Sistemas Operativos

2025/26

Gestão de memória
Endereçamento virtual
Memória virtual

1

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

2

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 diretamente a posições na memória física existente no computador
  - Isto significa que as instruções (instruções do processador: código máquina) em execução no processo "vêm" a memória tal como ela **real**mente é e têm acesso não restringido a qualquer zona da memória do computador
  - *"as instruções «verem» a memória tal como ela é"* parece ser a situação normal mas traz problemas graves
    - Falta de segurança e estabilidade
    - Falta de flexibilidade na gestão da memória
    - Dificuldade em ter vários programas a correr ao mesmo tempo

    Apesar de parecer a forma mais "normal", devido a estes problemas o endereçamento real não é usado a não ser em sistemas muito simples -> existe uma alternativa que é descrita mais adiante

Gestão de memória

Endereçamento real (memória real)

- Problemas no endereçamento real

- Falta de segurança e de estabilidade
  - Qualquer programa pode ver os dados dos outros processos e interferir com a sua execução, inclusivamente com o próprio sistema operativo
  - Isto pode ocorrer por intenção maliciosa ou por mero bug
- Dificuldade na capacidade de multi-programação
  - Devido à falta de estabilidade e de segurança (pontos anteriores)
  - Devido à eventual colisão de endereçosnecessários a aplicações diferentes: eventualmente as aplicaçõespoderão precisar de usar endereços específicos que podem entrar em conflito com igual necessidade por parte de outras aplicações
- Dificuldade na gestão e flexibilização do uso de memória
  - Torna-se mais difícil simular a existência de mais memória que aquela que realmente existe, impedindo a execução e aplicações de grande dimensão

Gestão de memória – Sistemas de endereçamento real

Endereçamento real ("memória 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

- A cada processo é dado um "conjunto de endereços" (ou seja um ***espaço de endereçamento***) **sem relação direta** com a memória real
  - O processador traduz automaticamente esses endereços para endereços físicos ("físicos" -> os que realmente existem na RAM).
  - A tradução é feita de forma transparente às instruções a executar

Endereços virtuais de um determinado **espaço de endereçamento virtual**

- São os endereços utilizados pelos processos em execução e que não correspondem diretamente à memória física (memória física = RAM) Cada processo tem o seu espaço de endereçamento virtual

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 em Processos

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

- Cada bloco é indivisível e todos os bytes do mesmo bloco partilham a mesma forma de tradução. Isto significa que:
  - Onde quer que o primeiro byte (o primeiro endereço virtual desse bloco) seja mapeado em memória física, o segundo byte do bloco estará no byte seguinte na memória física e por assim a diante
  - A tradução (feita pelo processador) é feita com base numa tabela designada tabela de tradução e endereços
  - Nessa tabela bastará referir onde fica mapeado o primeiro byte do bloco: os bytes seguintes estão "a seguir"
  - Existem outros aspetos relativos à forma como o bloco é usado. Esses aspetos são também descritos na tabela e todos os bytes do bloco são geridos da mesma forma
  - Blocos diferentes terão mapeamento e regras de uso diferentes

Gestão de memória – Sistemas de endereçamento virtual

Endereçamento virtual

Algumas características mais salientes

- O espaço de endereçamento de cada processo é subdividido num conjunto de blocos.
- O conjunto de blocos no espaço de endereçamento de um processo não abrange a totalidade desse espaço de endereçamento: o espaço não abrangido por blocos corresponde a zonas de memória que o processo não pode usar e não ocupa memória RAM.
- A tradução **endereço virtual → endereçamento real** é feito **bloco a bloco** pelo processador de forma transparente para as instruções e recorrendo a uma tabela designada ***tabela de tradução e endereços***
  - Preenchida pelo sistema e à qual o programa no processo não tem acesso
  - Cada processo tem a sua tabela, fazendo com que cada processo tenha um espaço de endereçamento distinto e independente

