de memória virtual pode dobrar o tempo de acesso à memória, inviabilizando a utilização do esquema no gerenciamento de memória.

✰ Para tratar este problema, muitos esquemas de memória virtual empregam um *cache* para entradas da tabela de páginas — **Translation Lookaside Buffer**.

- ... estas funções da *cache* são as mesmas que aquelas da *cache* de memória principal e contém as entradas mais recentemente referenciadas da tabela de páginas.

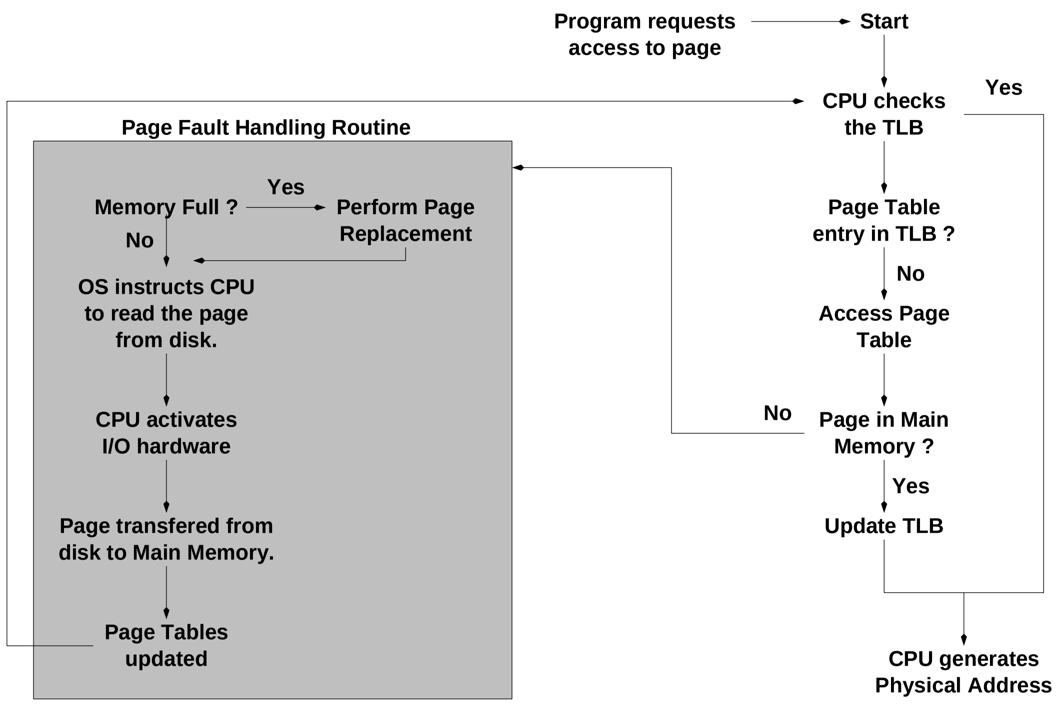
... 8.1.2 - Memória Virtual e Paginação

✰ *Cache* para entradas da Tabela de Páginas — **Translation Lookaside Buffer**.



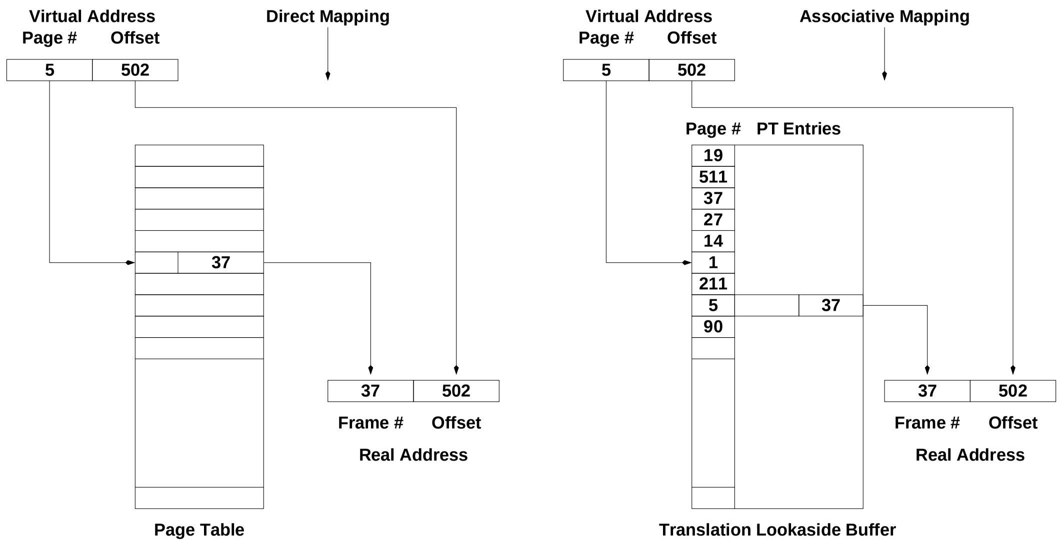
... 8.1.2 - Memória Virtual e Paginação

✰ O diagrama abaixo ilustra o uso da *cache* TLB:



