

---

# Sistemas Operativos

2021 – 2022

---

**Gestão de memória**  
**Endereçamento virtual**  
**Memória virtual**

DEIS/ISEC

Sistemas Operativos – 2021/22

1

João Durães

---

## Tópicos

Conceitos fundamentais de gestão de memória;  
Endereçamento real vs. endereçamento virtual;  
Memória segmentada; memória paginada;  
Memória virtual;  
Algoritmos de atribuição, de transferência e de substituição;  
Modelo de Espaço de Trabalho.

---

### Bibliografia específica:

- *Fundamentos de Sistemas Operativos*; 3ª Ed.; Marques & Guedes
- Capítulos 4 e 5
- *Operating Systems Concepts*; Silberschatz & Galvin  
Capítulos 8 e 9
- *Operating Systems - Internals and Design Principles*; William Stallings  
Capítulos 7 e 8

DEIS/ISEC

Sistemas Operativos – 2021/22

2

João Durães

## Gestão de memória

### Mecanismos de gestão de memória

- Proporcionado processador/Hardware
- Determinam a organização de memória do computador
  - Qual o modo de endereçamento: real ou virtual
  - Qual o tamanho dos blocos (páginas ou segmentos)
  - Quais os mecanismos de protecção e validação de endereços

Cabe ao S.O. configurar e aproveitar adequadamente os mecanismos de gestão proporcionados pelo hardware

## Gestão de memória

### Algoritmos de gestão de memória

- Técnicas para a gestão de memória principal (física) e secundária (ex. implementação de políticas de atribuição de blocos)
  - Os detalhes da gestão de memória efectuada pelo S.O. devem ser ocultos aos diversos processos em execução
  - Os algoritmos de gestão de memória têm forte impacto no desempenho do sistema

É da responsabilidade do S.O. gerir de forma eficiente a memória, utilizando para isso os mecanismos proporcionados pelo hardware

## Gestão de memória

### Endereçamento real (memória real)

- Os endereços gerados e utilizados pelos programas em execução correspondem directamente a posições na memória física existente no computador
  - A dimensão dos programas fica limitada pela dimensão da memória física o computador
  - Normalmente um programa é preparado (pelo compilador) para correr numa determinada gama de endereços
  - O facto anterior dificulta a multi-programação pois quase de certeza que irão existir diversos programas a querer utilizar a mesma gama de endereços

## Gestão de memória – Sistemas de endereçamento real

### Ponto de vista de hardware:

→ Mais simples de implementar (circuitos mais simples)

### Ponto de vista do S.O.:

→ Apresenta dificuldades e limitações importantes



- Fragmentação decorrente da atribuição/libertação de memória
- Código não recolocável
- Limites quanto ao tamanho máximo dos programas
- Dificuldades na concretização de isolamento e segurança

Estes problemas têm solução parcial mas implicam uma perda de eficiência no desempenho da máquina, e complexidade acrescida para o SO e para os programadores das aplicações utilizador.



## Gestão de memória

### Endereçamento virtual

- Cada processo apercebe-se um conjunto de endereços (espaço de endereçamento) **sem relação directa** com qualquer porção de memória real
  - O processador traduz automaticamente esses endereços para endereços físicos.
  - A tradução é feita de forma transparente

### Endereços virtuais

- São os endereços utilizados pelos processos em execução e que não correspondem directamente à memória física

### Endereços reais (endereços em memória física)

- Referem posições da memória física (memória que existe de facto)

## Gestão de memória – Sistemas de endereçamento virtual

### Endereçamento virtual

O espaço de endereçamento de cada processo é subdividido num **conjunto de blocos**.

- A tradução **end. virtual** → **end. real** é feito **bloco a bloco**
- Todos os endereços virtuais **do mesmo bloco** têm as **mesmas regras** de tradução para endereços reais
- Dois blocos distintos podem ter regras distintas e tradução para endereçamento real
- Cada bloco pode estar mapeado em qualquer zona da memória física: não é preciso manter a mesma ordem que os blocos aparentam ter no espaço de endereçamento virtual
- Usando memória virtual, pode-se armazenar temporariamente em disco alguns dos blocos do espaço de endereçamento que não façam falta de momento ( → “**Memória virtual**”)