Gestão de memória – Sistemas de endereçamento virtual

Tabela de tradução de endereços

Cada linha desta tabela descreve um bloco. Existirá uma linha por bloco e cada processo tem a sua própria tabela. A informação nesta tabela descreve o espaço de endereçamento virtual do processo

Campos típicos da tabela

- **Endereço Base**: endereço em memória física (RAM) onde ficou mapeado o primeiro byte do bloco (os bytes restantes desse bloco "estão a seguir pela mesma ordem")
- **Limite** ou **Tamanho**: tamanho do bloco em questão (exceto em memória paginada em que todos os blocos têm o mesmo tamanho)
- **Proteção** ou **Permissões**: bits que descrevem se os bytes no bloco podem ser lidos, escritos executados
- **Modo** de execução: modo que o processador assume quando executa instruções neste bloco (esta informação pode estar junta com a proteção)
- **Presença** ou **P**: indica se o bloco está efetivamente em memória, ou temporariamente em disco bloco segundo a lógica da memória virtual.
- Bits **M** e **R**: para memória paginada (descritos mais adiante)

Gestão de memória – Sistemas de endereçamento virtual

Endereçamento virtual

Algumas características mais salientes

- 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**")
- A tabela de tradução de endereços permite especificar diversos aspetos do uso do bloco: qual o seu tamanho, se está ou não em memória (ou disco), qual o uso a que se destina (código, dados), se for código, qual o nível de execução, etc. O processador usa estes dados.

Gestão de memória – Sistemas de endereçamento virtual

Endereçamento virtual – espaço de memória de um processo

- O sistema configura para cada processo um espaço de endereçamento virtual coincidente coma capacidade máxima que o processador consegue endereçar (dependendo da arquitetura: 32 bits, 64 bits, etc.)
- O espaço de endereçamento (virtual) de um processo fica organizado em blocos. Um endereço virtual é composto pelo bloco e pelo deslocamento dentro desse bloco
- A tradução de endereço feita pelo processador usa o número do bloco para aceder à tabela de endereços. O processador valida o acesso de acordo como que está descrito nessa linha da tabela (exemplo, o deslocamento está dentro do tamanho desse bloco? O acesso está de acordo com as permissões de uso desse bloco? Etc.)
- O uso de um endereço válido (bloco existe e deslocamento é correto) mas contrário ao que está descrito na tabela de endereços causará um erro (exemplo: operação de escrita em blocos só de leitura)

Gestão de memória – Sistemas de endereçamento virtual

Endereçamento virtual – espaço de memória de um processo

- Apenas os blocos que se encontram efetivamente mapeados em RAM é que ocupam efetivamente espaço. Estes blocos, no seu total, ocupam uma ínfima parte do total do espaço endereçável do processo.
- Ou seja, apesar do processo ter a ideia de um espaço de memória coincidente com a capacidade máxima endereçável pelo processador, na verdade, o processo ocupa muito menos espaço em memória do sistema
- Isto possibilita a existência de mais processos. Cada processo temos seus blocos mapeados em **zonas não sobrepostas na RAM**, impossibilitando os processos de verem a memória uns dos outros
- No entanto, dois processos podem pedir ao SO que mapeie blocos em zonas coincidentes – trata-se do **mecanismo de memória partilhada**. O SO usa o mesmo mecanismo para se manifestar simultaneamente nos espaços de endereçamento de todos os processos na máquina.
- A tentativa de uso de uma posição de memória que não esteja abrangida por um bloco efetivamente mapeado em RAM causará o processador lançar uma exceção “endereço inválido”.