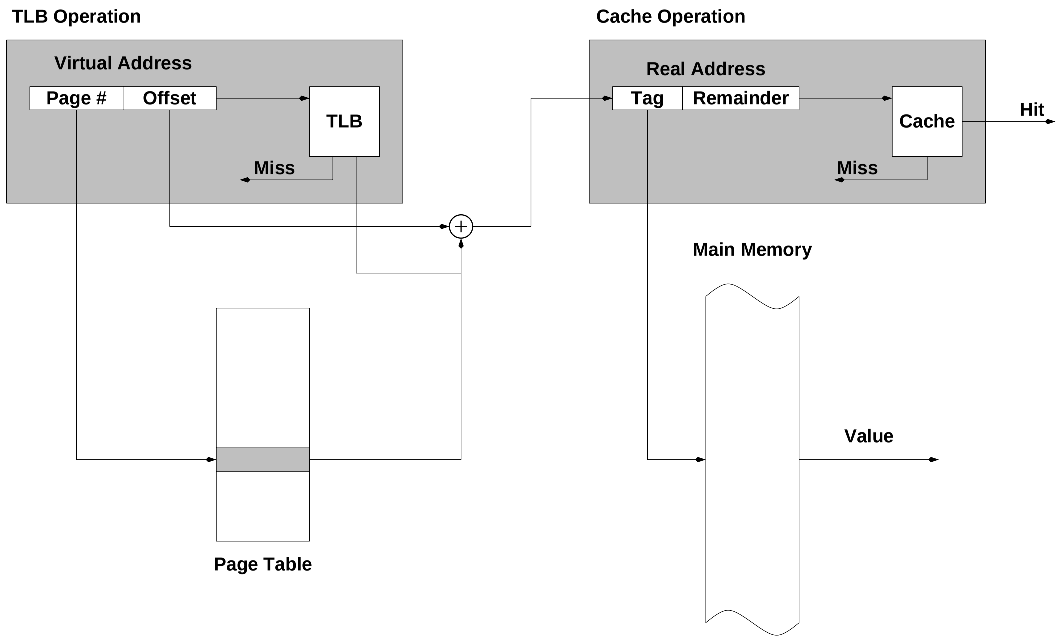
... 8.1.2 - Memória Virtual e Paginação

✰ Pelo fato da TLB conter algumas das entradas da Tabela de Páginas, não se pode indexar com base somente no número da página, outros dados fazem-se necessários:



... 8.1.2 - Memória Virtual e Paginação

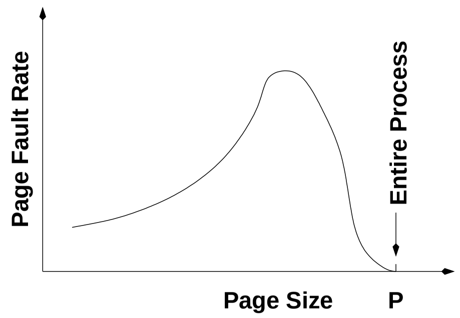
✰ Finalmente, o mecanismo de memória virtual precisa interagir com a *cache* do sistema para efetuar a leitura do dado (palavra) da *cache* ou da memória principal.

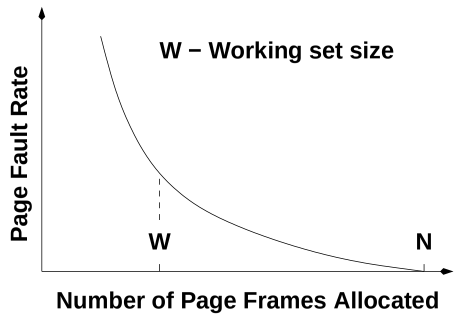


... 8.1.2 - Memória Virtual e Paginação

✰ Um aspecto importante de projeto no *hardware* é o tamanho da página e, vários são os fatores a considerar:

- **fragmentação interna:** por um lado, páginas menores significam menor fragmentação interna mas por outro maior será o número de páginas por processo;
- **características físicas da memória secundária:** favorece às páginas maiores quanto a eficiência na transferência dos blocos de dados;
- **freqüência de falta de página:** a freqüência com que faltas de páginas ocorrem é dependente do tamanho da página (gráfico abaixo).