## Gestão de memória

---

### ➤ *memória virtual:*

- O espaço de endereçamento virtual é bastante maior que o total da memória física realmente existente.
- Algumas áreas do espaço de endereçamento encontram-se efectivamente mapeadas (algures) em memória física.
- As áreas dos espaços de endereçamento que não estão mapeadas em memória física encontram-se temporariamente armazenadas em memória secundária (disco).
- O S.O. gere a transferência das diversas áreas entre memória física e memória secundária (de uma forma transparente para os processos)

## Gestão de memória

---

### Endereçamento virtual e memória virtual

- É da responsabilidade do S.O. gerir os vários espaços de endereçamento virtuais:
  - Trazer para memória física os blocos que estão a ser necessários
  - Libertar espaço em memória física, transferindo para memória secundária blocos que não estejam a ser usados
  - Indicar ao processador de que forma é efectuada a tradução de endereços virtuais para endereços reais
- Esta gestão é efectuada de uma forma transparente para os processos em execução. Cada processo vê a memória como estando toda disponível para si.

## Gestão de memória – Sistemas de endereçamento virtual

### Memória segmentada

- Bloco = “segmento”
- Segmentos diferentes podem ter tamanhos diferentes
- A tradução de endereços virtuais em endereços reais é feita pelo próprio processador com recurso a uma ou mais tabelas, chamadas **tabelas de segmentos**, as quais são criadas e preenchidas pelo S.O.
- Cada entrada numa tabela de segmentos chama-se **descritor de segmento**, e descreve o segmento quanto a:
  - Tamanho e localização em memória física
  - Tipo de acesso permitido (protecção: leitura, escrita, execução)
  - Estado de presença em memória física (bit “P”)
- Registos específicos no processador indicam a base da tabela de segmentos e a dimensão da tabela de segmentos em uso.

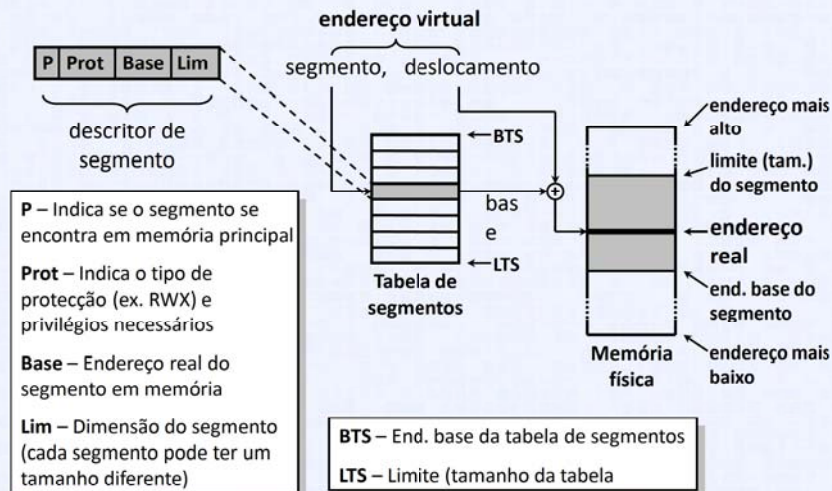
## Gestão de memória – Sistemas de endereçamento virtual

### Memória segmentada

- Um endereço virtual é composto por um conjunto de bits que indicam qual o descritor de segmento dentro da tabela de segmentos que vai ser utilizado (abrev. “qual o segmento”) e um conjunto de bits que indica qual o deslocamento dentro desse segmento
- Para otimizar o custo do acesso à tabela de segmentos durante a tradução de endereços, utiliza-se um conjunto de registos de segmento que contêm os valores dos descritores de segmento em uso. Se o programa não estiver sempre a mudar de segmento, evita-se grande parte dos acessos à tabela de segmentos.



