Universidade São Judas Tadeu
Ciência da Computação

Cecília de Oliveira Martins - RA:81620964
Dennis Siqueira de Oliveira - RA:81621468
Douglas de Oliveira Lima - RA:817124153
Gabriel Ferreira da Silva - RA: 81621851
Wellington Shiniti Kawashima - RA: 81622278

Projeto Integrado - Estratégias de gerenciamento e alocação de memória

Butantã - SP
2019

Cecília de Oliveira Martins - RA:81620964
Dennis Siqueira de Oliveira - RA:81621468
Douglas de Oliveira Lima - RA:817124153
Gabriel Ferreira da Silva - RA: 81621851
Wellington Shiniti Kawashima - RA: 81622278

Projeto Integrado - Estratégias de gerenciamento e alocação de memória

Trabalho de Projeto Integrado do curso de Ciência da Computação apresentado a Universidade São Judas Tadeu.
Orientadora: Prof.ª. Dra. Milkes Alvarenga

Butantã - SP
2019

Lista de figuras

Sumário

1. Introdução

Considerada uma das áreas mais fundamentais da programação de computador por consultores da própria IBM (International Business Machines), a organização e o gerenciamento de memória são fundamentais para o sistema operacional. Conhecer as capacidades e limitações do gerenciamento de memória leva a uma programação mais eficiente e ágil. Programações que envolvem C ou C++ é um exemplo onde esse gerenciamento precisa ser constante, diferente do assembly da Apple. Entender as limitações de memória vai desde a determinar qual memória é adequada para o processamento, se precisará ser realocada para memórias disponíveis e enfim, liberar para que outro processo possa utilizá-la.

2. Memória

A maioria dos computadores trabalha com o conceito de hierarquia de memória, possuindo uma pequena quantidade de memória cache, muito rápida, uma quantidade de memória principal (RAM) e uma quantidade muito grande de memória de armazenamento em disco (HD), considerada lenta [1].

2.1 Hierarquia de Memória

O conceito de hierarquia de memória fornece uma visão do custo, desempenho e capacidade das memórias.

Na **base da pirâmide**, tem-se as memórias de **baixo custo**, maior capacidade de armazenamento e menor desempenho. No **topo da pirâmide**, tem-se as memórias de **maior custo**, menor capacidade de armazenamento e maior desempenho [8].