8.1.3 - Memória Virtual e Segmentação

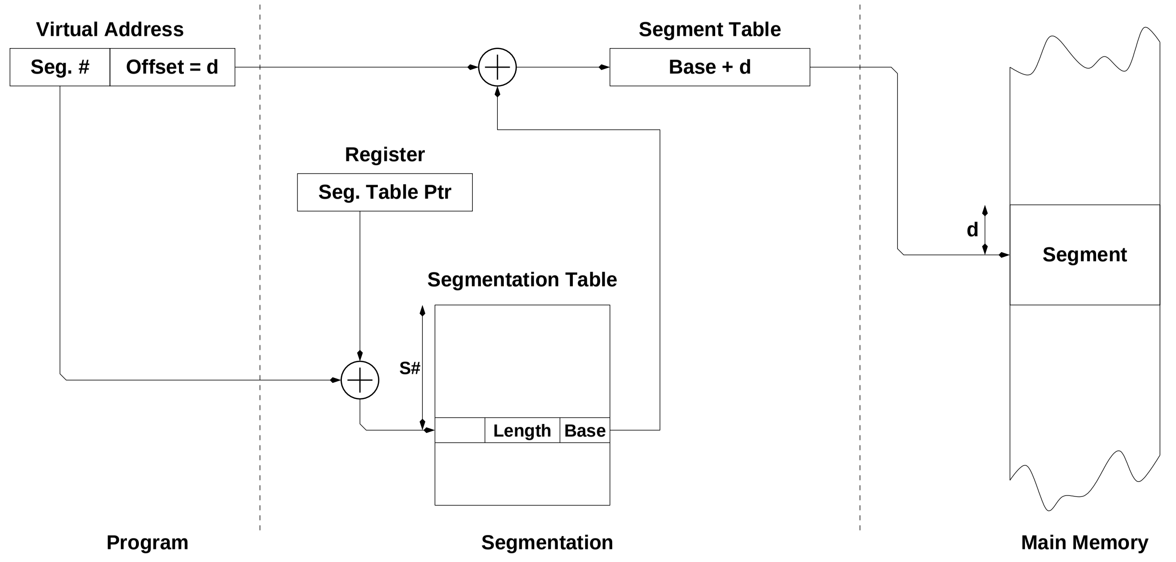
✡ Segmentação possibilita que a memória seja particionada em múltiplos segmentos cujos tamanhos podem ser diferentes bem como mudar no tempo (dinâmico).

✡ Vantagens da segmentação sobre sistemas não segmentados:

- simplifica o tratamento de estruturas de dados que exigem mais espaço de memória durante o tempo de execução do processo;
- possibilita que programas sejam alterados e recompilados independentemente, sem exigir que todo o conjunto seja *relinked* ou recarregado;
- presta-se para o compartilhamento de dados entre processos, posto que esta região da memória pode ser referenciada por mais de um processo;
- presta-se à própria proteção, posto que pode ser reservado para acomodar um conjunto de programas e dados bem definidos (aspectos de segurança).

... 8.1.3 - Memória Virtual e Segmentação

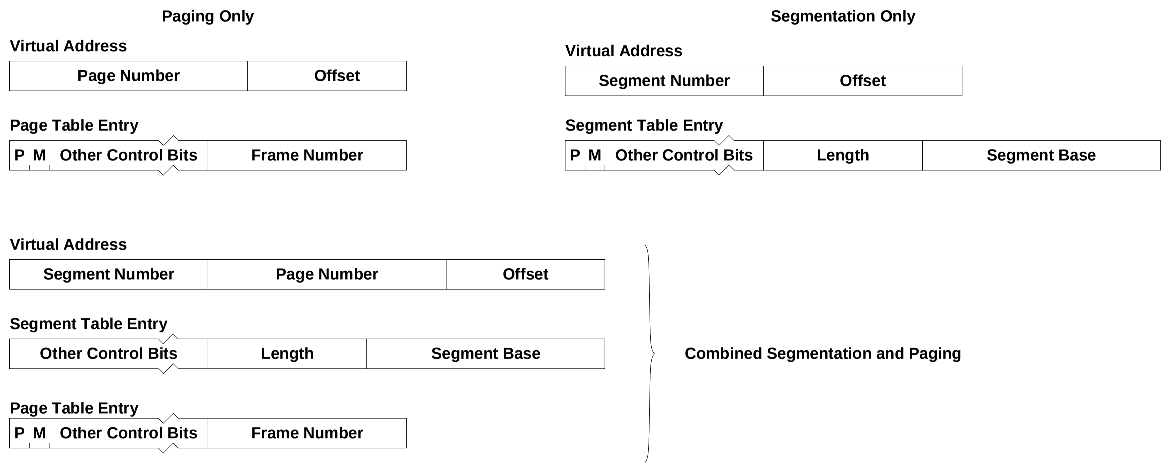
✡ **Implementação do Hardware:** pelo fato da tabela de segmentos ser de tamanho variável dependendo do tamanho do processo, não se espera que seja mantida em registradores, mas sim na memória principal.