## Memória segmentada – Tradução dos endereços



DEIS/ISEC

Sistemas Operativos – 2021/22

13

João Durães

## Gestão de memória – Sistemas de endereçamento virtual

### Memória paginada

Análoga à memória segmentada com a diferença que todos os blocos, chamados de páginas, têm o mesmo tamanho (geralmente pequenas)

- A tradução de endereços é feita de maneira análoga à da memória segmentada: em vez de tabelas de segmentos utilizam-se tabelas de páginas; cada entrada nessas tabelas denomina-se de PTE (*Page Table Entry*). Cada PTE descreve uma página quanto a:
  - Localização em memória física (por outras palavras, qual a *page-frame* em memória física onde está mapeada)
  - Tipo de acesso permitido (protecção)
  - Estado de presença em memória física (bit “P”)
  - Utilização da página quanto a referência (bit “R”) e modificação (bit “M”)
- Os registos BTP e LTP são os correspondentes aos BTS e LTS da memória segmentada

DEIS/ISEC

Sistemas Operativos – 2021/22

14

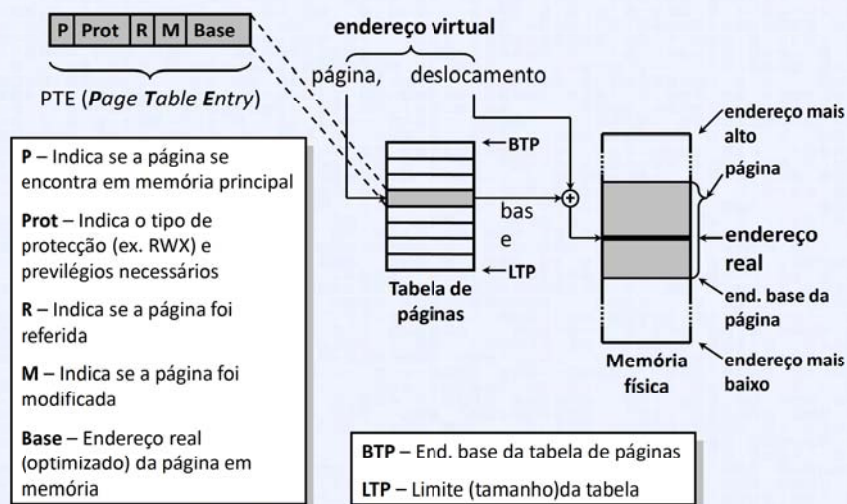
João Durães

## Gestão de memória – Sistemas de endereçamento virtual

### Memória paginada (cont.)

- O facto das páginas terem todas o mesmo tamanho tem como consequências:
  - É possível a criação de um espaço de endereçamento perfeitamente linear
  - É possível otimizar ou simplificar os algoritmos de gestão (a ver mais adiante)
- Para otimizar o custo do acesso à tabela de páginas durante a tradução de endereços, utiliza-se uma *cache* que consiste numa tabela de memória associativa muito rápida para servir de *cache* às PTE recentemente utilizadas (que serão provavelmente, pelo princípio da localidade de referência, as próximas a serem necessárias).  
Esta tabela denomina-se de TLB (*Translation Lookaside Buffer*)

## Memória paginada – Tradução dos endereços





### Gestão de memória – Memória paginada multi-nível

- Se o espaço de endereçamento for muito grande a memória paginada torna-se ineficiente

#### Exemplo

- Endereços de 64 bits
- Tamanho de página: 4k (dimensão típica): 12 bits
- Restante:  $64 - 12 = 52$  bits

→ Tabela de páginas teria (até)  $2^{52}$  entradas !



Não é viável manter a tabela em memória

#### Solução:

- Utilizar vários níveis de tabelas de páginas (em cascata)

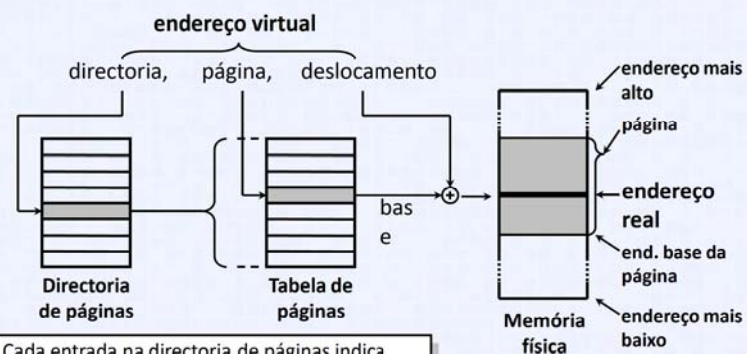
DEIS/ISEC

Sistemas Operativos – 2021/22

17

João Durães

### Memória paginada multi-nível



Cada entrada na directoria de páginas indica uma tabela de páginas. A partir deste ponto a tradução é como na situação anterior.