Gestão de memória

- ***Memória virtual***:

  (Não confundir com "endereçamento virtual")

  -> Corresponde à ideia de simular mais memória do que a que realmente existe no computador recorrendo a memória secundária (disco)

  - O espaço ocupado pelos blocos de todos os processos é, normalmente, maior que o total da memória física realmente existente.
  - Os blocos menos usados são colocados temporariamente em disco (partição swap / page file), ficando esse fato assinalado na tabela de tradução de endereços.
  - Quando o processo a que pertence o bloco gerar (usar) um endereço num bloco que está em disco, o processador apercebe-se dessa situação (está assinalada na tabela – bit **P**) e gera uma exceção que é tratada pelo sistema o qual recoloca o bloco em memória, atualizando a tabela para refletir a nova posição do bloco (pode ter ficado num local diferente do original)

Gestão de memória

- ***Memória virtual***:

  - O sistema gere a transferência de blocos de/para disco conforme a ocupação, uso e necessidade de memória.
  - A escolha do bloco a colocar em disco é um aspeto crítico com grande impacto na performance geral do sistema, existindo vários algoritmos para fazer esta escolha e que variam bastante consoante se trata de ***memória segmentada*** (poucos blocos, grandes e cada um com o seu tamanho) ou de memória paginada (mutos blocos, todos do mesmo tamanho e muito pequenos)
    - Memória segmentada e memória paginada são descritas mais adiante
  - A quantidade de espaço em disco usado para estender a memória RAM segundo a ideia de memória virtual é tipicamente 50% da quantidade de memória RAM
    - Se se exagerar na quantidade de disco para simular mais RAM a performance do sistema pode ser muito afetada, podendo entrar-se em situação de thrashing

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 (tal como já anteriormente descrito):
  - Tamanho e localização em memória física
  - Tipo de acesso permitido (proteção: leitura, escrita, execução) e modo de 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: genericamente registos BTS e LTS (isto depende muito do processador em questão)

Gestão de memória – Sistemas de endereçamento virtual

Memória segmentada

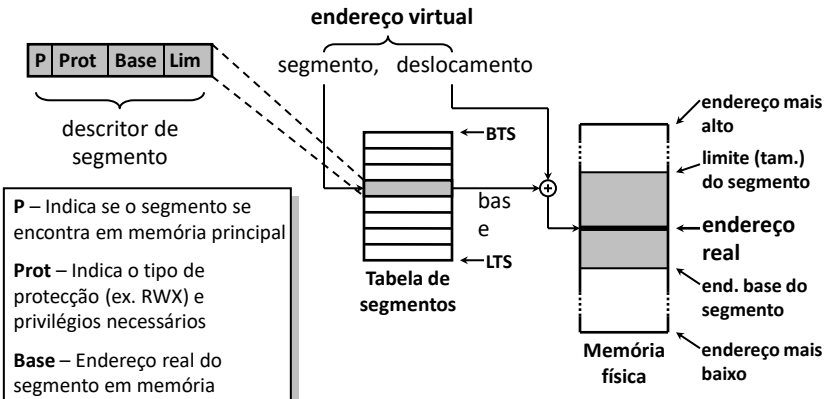
- Um endereço virtual é composto por um conjunto de bits que indicam qual o descritor de segmento dentro da tabela de segmentos que vai ser utilizado (abrev. "qual o segmento") e um conjunto de bits que indica qual o deslocamento dentro desse segmento
- A obtenção do endereço real que corresponde ao endereço virtual é simples
  - O número do bloco identifica a linha na tabela de segmentos
  - Se o bloco estiver presente e se o acesso for válido é somado ao endereço base o deslocamento.
  - São validados todos os aspetos descritos na tabela: tamanho do bloco vs. deslocamento, proteção, tipo de operação vs. Acessos permitidos, presença do bloco etc. Se algo falhar é gerada uma exceção (pelo processador) com a típica consequência do SO terminar o processo em questão
  - Se o bloco não estiver presente, é gerada uma exceção, mas a instrução e uso do endereço são válidos, sendo a instrução concluída assim que o sistema tiver trazido o bloco de disco para memória (entretanto o processador executa outro processo)