... 8.1.3 - Memória Virtual e Segmentação

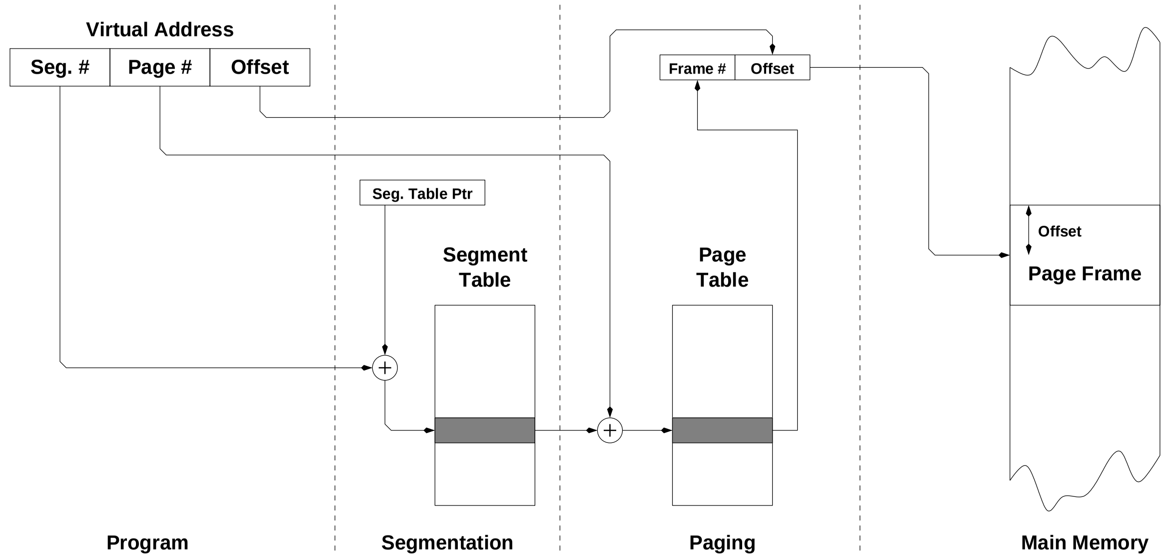
✠ Ambas paginação e segmentação têm suas vantagens:

- **paginação:** transparente para o programador da aplicação, elimina a fragmentação externa oferecendo uso eficiente da memória principal;
- **segmentação:** visível ao programador, oferece a habilidade de tratar alocação dinâmica de dados, modularidade e suporte ao compartilhamento e proteção.



... 8.1.3 - Memória Virtual e Segmentação

✠ Estrutura de suporte para Paginação e Segmentação combinadas:



8.1.4 - Mecanismos de Proteção e Compartilhamento

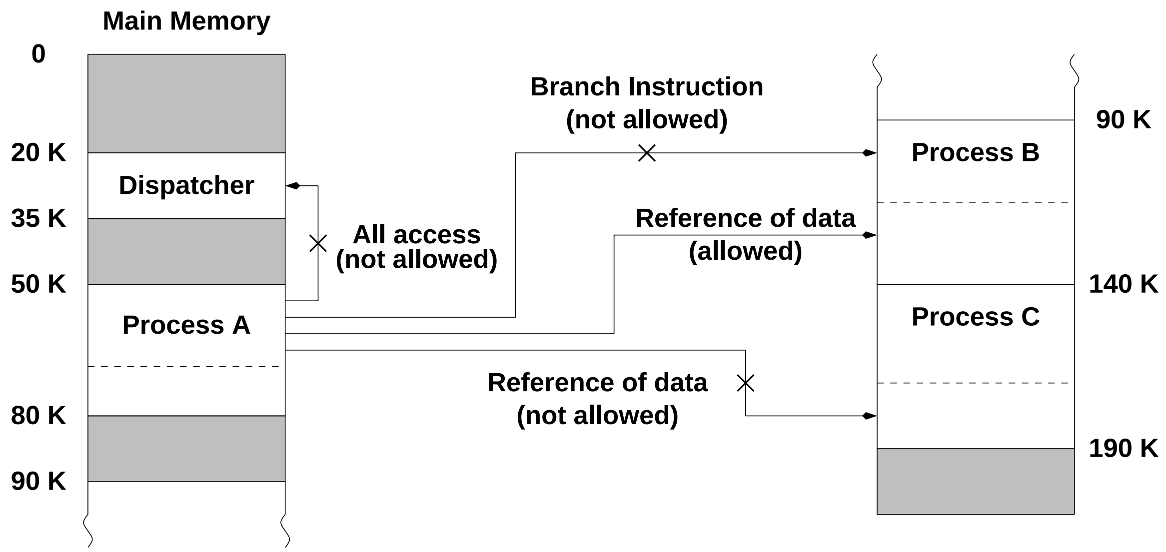
✱ Como na segmentação, cada entrada na tabela inclui o comprimento e o endereço base do segmento, não é possível a um programa acessar inadvertidamente uma posição da memória fora dos limites estabelecidos;