Pode existir mais que um nível intermediário

Pode-se associar uma entrada na directoria de páginas a cada processo

Exemplo - Processadores Intel:  
Memória paginada de nível duplo

Directoria – 10 bits  
Página – 10 bits  
Deslocamento – 12 bits

DEIS/ISEC

Sistemas Operativos – 2021/22

18

João Durães

## Gestão de memória – Sistemas de endereçamento virtual

### Memória segmentada

#### Vantagens

- Adequa-se bastante à divisão lógica dos programas: um segmento para código, outro para dados, etc. Não é obrigatório que só haja um segmento de cada tipo
- Permite trabalhar eficientemente sobre uma zona de memória inteira (um segmento)

#### Desvantagens

- Pode não permitir criar um espaço de endereçamento perfeitamente linear e contínuo de uma forma transparente para o processo
- Os algoritmos de gestão são mais complicados; o mapeamento de segmentos em memória pode gerar fragmentação (porque cada segmento pode ter um tamanho diferente dos outros)
- As transferências entre memória principal e secundária é sempre feita em blocos (segmentos) inteiros. No caso dos segmentos serem grandes, têm-se grandes custos de eficiência

DEIS/ISEC

Sistemas Operativos – 2021/22

19

João Durães

## Gestão de memória – Sistemas de endereçamento virtual

### Memória paginada

#### Vantagens

- Totalmente transparente para o programador e processo
- Os algoritmos de gestão são mais simples e mais eficientes pelo facto de todas as páginas terem o mesmo tamanho; as transferências entre memória principal e memória secundária são feitas por páginas individuais (tamanho típico: 512 bytes a 8K), sendo muito rápidas
- Só há fragmentação (interna) na última página do processo

#### Desvantagens

- Não se adequa à divisão lógica dos programas tão bem como a memória segmentada pois cada parte lógica dos programas (código, dados, etc.) irão precisar de mais do que uma página
- As faltas de páginas representam uma perda de eficiência maior do que as faltas de segmento (por serem potencialmente em maior número)

DEIS/ISEC

Sistemas Operativos – 2021/22

20

João Durães

## Gestão de memória – Sistemas de endereçamento virtual

### Memória segmentada-paginada

Mecanismo com semelhanças à memória paginada multi-nível e junta as melhores características da memória segmentada e da memória paginada

- Um endereço virtual é composto por um segmento, uma página e o deslocamento dentro da página; o segmento indica qual o descritor dentro da tabela de segmentos que deve ser utilizado na tradução do endereço; o descritor indicado contém informação acerca de qual a tabela de páginas a ser utilizada para o restante da tradução do endereço (esse restante é feito como no caso da memória paginada)
- Consegue-se ter, desta forma, um segmento subdividido em N páginas, mantendo-se as vantagens da adequação dos segmentos à estrutura lógica dos programas e a eficiência associada às páginas
- Continuam a ser utilizados ambos os mecanismos de optimização de eficiência: registos de segmentos e TLB

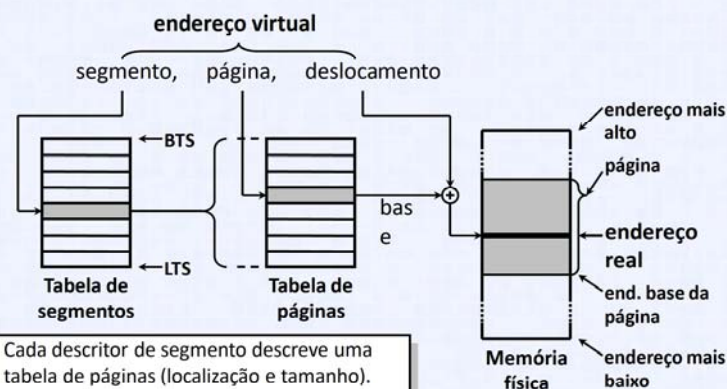
DEIS/ISEC

Sistemas Operativos – 2021/22

21

João Durães

### Memória segmentada-paginada – Tradução dos endereços



Cada descritor de segmento descreve uma tabela de páginas (localização e tamanho).  
A utilização do valor do segmento e página, assim como as respectivas tabelas, é análoga à dos casos da memória segmentada pura e da memória paginada pura.