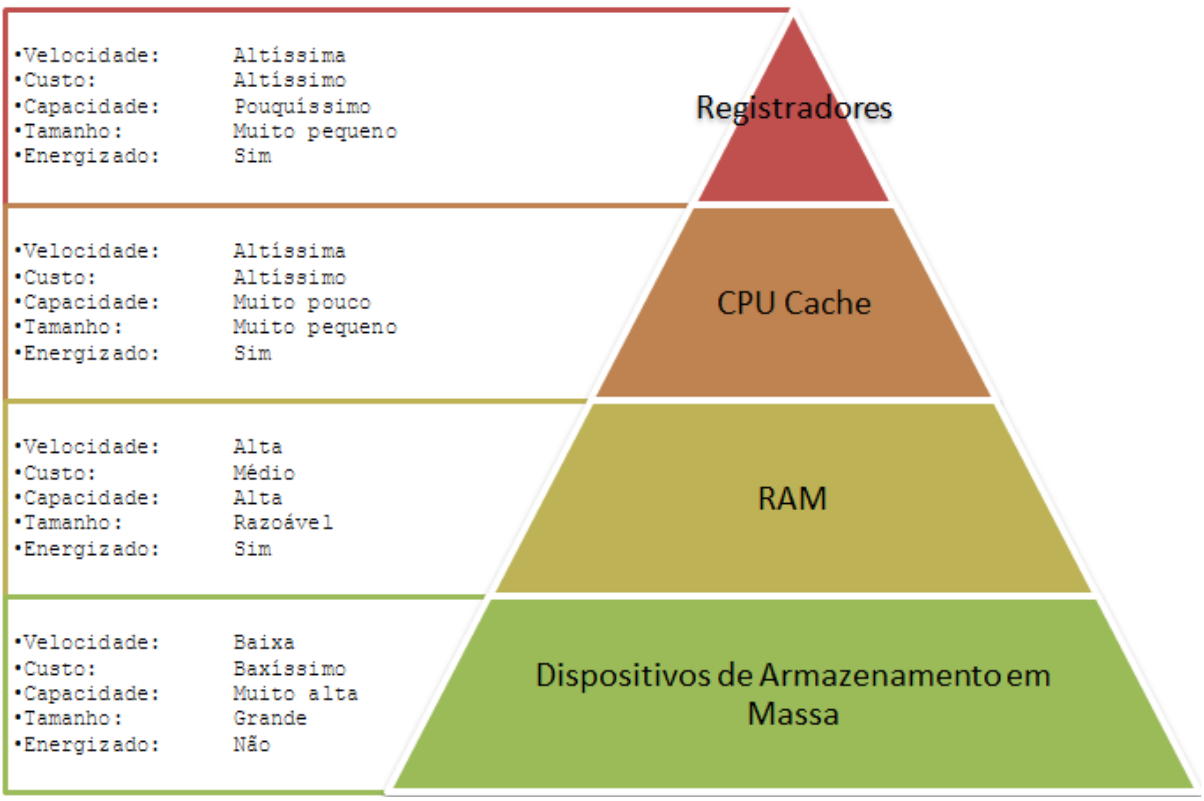


Figura 1 - Hierarquia de Memória

2.2 Registradores

Memórias individuais agrupadas para armazenar uma palavra em binário. São de alto desempenho, usadas pela CPU ou por outros dispositivos para armazenamento temporário de dados. Alguns registradores na CPU participam do processamento e podem ser acessados por programas. Registradores especiais somente podem ser acessados pela CPU.

2.3 Memória Cache

A memória cache é um tipo de memória que trabalha em conjunto com o processador. De fato, todos os processadores atuais trazem uma certa quantidade de memória cache embutida no encapsulamento. O objetivo é potencializar o desempenho do chip de processamento, evitando que fique ocioso por longos períodos. Esse tipo de memória possui alta velocidade e tem por função armazenar dados e instruções que a CPU poderá precisar em breve. Ela possibilita que o processador trabalhe com toda a capacidade e tenha o mínimo de tempo ocioso possível [2].

Cada fabricante utiliza a memória cache de uma forma diferente. Isso também pode variar de acordo com a microarquitetura usada no chip. No entanto, o padrão é que, quando a CPU precisa buscar a sua primeira instrução, ela terá de ir até a memória RAM, visto que a memória cache estará vazia. Apesar disso, em vez de trazer apenas a solicitação feita pela CPU, a unidade de busca traz um bloco inteiro de instruções que, por sua vez, é armazenado na memória cache. Assim, se o processador for continuar a executar o referido programa, as instruções subsequentes estarão já armazenadas na memória cache. Então, a unidade de busca não precisará ir até a memória RAM para obtê-las [2].

Nem sempre a unidade de busca armazena as informações corretas na memória cache. No entanto, a taxa de acerto é bem alta, cerca de 80% a 99% das vezes. Com isso, é possível afirmar que quase todo o acesso à memória RAM é feito através da memória cache.

A memória cache é dividida em alguns níveis, conhecidos como L1, L2 e L3 (L significa Level, em inglês). Eles dizem respeito à proximidade da memória cache das unidades de execução do processador. Quanto mais próxima ela estiver da unidade de execução do processador, menor será o seu número [2].

Assim, o cache L1 é o mais próximo possível. O L2 é um pouco mais distante e o L3 é ainda mais distante. Sempre que a unidade de busca do processador precisa de um novo dado ou instrução, ela procura inicialmente no cache L1. Se não encontrar, parte para o L2 e depois para o L3. Se a informação não estiver em nenhum dos níveis de memória cache, ela terá de ir até a memória RAM [2].

Uma peculiaridade a respeito dos níveis, é que o L1 é dividido em memória de instrução e memória para dados. Com isso, o processador vai direto à memória de instrução, se estiver buscando uma instrução, ou vai direto à memória de dados, se estiver buscando um dado. Isso agiliza ainda mais o processador de busca [2].