- ... para compartilhar um segmento, faz-se necessário referenciá-lo por mais de um processo em sua tabela de segmentos bem como na tabela de regiões por processo;

- ... na paginação, um mecanismo semelhante é possível, entretanto, neste caso a estrutura de página não é visível ao programador tornando a especificação de proteção e compartilhamento um tanto quanto desajeitada.

... 8.1.4 - Mecanismos de Proteção e Compartilhamento

✱ Um esquema mais sofisticado de proteção é a estrutura de proteção em anel que concede privilégio maior para os anéis mais internos e menor os externos.



8.2 - Sub-Sistema de Gerenciamento de Memória

✰ Quanto ao sub-sistema de gerenciamento de memória, os elementos de projeto (políticas) tem como aspecto chave a performance, ou seja, minimizar o número de faltas de páginas uma vez que causam considerável *overhead* de *software*.

- **Fetch Policy**
  - **Demand**
  - **Preparing**
- **Placement Policy**
- **Replacement Policy**
  - **Basic Algorithms**
    - **Optimal**
    - **Least Recently Used**
    - **First In, First Out**
    - **Clock**
  - **Page Buffering**
- **Resident Set Management**
  - **Resident Set Size**
    - **Fixed**
    - **Variable**
  - **Replacement Scope**
    - **Global**
    - **Local**
- **Cleaning Policy**
  - **Demand**
  - **Precleaning**
- **Load Control**
  - **Degree of Multiprogramming**

... 8.2 - Sub-Sistema de Gerenciamento de Memória

✰ No outro extremo, o sistema pode tomar a decisão que páginas devem permanecer e quais devem ser movidas para a memória secundária;

- ... na seqüência, o sistema operacional deve escalonar um outro processo durante a operação de entrada/saída da página, causando o chaveamento de processo.

✰ Adicionalmente, a performance de qualquer política é dependente:

- do tamanho da memória principal;
- da velocidade relativa entre memória principal e secundária;
- do número e tamanho de processos competindo por recursos;
- do comportamento de execução de cada um dos programas instanciados.

8.2.1 - Políticas de Busca de Página

✰ **Fetch Policy:** determina quando uma página deve ser trazida para a memória, podendo se dar por **demanda** ou por **pré-paginação**.

✰ **Paginação por Demanda:** página é trazida quando uma referência à mesma se fizer presente, ou seja, a busca está condicionada à referência de página;

- ... toda vez que um processo é instanciado tem-se um enxurrada de falta de página, mas a medida que mais páginas são referenciadas o número de faltas de páginas tende a cair por conta do princípio da localidade;

... 8.2.1 - Políticas de Busca de Página

✰ **Pré-Paginação:** página que não foram referenciadas por uma falta de página são trazidas para a memória principal por conta de outros aspectos;

- ... dispositivos de armazenamento secundário apresentam atrasos decorrentes de características intrínsecas aos mesmos, por isso, quando de uma falta de página um bloco do dispositivo de armazenamento é lido e trazido para a memória principal;

- ... esta política é ineficiente para a maioria das páginas extras que foram trazidas para a memória principal mas que não foram referenciadas.

8.2.2 - Políticas de Substituição de Página

✰ **Replacement Policy:** seleciona a página na memória principal que deve ser substituída pela página que causou a corrente falta de página.

✰ Vários são os aspectos a serem explorados e/ou considerados neste processo:

- quantos *frames* são alocados a cada processo?
- a que conjunto de *frames* deve estar limitado a política de substituição de página?
- dentro do conjunto selecionado, que página deve ser escolhida para a substituição?

✰ Um parâmetro que deve ser considerado é de que quanto mais elaborado e sofisticado é a política de substituição de página, maior será o *overhead* de *hardware* e/ou de *software* necessário para implementá-lo.

... 8.2.2 - Políticas de Substituição de Página

✰ **Troca da Página Ótima:** 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:** 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).

... 8.2.2 - Políticas de Substituição de Página

✰ **Página Não Recentemente Usada:** considera 02 bits, R e M para indiciar a operação de leitura e/ou escrita e de escrita na página;

- ... 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 distinguir 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

... 8.2.2 - Políticas de Substituição de Página

✰ O algoritmo *Not Recently Used* ou NRU remove uma página aleatória da classe não vazia de numeração mais baixa;

✰ 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.