DEIS/ISEC

Sistemas Operativos – 2021/22

22

João Durães



### **Gestão de memória – Sistemas de endereçamento virtual**

Vantagens da utilização de memória virtual (através de memória segmentada, paginada ou suas variantes)

- Facilidade na implementação de sistemas multi-programados: cada processo tem o seu próprio espaço de endereçamento, sendo evitada a necessidade de código recolocável; os algoritmos de gestão de memória física podem recolocar os blocos de memória em localizações diferentes sem que isso afecte os processos em execução
- Implementação automática de segurança e isolamento dos espaços de endereçamento de processos diferentes: cada processo terá a(s) sua(s) própria(s) tabela(s) de segmentos ou páginas
- A partilha de zonas de memória entre processos distintos é simples: dois (ou mais) descritores de segmento (ou PTE's) descreverão o mesmo bloco em memória física (eventualmente com protecções diferentes, se isso for desejável)

### **Gestão de memória – Políticas de transferência**

**Matéria seguinte:**

- Algoritmos de atribuição
- Algoritmos de transferência
- Algoritmos de substituição

## Gestão de memória

### Algoritmos de gestão de memória

- **Algoritmos de atribuição (alocação)**
  - Determinam onde (em memória física) fica mapeado um bloco de memória virtual
- **Algoritmos de transferência**
  - Determinam a política quanto às transferências entre memória principal e memória secundária
- **Algoritmos de substituição**
  - Determinam qual o bloco que deve ser retirado de memória física para dar lugar a outro

## Gestão de memória - Algoritmos

### Algoritmos de atribuição

#### Em memória paginada

- Uma vez que todas as páginas têm o mesmo tamanho, a decisão de qual a **page-frame** (página de memória física) que fica atribuída a uma página de memória virtual resume-se a encontrar uma que esteja livre.
- Para tal apenas é preciso manter uma estrutura de dados que indique quais as páginas ocupadas e quais as páginas livres.
- A primeira página livre encontrada é a escolhida.

## Gestão de memória – Algoritmos de alocação

### Algoritmos de atribuição

#### Em memória segmentada

- Cada segmento pode ter um tamanho diferente.
- Pode surgir o problema de fragmentação se não se escolher bem qual o bloco a atribuir.

⇒ A decisão deixa de ser trivial como no caso da memória paginada

Existem vários algoritmos

- *Best-Fit*
- *Worst-Fit*
- *First-Fit*
- *Next-Fit*
- *Buddy (binário)*

## Gestão de memória – Algoritmos de alocação

### Algoritmo *Best-Fit*

- Procura o bloco que melhor se adapte à dimensão pedida (que será o menor bloco que ainda consiga conter aquilo que foi pedido)
- Este algoritmo visa otimizar a ocupação da memória. No entanto, geralmente o bloco atribuído excede em pouco a dimensão pedida. O excesso vai formar um novo bloco, geralmente demasiado pequeno para servir para alguma coisa.
- A estrutura de dados associada consiste numa lista de blocos, ordenada por ordem crescente. Em média será preciso percorrer metade da lista até encontrar o bloco desejado. O bloco resultante do excedente tem que ser inserido na posição adequada na lista.



## Gestão de memória – Algoritmos de alocação

### Algoritmo *Worst-Fit*

- Procura o maior bloco, sendo esse o bloco atribuído.
- Este algoritmo tenta prevenir o aparecimento de blocos pequenos como no caso do *best-fit*. A ideia é a seguinte: o excedente do bloco (a diferença entre o que foi atribuído e o que foi pedido) ainda será suficientemente grande para ser útil a pedidos posteriores. No entanto os blocos grandes desaparecem rapidamente tendo como consequência a possibilidade de pedidos grandes posteriores não poderem ser satisfeitos.
- A estrutura de dados associada será uma lista de blocos ordenada por ordem decrescente. A determinação de qual é o maior bloco fica bastante facilitada. O bloco resultante do excedente tem que ser inserido na posição adequada na lista.

## Gestão de memória – Algoritmos de alocação

### Algoritmo *First-Fit*

- Escolhe para atribuição o primeiro bloco que é suficientemente grande para satisfazer o pedido.
- Este algoritmo tem como objectivo diminuir o tempo de pesquisa do bloco a atribuir sem ter a desvantagem do *worst-fit*.
- É gerada fragmentação num dos extremos de memória enquanto se salvaguarda a existência de blocos grandes livres no outro extremo.
- A estrutura de dados associada consiste numa lista de blocos ordenada pelo endereço do bloco. Os blocos resultantes do excedente entre o bloco atribuído e o bloco pedido são mais facilmente reintroduzidos que nos algoritmos anteriores.