Gestão de memória – Sistemas de endereçamento virtual

Memória segmentada

- 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.
- Numa arquitetura semelhante à da Intel, existirão processadores dedicados para conter a totalidade das linhas da tabela mais um uso. Sempre que se muda o valor num desses registos, a linha correspondente da tabela é copiada para esse registo.
  - Os DS, ES,SS e CS do antigo 8086 são agora a "ponta do iceberg" sendo a parte visível de registos muito maiores que contêm linhas inteiras da tabela.
    - Nota: o 8086 não suportava endereçamento virtual e os nomes "segmento" e "offset" no contexto do 8086 não significam coisa nenhuma relativamente a endereço virtual (o cenário muda de figura a partir do 80386)
- 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



P – Indica se o segmento se encontra em memória principal

Prot – Indica o tipo de protecção (ex. RWX) e privilégios necessários

Base – Endereço real do segmento em memória

Lim – Dimensão do segmento (cada segmento pode ter um tamanho diferente)

BTS – End. base da tabela de segmentos

LTS – Limite (tamanho da tabela

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)
- Tabelas de páginas pequenas e fáceis de gerir

Gestão de memória – Sistemas de endereçamento virtual

Memória segmentada

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
- Se se retirar um bloco a um processo (para transferir para disco) todo o processo ficará muito provavelmente impedido de executar. A escolha do bloco a retirar passa a incidir em politicas de escalonamento em vez de politicas de gestão de memória

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 (***P**age **T**able **E**ntry*). Cada PTE descreve uma página quanto a:
  - Localização em memória física (por outras palavras, qual a *page-frame* em memoria física onde está mapeada)
  - Tipo de acesso permitido (proteção) e modo de execução
  - Estado de presença em memória física (bit "P")
  - Utilização da página quanto a referência (bit "R") e modificação (bit "M")
  - Não existe o campo de tamanho ou limite dado que as páginas tem todas o mesmo tamanho e esse conhecimento está "*built in*" no proprio processador
- Os registos BTP e LTP são os correspondentes aos BTS e LTS da memória segmentada

Gestão de memória – Sistemas de endereçamento virtual

Memória paginada

Na tradução de endereços em memória paginada existe uma diferença relativamente à memória segmentada

- O endereço virtual não tem duas partes explicitas (bloco e segmento). Em vez disso, o endereço virtual é apenas um único número com o significado "*número de bytes a contar desde o início do espaço de endereçamento*" (conceito tradicional de endereço)
- O processado atribui automaticamente aos bits menos significativos o significado de deslocamento. Aos restantes atribui o significado de número de página (número de bloco).
- O número de bits que são usados para o significado de deslocamento depende do tamanho da página. São usados tantos bits quantos os necessários para gerar números suficientemente grandes para cobrir a totalidade da página.
  - Exemplos:

    páginas de 256 bytes-> 8 bits. Páginas de 1K -> 10 bits. Páginas de 4K -> 12 bits

Gestão de memória – Sistemas de endereçamento virtual

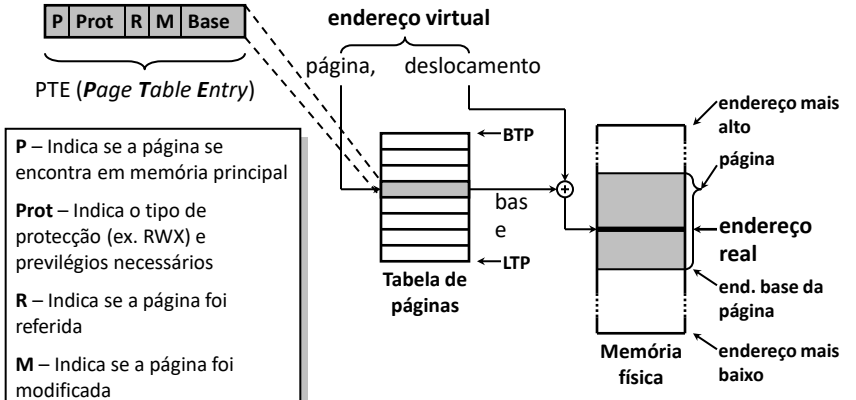
Memória paginada (cont.)