✰ 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;

✰ Dentre as características principais do algoritmo destacam-se: é fácil de entender; eficiente de se implementar; e tem desempenho que, enquanto certamente não ótimo, é geralmente tido como adequado para o gerenciamento de memória.

... 8.2.2 - Políticas de Substituição de Página

✰ **Troca de Página FIFO:** 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.

... 8.2.2 - Políticas de Substituição de Página

✰ Uma variação do FIFO é o algoritmo *Segunda Chance* - que primeiro examinar a página mais velha como uma vítima potencial para ser substituída;

- ... 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;

✰ **FIFO Second 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.

... 8.2.2 - Políticas de Substituição de Página

✰ **Página Menos Recentemente Usada:** uma boa aproximação para o algoritmo da página ótima 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.

... 8.2.2 - Políticas de Substituição de Página

✰ 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.

... 8.2.2 - Políticas de Substituição de Página

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 *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.

... 8.2.2 - Políticas de Substituição de Página

✰ 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.

✰ **Solução:** usar um contador em *software* associado a cada página, que 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 constituem-se uma tentativa de guardar a freqüência 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.

... 8.2.2 - Políticas de Substituição de Página

✰ **Principal Problema:** o algoritmo nunca esquece referências anteriores, ou seja, páginas muito (e não mais) referenciadas 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*);

✰ Modificações no Algoritmo *Not Frequently Used*:

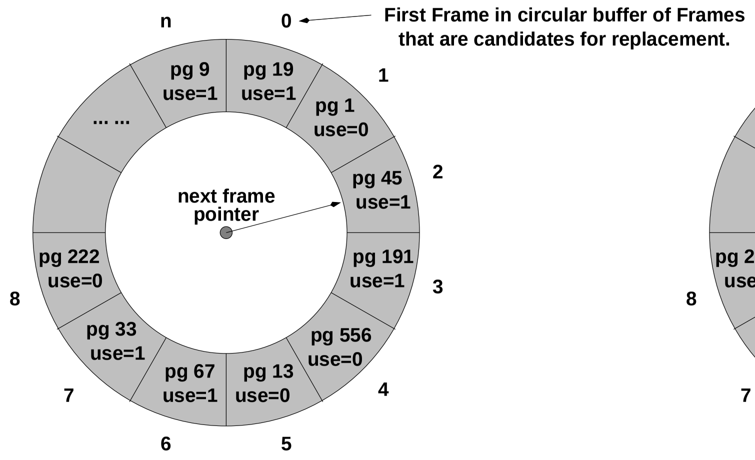
❶ os contadores são cada um deslocados 1 bit para a direita, ou seja, são divididos por 2 antes do bit R ser incrementado;
❷ o bit R é incrementado no bit mais a esquerda.

- ... 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*.

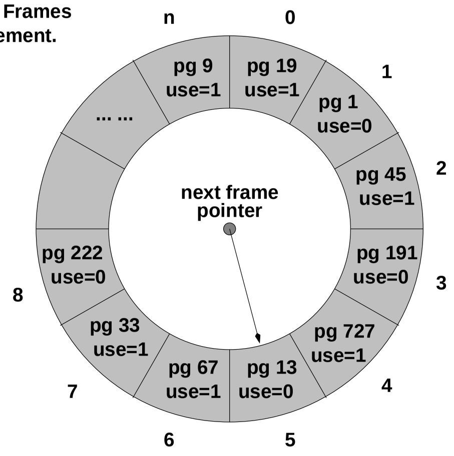
... 8.2.2 - Políticas de Substituição de Página

✍ **Clock:** na sua forma mais simples, requer a associação de um bit a cada *frame* para indicar quão recente foi a carga da página na memória principal;

- ... assim, ao ser referenciada pela primeira vez, atribui-se 1 a este bit, enquanto que nas próximas referências atribui-se 0 a este bit.