## Gestão de memória – Algoritmos de alocação

### Algoritmo *Next-Fit*

- Versão modificada do *first-fit*, em que cada pesquisa começa no ponto onde a anterior terminou.
- Esta modificação visa evitar o aparecimento de fragmentação apenas num dos extremos da lista, melhorando a velocidade de pesquisa, tendo no entanto como consequência a dispersão de blocos pequenos por toda a memória.

## Gestão de memória – Algoritmos de alocação

### Algoritmo *Buddy-Binário*

- Este algoritmo atribui blocos de tamanho  $2^n$  de tal forma que a dimensão pedida  $TAM$  verifica  $2^{n-1} < TAM \leq 2^n$ . Se não existir nenhum bloco livre nessa condição, tenta-se encontrar um bloco maior (sempre com uma dimensão potência de 2), o qual será dividido em dois de uma forma recursiva até se obter um bloco que verifique a condição. Os outros blocos resultantes deste processo de divisão ficam livres para outros pedidos.
- Este algoritmo tem a vantagem de conseguir um bom equilíbrio entre a velocidade de pesquisa e a fragmentação: A quantidade de blocos manipulados nunca é demasiado grande; por outro lado os blocos libertos vão sendo recombinados, reconstruindo-se facilmente os blocos maiores originais.

## Gestão de memória – Políticas de transferência

### Políticas de transferência

- Três situações
  - **Por pedido**  
O S.O. (ou um processo) determina quando se deve carregar um bloco em memória (ex.: na criação de um processo)
  - **Por necessidade**  
O bloco de memória é acedido numa altura em que não se encontrava em memória principal; dá-se uma falta de página ou segmento e torna-se necessário carregar o bloco em memória física
  - **Por antecipação**  
Para tentar aumentar o desempenho, o bloco é carregado em memória quando há fortes probabilidades de ele vir a ser necessário nos próximos instantes (mas antes de ser efectivamente necessário)

## Gestão de memória – Políticas de transferência

### Transferência de páginas

- As transferências de páginas são geralmente feitas por necessidade:
  - Quando um processo é criado, as suas tabelas de páginas são inicializadas mas as páginas não são colocadas em memória;
  - À medida que o processo vai progredindo, irá gerar endereços que irão causar faltas de páginas - nessas ocasiões, o S.O. colocará as páginas em memória de uma forma transparente para o processo (porque estão a ser necessárias);
  - Também é habitual usar a política de antecipação para optimização dos acessos a memória secundária
  - As páginas que nunca chegam a ser necessárias (código que não chegou a ser executado ou dados que não chegaram a ser necessários, por exemplo) nunca serão carregadas, tendo-se poupado tempo.



## Gestão de memória – Políticas de transferência

### Transferência de segmentos

Normalmente, um processo precisa de vários segmentos em simultâneo para se poder executar: código, dados, pilha, *heap*, etc. Em processos pequenos pode coincidir com a totalidade da sua memória. Quando o sistema coloca o processo em execução, coloca em memória os seus segmentos.

- As transferências de segmentos são, geralmente, feitas por **pedido**
- Se o processador suportar excepções do tipo falta de segmento, então também se podem efectuar transferências de segmento por necessidade. No entanto, mesmo que o processador suporte este tipo de excepções, raramente os SOs as utilizam pois uma falta de segmento tem um *overhead* muito maior que uma falta de página

## Gestão de memória – Algoritmos de Substituição

### Substituição de páginas

- O factor decisivo quanto à escolha das páginas que são retiradas de memória para dar lugar a outras está relacionado com a sua utilização.
- Idealmente, as páginas a retirar serão aquelas que apenas serão necessárias em último lugar. Este algoritmo não é directamente concretizável pois depende de conhecimento acerca do futuro.
- Através do **princípio de localidade de referência**, pode-se utilizar o conhecimento da utilização passada das páginas para inferir qual irá ser a sua utilização futura.
- Os seguintes algoritmos consistem na tentativa de aproximação ao algoritmo ideal:
  - FIFO *First In First Out*
  - LRU *Least Recently Used*
  - NRU *Not Recently Used*

### Gestão de memória – Substituição de páginas – FIFO

#### Algoritmo **FIFO** – *First In First Out*

- Este algoritmo mantém as páginas numa lista ordenada pelo tempo em que foram carregadas em memória.
- Quando se carrega uma página em memória, coloca-se no fim da lista.
- Quando é necessário remover uma página, escolhe-se a que está no início da lista.