- O facto das páginas terem todas o mesmo tamanho tem como consequências:
  - É possível a criação de um espaço de endereçamento perfeitamente linear
    - Significa que quando se ultrapassa uma página se passa automaticamente para a seguinte (se existir), coisa que não acontece na memória segmentada
    - Um endereço é apenas um número, sem partes explícitas (visíveis para o programador) de número de bloco e de deslocamento, tornando-se mais fácil na programação em assembly
  - É possível otimizar bastante e simplificar muito 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 principio da localidade de referência, as próximas a serem necessárias).

  Esta tabela denomina-se de TLB (***T**ranslation **L**ookaside **B**uffer*)

Memória paginada – Tradução dos endereços



P – Indica se a página se encontra em memória principal

Prot – Indica o tipo de protecção (ex. RWX) e previlégios necessários

R – Indica se a página foi referida

M – Indica se a página foi modificada

Base – Endereço real (optimizado) da página em memória

BTP – End. base da tabela de páginas

LTP – Limite (tamanho)da tabela

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 com vantagens de performance na implementação e memória virtual
- Não existe o problema de fragmentação no uso de memória RAM dado que todas os espaços vazios em RAM candidatos a receber um bloco "página") são igualmente bons por terem todos o mesmo tamanho que os blocos
- Pode-se tirar uma página a um processo que este não fica necessariamente impedido de executar.
  - Graças ao principio de localidade de referência, o processo nem precisará dessa página no imediato, facilitando a implementação de memória virtual

Gestão de memória – Sistemas de endereçamento virtual

Memória paginada

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)
- Causa tabelas de páginas muito grandes, com prejuído da sua gestão por parte do sistema

Gestão de memória

- Em 25/26 não foram abordados os esquema de endereçamento paginado multi-nível nem paginado-segmentado.
  - Em termos de estudo para o exame, pode passar para o slide **35**
    - A matéria nesse slide e seguintes **foi** abordada

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

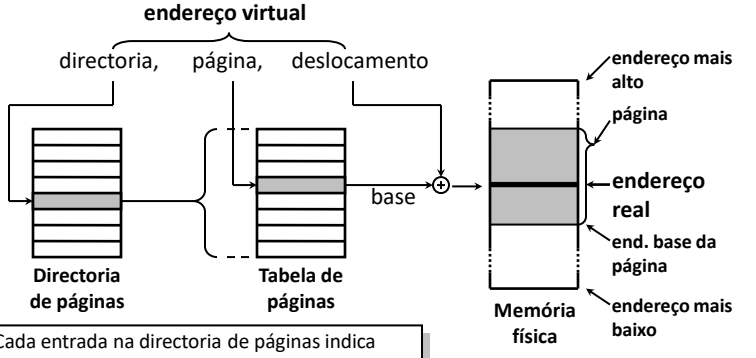
→ Tabela de páginas teria (até) 2^{52} entradas !

↓

Não é viável manter a tabela em memória

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

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

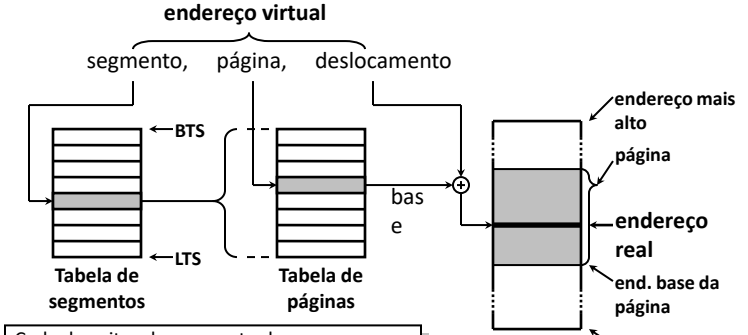
Gestão de memória – Sistemas de endereçamento virtual

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

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.