2.4 Memória RAM

A memória RAM é um tipo de tecnologia que permite o acesso aos arquivos armazenados no computador. Diferentemente da memória do HD, a RAM não armazena conteúdos permanentemente. É responsável, no entanto, pela leitura dos conteúdos quando requeridos. Ou seja, de forma não-sequencial, por isso, a nomenclatura em inglês de Random Access Memory (Memória de Acesso Aleatório)[3].

Para simplificar a lógica por trás da função da memória RAM, é possível fazer uma analogia com uma mesa de estudos, onde se reúne todo o material necessário para realizar os deveres de casa: como canetas, lápis, caderno e livros. Os materiais seriam os arquivos e a memória RAM, a mesa, onde tudo se reúne e o trabalho é feito[3].

Sendo assim, a memória RAM pode ser entendida como um espaço temporário de trabalho, pois, após a tarefa ser realizada, os arquivos (material de estudos) são retirados da memória (mesa) e mantidos no HD (armário)[3].

Assim como a mesa, quanto maior a memória RAM, maior sua capacidade de trabalho. Mas a capacidade da mesa é medida em área. Quanto maior a área da mesa, mais livros cabem e mais rapidamente se faz o trabalho. Já a capacidade da memória RAM, mede-se pelo fluxo de bits suportados nas operações. Ou seja, para se acessar uma grande quantidade de memória no HD de uma só vez, como muitos programas atuais exigem, é necessário uma grande quantidade de memória RAM. São estes, portanto, os megabites ou gigabites que aparecem nas configurações[3].

A memória RAM é um chip semelhante a um micro-processador, composto por milhões de transistores e capacitores. O capacitor é uma peça capaz de armazenar elétrons. Quando ele

está carregado, o sistema faz uma leitura com base no famoso código binário. Cada leitura dessa em zero ou um significa um bit de informação. Essa leitura é feita de forma muito rápida, são muitas em poucos milésimos de segundos. É assim que a memória RAM processa todas as ações executadas pelo usuário. Outras características que influenciam na capacidade de processamento da memória RAM são a largura e a velocidade do barramento, que é um conjunto de "fios" responsáveis pela conexão da memória com os outros componentes. A largura nos diz o número de bits que podem ser enviados ao CPU simultaneamente. A velocidade é o número de vezes que esse grupo de bits pode ser enviado a cada segundo.

A memória comunica-se com o CPU, trocando dados, e completa o que se conhece como ciclo de barramento. É esse período que apresenta o desempenho da memória que, pode ser de 100MHz e 32bits, por exemplo. Isto significa que tal memória é capaz de enviar 32bits de dados ao processador 100 milhões de vezes por segundo. No entanto, existe um efeito chamado latência, que atrasa a taxa de transferência de dados de forma significativa quando se envia o primeiro bit[3].

Ao se comprar uma memória deve-se ficar atento para essa questão da taxa de transferência. Não adianta a memória ter uma frequência alta e a frequência do sistema ser menor, pois a taxa do sistema vai limitar a da memória RAM. Portanto, para um sistema que rode a 100MHz e 32bits, compre uma memória com os mesmos aspectos[3].