Este algoritmo tem a desvantagem de não conseguir distinguir uma página que já não é necessária (foi carregada à muito tempo) de uma que está a ser sempre necessária (de tal forma que foi das primeiras a serem carregadas).

⇒ Pode acontecer retirar-se uma página que vai voltar a ser necessária daí a pouco tempo.

### Gestão de memória – Substituição de páginas – LRU

#### Algoritmo **LRU** – *Least Recently Used* (menos usada recentemente)

- Este algoritmo baseia-se no tempo em que a página foi acedida em vez de o tempo em que a página foi carregada. Desta forma consegue-se distinguir páginas que estão sempre a ser acedidas e que foram carregadas há muito tempo das que foram carregadas há muito tempo mas que já não são utilizadas.
- Idealmente, o próprio processador manteria a idade da página desde o último acesso nas PTE. No entanto, tal solução seria demasiado custosa em termos de complexidade de *hardware*, pelo que não é vulgar.  
⇒ A contabilização do tempo que passou desde o ultimo acesso a cada página será feita pelo S.O.

### Gestão de memória – Substituição de páginas – LRU

#### Algoritmo LRU (cont.)

- A cada página é associado um contador, inicializado a zero quando a página é carregada.
- Sempre que uma página é acedida, o bit “R” da PTE através do qual foi acedida é colocado a “1”. Normalmente, esta actualização é feita pelo próprio processador (caso o processador não suporte esta característica, ainda é possível ao S.O. conseguir efectuar esta actualização mas com grande penalização em termos de eficiência).
- Periodicamente todas as tabelas de páginas são inspeccionadas; se o bit “R” estiver a “0”, o contador associado será incrementado (a página está a ficar “velha” por não ser utilizada); se o bit “R” estiver a “1”, o contador e esse bit são recolocados a zero (a página voltou a ser nova porque está a ser utilizada).
- Quando o contador atinge o valor máximo, a página associada é marcada como inválida (o bit “P” é posto a zero) para todos os efeitos, o espaço ocupado por ela em memória física está livre.

### Gestão de memória – Substituição de páginas – NRU

#### Algoritmo NRU – *Not Recently Used* (não usada recentemente)

- Simplificação do algoritmo LRU e que consiste analisar os bits “R” e “M” das PTE para classificar as páginas quanto à sua utilização.
- Interessa averiguar apenas se uma página foi ou não acedida ou modificada recentemente (não interessando distinguir o tempo decorrido); as páginas serão classificadas consoante as 4 combinações possíveis de valores para estes dois bits.
  - Periodicamente as tabelas de páginas são percorridas:
    - O bit “R” é posto a zero pelo processo paginador
    - O bit “M” é posto a zero quando a página é gravada em disco

#### Convém lembrar que:

- O bit “R” é colocado a “1” sempre que a página é acedida;
- O bit “M” é colocado a “1” sempre que a página é modificada;
- Normalmente, estas duas actualizações são feitas automaticamente pelo próprio processador quando traduz os endereços virtuais.



## Gestão de memória – Substituição de páginas – NRU

Algoritmo **NRU** (cont.)

- As páginas são escolhidas como vítimas para saírem da memória pela seguinte ordem

ordem	R	M	
1º	0	0	Nem referida nem modificada
2º	0	1	Não referida mas modificada
3º	1	0	Referida mas não modificada
4º	1	1	Referida e modificada (pior caso)

**Referida**  $\Rightarrow$  está a ser necessária (utilizada recentemente) – provavelmente será necessária brevemente (pelo princípio da localidade de referência)

**Modificada**  $\Rightarrow$  Tem que se gravar em memória secundária para não se perderem as modificações – tem o custo adicional de I/O

## Gestão de memória – Algoritmos de Substituição

### Substituição de segmentos

Como cada segmento tem o seu próprio tamanho, a escolha do(s) segmentos(s) a rer(em) retirado(s) de memória já não pode ser feito da mesma forma que em memória paginada

$\Rightarrow$  Normalmente retiram-se segmentos em função do processo a que pertencem