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 os 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 (políticas) 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 optimizar 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 lisa 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 geralmente por **necessidade**:
  - Quando um processo é criado, as suas tabelas de páginas são inicializadas mas as páginas não são colocadas em memória física
  - 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). Só ocupa memória física se realmente precisar.
  - Também é habitual usar a política de antecipação para otimização dos acessos a memória secundária (traz a página em falta e as seguintes)
  - 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 e memória física.

  Exemplo de otimização: o mecanismo fork -> usa ***copy-on-write***

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 exceptções, 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 as 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 ***F**irst **I**n **F**irst **O**ut*
  - LRU ***L**east **R**ecently **U**sed*
  - NRU ***N**ot **R**ecently **U**sed*

Gestão de memória – Substituição de páginas – FIFO

Algoritmo **FIFO** – ***F**irst **I**n **F**irst **O**ut*

- 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** – ***L**east **R**ecently **U**sed* (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 (o processador*, pelo que é invulgar.

  ⇒ 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** – ***N**ot **R**ecently **U**sed* (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 sairem 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 ⇒ está a ser necessária (utilizada recentemente) – provavelmente será necessária brevemente (pelo princípio da localidade de referência)

Modificada ⇒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

⇒ 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 "vitima" para ter os seus segmentos retirados de memória principal

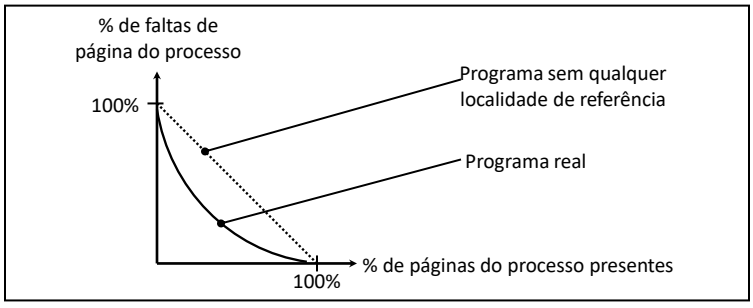
Factores a levar em conta em relação ao processo (cujos segmentos vão ser retirados)

- 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 memoria 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?

(continua)

Aferição de conhecimentos neste capítulo

Tipos de perguntas que se podem fazer sobre esta matéria (meros exemplos)

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

- Dado este conjunto de páginas e os instantes em que foram acedidas e modificados, diga qual será a primeira a sair de acordo com o algoritmo FIFO/LRU/NRU e porquê
- Dada esta tabela de páginas (ou de segmentos) (dados dessa tabela) , diga quais os endereços reais que corresponde a estes (quais…) endereços virtuais.
- Dada esta (dados da tabela) tabela de páginas (ou de segmentos) 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)

(continua)

Aferição de conhecimentos neste capítulo

Tipos de perguntas que se podem fazer sobre esta matéria (meros exemplos)

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

- Dada esta informação acerca de endereços virtuais e os correspondentes endereços reais, reconstrua a tabela de tradução de endereços que lhe corresponde.
- Dado este gráfico (imagem desse gráfico) de variação de page faults face à percentagem de páginas do processo presentes, qual seria a escolha ideal para a dimensão do espaço de trabalho do processo que lhe corresponde
- Dado este processo (código) diga qual acha que será a dimensão (%) do seu espaço de trabalho.
- Dado um determinado (descrição) cenário de ocupação de memória real em memória segmentada, diga onde irá ser encaixado um novo segmento de tamanho segundo o algoritmo … (indicação do algoritmo)

Estes são exemplos de tipos de perguntas possíveis. Mas podem sair outras aqui então estejam aqui