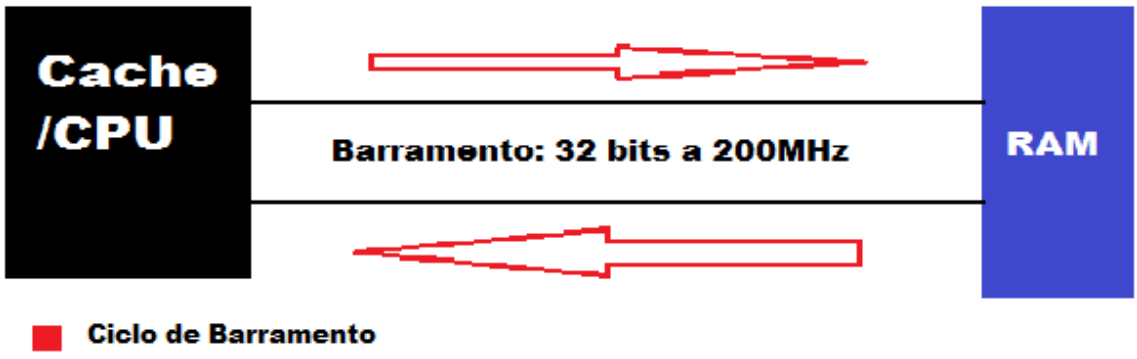


Figura 2 - Tráfego de dados Memória RAM x CPU

2.5 Memória de Armazenamento em disco (HD)

Disco rígido ou disco duro, popularmente chamado também de HD (derivação de HDD do inglês hard disk drive), "memória de massa" ou ainda de "memória secundária" é a parte do computador onde são armazenados os dados. O disco rígido é uma memória não-volátil, ou seja, as informações não são perdidas quando o computador é desligado, sendo considerado o principal meio de armazenamento de dados em massa. Por ser uma memória não-volátil, um sistema é necessário para se ter um meio de executar novamente programas e carregar arquivos contendo os dados inseridos anteriormente quando ligamos o computador. Nos sistemas operativos mais recentes, ele é também utilizado para expandir a memória RAM, através da gestão de memória virtual. Existem vários tipos de interfaces para discos rígidos diferentes: IDE/ATA, Serial ATA, SCSI, Fibre channel, SAS [4].

3. Gerenciamento de Memória

Gerenciador de memória é uma parte principal do Sistema Operacional que controla quais partes da memória estão em maior uso, como alocar de maneira adequada à processos quando necessário, especialmente entre a memória principal e o disco [5].

3.1. Formas de gerenciamento de Memória

Dentre as maneiras para se gerenciar a memória, podemos citar o **gerenciamento sem troca ou paginação** – sendo o mais simples, que não havendo transferência entre a memória e o disco –, a **monoprogramação sem troca ou paginação** – ocorre quando há apenas um processo por vez utilizando toda a memória disponível, onde o sistema operacional carrega e executa um programa do disco para a memória, aguardando um novo programa ser executado para sobrepor o primeiro – e a **multiprogramação** – Não adequada para sistemas grandes, facilita a programação de aplicações por ser dividida em mais processos e auxiliam em serviços interativos simultâneos [5].

Outra forma de gerenciar a memória é gerenciar o espaço utilizado, sendo por mapeamento de bits – a memória fica dividida em unidades de tamanho específico, representando um bit, sendo 0 interpretado como “memória livre” e 1 como “memória ocupada”, sendo seu ponto negativo a escalação de processos com um número específico de

bits necessários, tendo que navegar a memória para achar o local onde esse mesmo número é consecutivo.

O gerenciamento de espaço pode ser executado por listas encadeadas também, sendo listas de segmentos alocados e livres, ligados entre si ou com buracos, sendo organizada em ordem de endereço. Quando um espaço é liberado, os processos são reestruturados, com exceção do primeiro e do alocados, para liberação do espaço necessário para o próximo processo.

A alocação de espaço de troca (SWAP) e Memória virtual. A memória virtual é uma técnica que aloca memória quando a mesma não cabe no espaço existente, sendo parte alocada parte no disco e parte na memória RAM, memória de cache temporário, sendo feita de forma automática pelo Sistema Operacional, que soma a quantidade de memória, havendo uma soma que ultrapassa a quantidade real de memória.

3.2. Memória Virtual

A primeira solução adotada para programas grandes demais para a quantidade de memória foi a utilização de overlays. Nesta técnica o programa era subdividido em partes menores (overlays), que podiam ser rodadas separadamente e quando um overlay terminava a execução um outro poderia ser carregado na mesma posição de memória utilizada pelo anterior. O problema é a divisão do programa em overlays não é simples e deve ser realizada pelo programador.