Interessa ver qual o processo que não precisa (não se prevê que vá) executar nos próximos instantes; tal processo é uma boa “vítima” para ter os seus segmentos retirados de memória principal

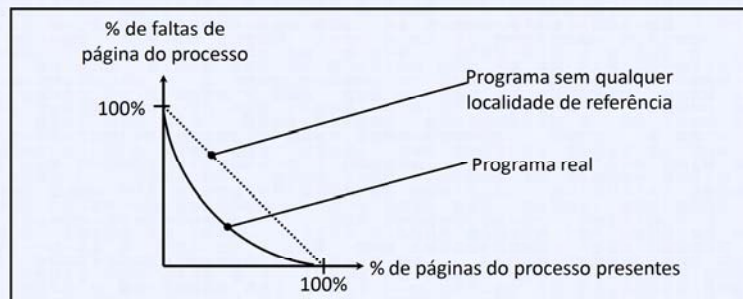
Factores a levar em conta em relação ao processo  
(cujos segmentos vão ser retirados)

- Qual a sua importância / prioridade
- Qual a sua utilização do processador
- Qual a sua dimensão (total dos seus segmentos)

## Gestão de memória – Substituição de páginas

### Modelo de espaço de trabalho

- Observa-se que compensa nunca colocar em memória mais do que uma determinada percentagem das suas páginas:
  - Não aumenta muito o número de faltas de página
  - Ficam mais páginas livres para outros processos



Variação das faltas de páginas em função da percentagem de páginas presentes

## Gestão de memória – Substituição de páginas

### Modelo de espaço de trabalho (cont.)

“Espaço de trabalho” : quantidade de páginas que um processo deve ter presentes em memória

- A dimensão do espaço de trabalho não é necessariamente a mesma para todos os processos.
- Quando um processo é criado, são carregadas algumas páginas (por antecipação) até se atingir um espaço de trabalho mínimo.
- À medida que o processo vai pedindo mais páginas, sem exceder o seu espaço de trabalho máximo, as novas páginas são obtidas do espaço atribuído a outros processos.
- Quando o processo atinge o valor máximo do seu espaço de trabalho, as novas páginas que lhe são atribuídas são obtidas do seu próprio espaço.
- Quando a quantidade de páginas de um processo presentes em memória desce abaixo do seu valor mínimo de espaço de trabalho (por lhe terem sido tiradas por outros processos), todo o processo é enviado para memória secundária.

## Aferição de conhecimentos neste capítulo

### Perguntas que se podem fazer sobre esta matéria (exemplos não exaustivos\*)

#### Compreensão/descrição directa

- Quais as desvantagens do endereçamento real face ao endereçamento virtual
- Descreva o mecanismo de tradução de endereço de memória *<um dos que foi dado>*
- Elabore uma análise comparativa de memória segmentada e memória paginada salientando as vantagens e desvantagens de um face ao outro.
- Descreva o conceito de espaço de trabalho e diga como pode ser usado para aumentar o desempenho do computador

#### Compreensão com um pouco de raciocínio

- Porque razão o mecanismo de excepções é fundamental para a construção de memória virtual?
- Como é que se constrói o isolamento entre processos recorrendo à memória segmentada? E em memória paginada, há alguma diferença significativa?
- Quais as vantagens da memória segmentada-paginada face à memória segmentada pura e memória paginada pura?

(\*) Convém sobrar uma ou duas para os exames (mas isto não quer dizer que saiam estas)

## Aferição de conhecimentos neste capítulo

### Perguntas que se podem fazer sobre esta matéria (cont.)

#### Aplicação de conhecimentos – exemplo de perguntas (dados concretos em aberto)

- Dado este conjunto de páginas e os tempos em que foram acedidos e modificados, diga qual será a primeira a sair de acordo com o algoritmo FIFO/LRU/NRU e porquê
- Dada esta tabela de páginas e estas instruções a serem executadas no espaço de endereçamento que lhe corresponde, diga quais irão ser os endereços reais manipulados e se existirá algum comportamento específico por parte do SO (se sim, qual e porquê)
- Dado este processo (código) diga qual acha que será a dimensão (%) do seu espaço de trabalho.
- Dado este cenário de ocupação de memória real em memória segmentada, diga onde irá ser encaixado um novo segmento de tamanho segundo o algoritmo *<inserir algoritmo aqui>*