State of Buffer just prior to a Page Replacement



State of Buffer after the Next Page Replacement

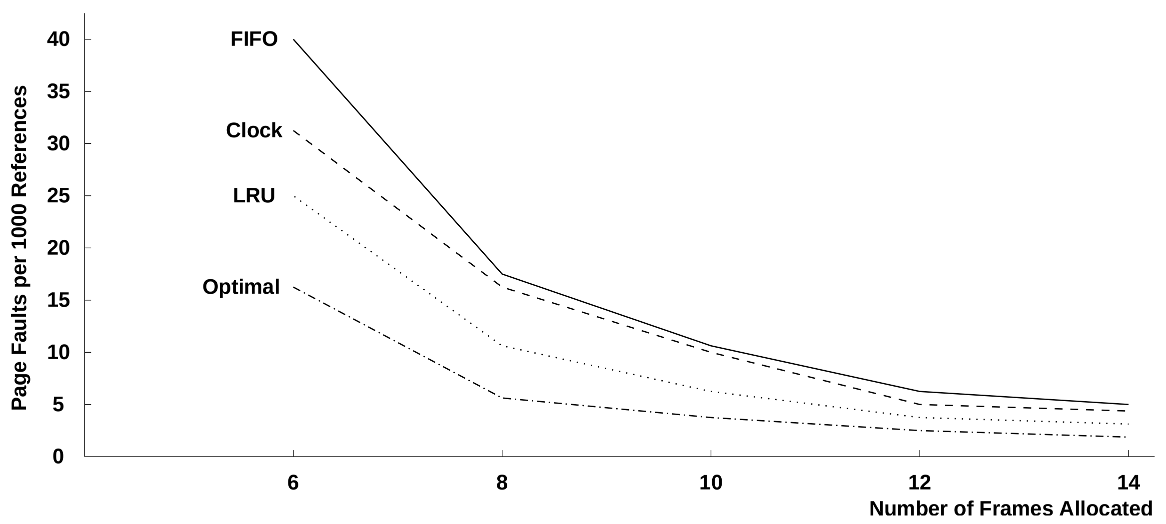
... 8.2.2 - Políticas de Substituição de Página

✰ Considere a execução de um processo que requer a referência de 05 páginas distintas:

Page Address Stream	2	3	2	1	5	2	4	5	3	2	5	2
Optimal	2	2	2	2	2	2	4	4	4	2	2	2
		3	3	3	3	3	3	3	3	3	3	3
				1	5	5	5	5	5	5	5	5
					F		F			F		
Least Recently Used	2	2	2	2	2	2	2	2	3	3	3	3
		3	3	3	5	5	5	5	5	5	5	5
				1	1	1	4	4	4	2	2	2
					F		F		F	F		
First In, First Out	2	2	2	2	5	5	5	5	3	3	3	3
		3	3	3	3	2	2	2	2	2	5	5
				1	1	1	4	4	4	4	4	2
					F	F	F		F		F	F
Clock	2*	2*	2*	2*	5*	5*	5*	5*	3*	3*	3*	3*
		3*	3*	3*	3	2*	2*	2*	2	2*	2	2*
				1*	1	1	4*	4*	4	4	5*	5*
					F	F	F		F		F	

... 8.2.2 - Políticas de Substituição de Página

✰ Comparação de alocação fixa para o Algoritmo LRU:



8.2.3 - Políticas de Manutenção de Página

✡ Sistema Operacional pode decidir quantas páginas irá manter em memória principal para cada processo, embora vários aspectos devam ser considerados:

- quanto menor for a quantidade de memória alocada a um processo, mais processos poderão residir de uma só vez na memória principal;
- se o número relativo de páginas de um processo mantidas em'q memória for baixo, a taxa com que ocorreram faltas de páginas será alta (princípio da localidade);
- depois de um dado tamanho, alocações adicionais de memória para um processo não irão diminuir a taxa de faltas de páginas (princípio da localidade).

✡ Considerando tais fatores, 02 políticas são encontradas nos SOs Contemporâneos:

- **alocação fixa:** cada processo recebe, quando instanciado, um número fixo de páginas tendo por base o tipo de processo ou parâmetros fornecidos pelo programador ou pelo gerenciador do sistema;
- **alocação variável:** permite que o número de páginas alocadas para um processo varie enquanto está sendo executado pelo processador.

... 8.2.3 - Políticas de Manutenção de Página

✡ Quanto ao escopo em que a estratégia de substituição é aplicada, 02 políticas podem ser adotadas: **local replacement policy** e **global replacement policy**;

- ... ambas são ativadas quando de um falta de página mas diferem na seleção da página a ser substituída do conjunto de páginas residentes.

	Local Replacement	Global Replacement
Fixed Allocation	Number of frames allocated to process is fixed. Page to be replaced is chosen from among the frames allocated to that process.	Not prossible.
Variable Allocation	The number of frames allocated to a process may be changed from time to time, to maintain the working set of the process. Page to be replaced is chosen from among the frames allocated to that process.	Page to be replaced is chosen from all available frames in main memory; this causes the size of the resident set of processes to vary.

8.2.4 - Políticas de Retirada de Página

✡ **Cleaning Policy:** define quando uma página modificada e residente em memória principal deve ser copiada na memória secundária.