A técnica de memória virtual para executar um programa que não cabe na memória existente consiste em manter partes do programa, dos dados e da pilha no disco, sendo que existe uma forma de decisão de quais processos devem permanecer no disco e quais na memória. Esta técnica é realizada de forma automática pelo computador. Podemos alocar diversos processos na memória virtual, de forma que cada um pensa ter uma quantidade de memória que somadas ultrapassam a quantidade real de memória.

Técnicas utilizadas em sistemas com memória virtual:

Paginação: espaço virtual é o espaço de memória que pode ser referenciado por um programa qualquer em dado processador. Ex: um processador com endereçamento de 16 bits, possui um espaço virtual de 64 kbytes. Quando uma instrução como: LD A,(1000h) o 1000h corresponde a um endereço virtual, de um espaço de endereçamento virtual de 64k bytes. Em

um computador sem memória virtual, o endereço virtual corresponde ao endereço efetivamente colocado no duto de endereçamento da memória. Quando o computador possui memória virtual, esse endereço virtual é enviado para uma unidade de gerenciamento de memória (MMU), que corresponde a um chip ou um conjunto de chips que translada esse endereço virtual em um endereço físico, de acordo com uma tabela.

O espaço de endereços virtuais é dividido em unidades chamadas páginas e o espaço de memória física é dividido em unidades chamadas quadros de página, de mesmo tamanho das páginas. A MMU tem uma tabela que indica para cada página, qual o quadro de página que corresponde à mesma. Se o processador tenta acessar o endereço 0, a MMU verifica que isto corresponde ao primeiro endereço da primeira página, verifica então que essa primeira página está alocada no terceiro quadro de página. Converte então esse endereço para 8192 (decimal) e envia o endereço convertido para a memória (nem a memória e nem o processador precisam ficar sabendo da existência de paginação).

Como nem todas as páginas do espaço virtual podem estar residentes na memória simultaneamente, ocorrer o caso de que um acesso seja realizado para uma página que não está na memória. Para saber isso a MMU mantém na tabela de translação um bit para cada página que indica se a mesma está presente ou não na memória. Se um acesso for realizado a uma página ausente, é gerada uma falta de página, que chama uma rotina de tratamento de interrupção específica para o SO, que então se encarrega do carregamento da página faltante e o ajuste correspondente na tabela de translação.

A forma mais comum de implementação da MMU, é escolher alguns dos bits mais significativos do endereço virtual como indicadores do número de página e o restante dos bits como um deslocamento dentro dessa página. Ex: na figura acima, de 16 bits do endereço virtual, 12 serão utilizados para o deslocamento e 4 serão utilizados como um índice para qual das 16 páginas está sendo referenciada. A MMU pega os 4 bits do índice da página, acessa a posição correspondente da tabela de translação, verifica se a página está presente na memória, se não estiver, gera uma interrupção para carregamento e depois verifica o valor colocado nessa entrada da tabela de translação e os junto aos 12 bits de deslocamento dentro da página.

A paginação fornece uma forma de se conseguir grande espaços de endereçamento lineares em uma quantidade finita de memória física.

Segmentação: Na segmentação temos um espaço bidimensional no sentido de que é dividido em uma um certo número de segmentos, cada um com dado número de bytes. Dessa

forma, um endereçamento é sempre expresso da forma (segmento, deslocamento). Os diferentes segmentos são associados a diversos programas ou mesmo arquivos, de forma que neste caso, os arquivos podem ser acessados como se fossem posições de memória. Na segmentação os programadores tentam colocar entidades diferentes em segmentos diferentes para facilitar o compartilhamento de objetos entre processos diferentes.

4. Estratégias de Gerenciamento de Memória

Estratégias de gerenciamento de memória são projetadas para conseguir o melhor uso possível da memória principal. São divididas em:

Estratégia de busca: determinam quando transferir a próxima porção do programa ou dados para a memória principal por meio do armazenamento secundário.

Estratégia de posicionamento: determina em que lugar da memória principal o sistema deve colocar programas ou dados que chegam.

Estratégias de substituição: quando a memória estiver muito cheia para acomodar um novo programa, o sistema poderá remover uma parte de um programa (ou todo ele) e dos dados que residem concorrentemente na memória. Esta estratégia determina que parte remover.

5. Alocação de Memória Contígua e não Contígua

A alocação contígua consiste em armazenar um arquivo em blocos seqüencialmente dispostos, permitindo ao sistema localizar um arquivo através do endereço do primeiro bloco e da sua extensão em blocos. O aceso é feito de maneira simples, tanto para a forma seqüencial quanto para a direta [9].

Um problema desse tipo de alocação é que quando um arquivo é criado com n blocos, é necessário que exista uma cadeia de n blocos livres disposto seqüencialmente. Nesse tipo de alocação, o disco é visto como um grande vetor, com segmentos ocupados e livres [9].

Na alocação de memória não contigua o processo é alocado na memória onde há espaço disponível, Divide a memória física com tamanho de blocos fixos denominado quadros.

O tamanho da página é uma potência de 2, variando de 512 bytes até 16mb. O S.O. guarda todos os quadros livres e divide a memória lógica em blocos do mesmo tamanho chamado de página. Para executar um programa com tamanho de n páginas, é necessário encontrar n páginas livre para carregá-lo [10].

6. Sobreposições (overlays)

A sobreposição é um método de programação que permite que programas sejam maiores que a memória principal do computador. Um sistema embarcado normalmente usaria a sobreposição devido a limitação da memória física, que é a memória interna para um sistema em chip e a falta das facilidades de memória virtual [11].

A construção de um programa de sobreposição envolve dividir manualmente um programa em blocos de código objeto independentes, chamados de overlays, dispostos em uma estrutura de árvore. Segmentos irmãos, aqueles que estão no mesmo nível de profundidade, compartilham a mesma memória, chamada de região de sobreposição ou região de destino. Um gerente de sobreposição, a parte do sistema operacional ou a parte do programa de sobreposição, carrega a sobreposição desejada da memória externa na sua região de destino quando é necessário. Muitas vezes os ligadores fornecem suporte para sobreposições [11].

7. Multiprogramação por partição fixa e por partição variável

Aumentando a utilização da CPU, a multiprogramação permite que enquanto um processo é bloqueado esperando a E/S acabar, outro pode utilizar a CPU, melhorando a utilização da CPU e assim evitando desperdícios de ciclos de processamento. Na multiprogramação a memória é dividida em N partes, normalmente quando o sistema é iniciado, Os jobs são colocados em filas de entrada associadas à menor partição capaz de armazena-lo, por usar partições de tamanho fixo, todos o restante de espaço de memória não utilizado pelo Job será perdido, tal desperdício é chamado de Fragmentação Interna. Outro tipo de fragmentação existente é a Fragmentação Externa que acontece na seguinte ocasião: imagine que exista duas partições livres, uma de 50 e outra de 80K bytes, nesse instante é criado um

processo de 90K bytes que não poderá ser carregado em memória pela forma como ela é gerenciada, sendo assim acontece uma fragmentação externa para que processo possa "interagir" com a memória. Na Multiprogramação com partições fixas os processos são colocados em filas únicas ou não, esperando alguma partição que possa armazena-los [12].