✡ Duas são as abordagens para a Política de Retirada de Páginas:

- **demand cleaning:** a página terá seu conteúdo salva-guardado em memória secundária somente depois de ter sido escolhida para ser substituída;

- **precleaning:** salva-guarda as páginas modificadas antes que os *frames* correspondentes sejam requisitados, assim, podem ser escritas em lotes.

✡ Há vários perigos na adoção estrita de uma única abordagem, por exemplo, com a pré-retirada a página permanece na memória até ser marcada para ser substituída;

- ... a capacidade de transferência da memória secundária é limitada e não deve ser desperdiçada com inúmeras operações de *cleaning*.

... 8.2.4 - Políticas de Retirada de Página

✡ Uma abordagem alternativa incorpora a bufferização, possibilitando que apenas páginas marcadas como substituíveis sejam retiradas, ou seja, 02 listas são mantidas: páginas modificadas e não modificadas;

- ... com a bufferização as operações de *cleaning* e de substituição ficam desaclopadas.

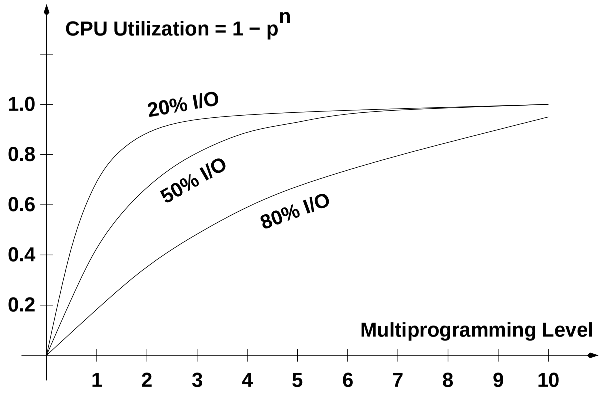
✡ Páginas na lista modificada são periodicamente escritas na memória secundária em lotes e movidas para a lista de páginas não modificadas;

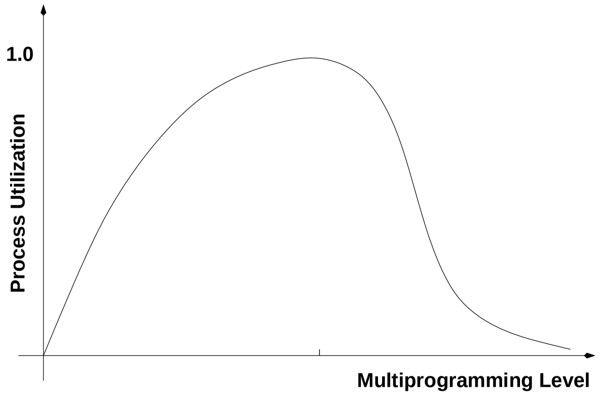
- ... uma página na lista de não modificadas poderá ser reativada se for novamente referenciada ou perdida caso em que o *frame* é atribuído a outra página.

8.2.5 - Políticas do Grau da Multiprogramação

✰ **Load Control:** determina o número de processos que estarão residentes em memória principal, ou seja, determina o grau da multiprogramação;

- ... esta política é crítica no gerenciamento efetivo da memória, posto que se alguns processos estão residentes a qualquer tempo, iremos encontrar várias ocasiões em que todos estarão bloqueados e muito tempo será gasto com a sua permutação.





... 8.2.5 - Políticas do Grau da Multiprogramação

✰ Se o grau da multiprogramação deve ser reduzido, um ou mais processos residentes em memória principal deverão ser suspensos (*swapped out*).

✰ Entretanto, outras possibilidades incluem a suspensão de:

- processos com baixa prioridade através de política de escalonamento;
- processos com o menor conjunto de páginas residentes, pois penaliza programa cujo princípio da localidade está restrito a poucas páginas;
- processos grandes, pois liberam muitos *frames* na memória alocada tornando improváveis desativações adicionais.

8.3 - Gerenciamento de Memória no UNIX

- ✰ Implementações correntes utilizam-se do conceito de Memória Virtual, entretanto, as primeiras versões do UNIX utilizaram a abordagem de partições variáveis;

- … para as versões SVR4 e Solaris, 02 esquemas de gerenciamento de memória são utilizados: **paging system** e **kernel memory allocator**.

- ✰ **Paging System:** acomoda funcionalidades que possibilitam a alocação de páginas para processos na memória principal bem como a alocação de páginas em *buffers* da memória secundária, p.ex.: disco rígido.

- ✰ **Kernel Memory Allocator:** mecanismo personalizado de gerenciamento de alocação de memória do *kernel* do sistema operacional.

… 8.3 - Gerenciamento de Memória no UNIX

✰

8.4 - Gerenciamento de Memória no Windows NT

... 8.4 - Gerenciamento de Memória no Windows NT