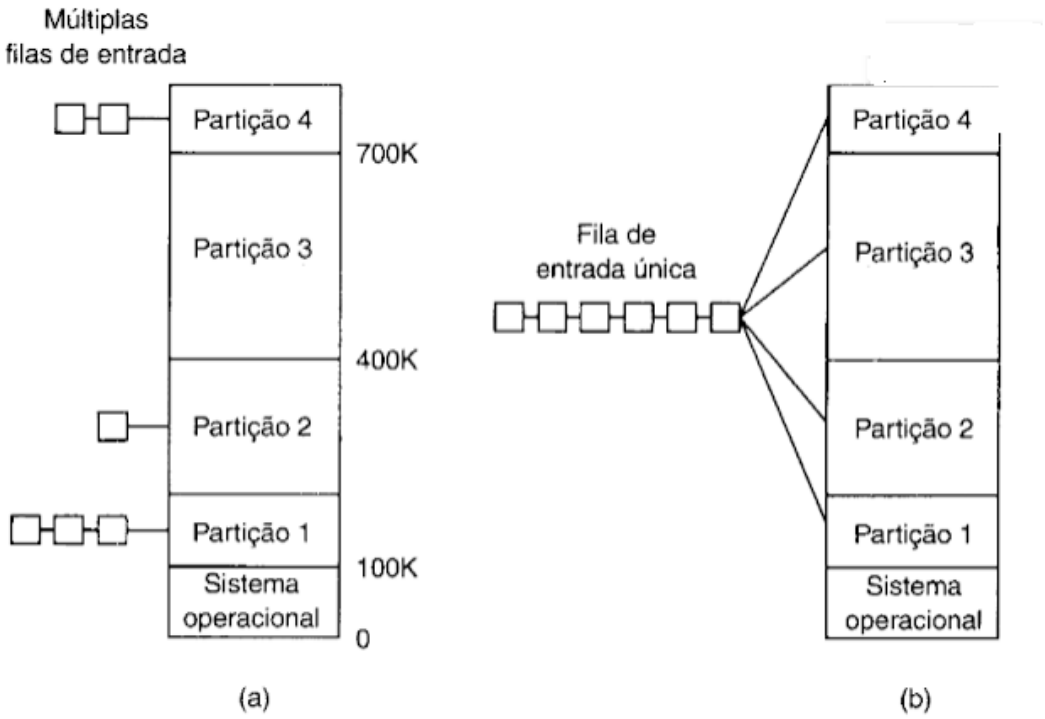


Figura 3 - Multiprogramação com partições fixas

(a) Partições fixas de memória com filas separadas para cada partição.
(b) Partições fixas de memória com uma única fila de entrada.

Multiprogramação com partições Variáveis: o tamanho dos processos na memória podem variar dinamicamente com o passar do tempo, o tamanho das partições é ajustado dinamicamente às necessidades exatas dos processos. A imagem abaixo ilustra o funcionamento deste algoritmo, considerando a ocorrência de swapping(trazer um processo do disco para a memória [swap in] executá-lo durante um intervalo de tempo e depois devolvê-lo ao disco [swap out] [12].

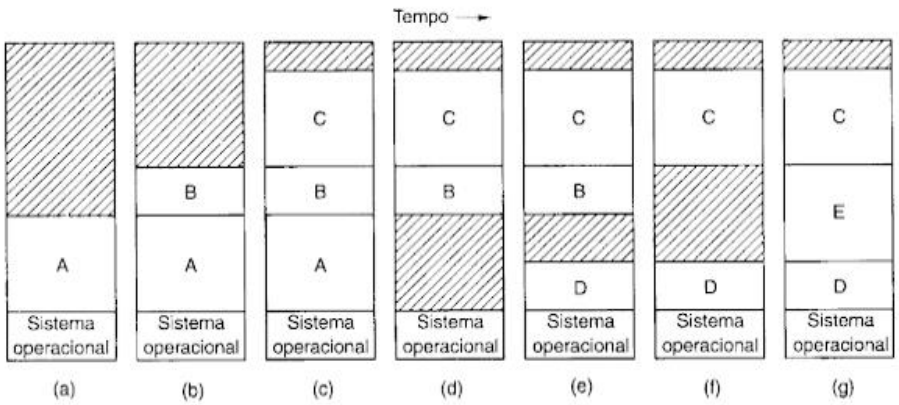


Figura 4 - Multiprogramação com partições Variáveis

8. Estratégias de posicionamento de memória em sistemas de Multiprogramação por partição variável

Diferentemente do esquema de partições fixa, na multiprogramação com partições variáveis o tamanho e a localização dos processos variam, à medida que o mesmo deixa e retorna à memória. A gerencia dos espaços vazios é mais complicada, bem como a alocação e liberação das partições. O sistema operacional mantém uma lista de espaços livres na memória física. Sempre que um novo processo é criado esta lista é percorrida e será usada uma lacuna maior ou igual ao tamanho do processo em questão. O espaço que ultrapassar o tamanho do processo pode dar origem a uma nova partição. As formas de percorrer esta lista é:

First-fit: Inicia a procura a partir da primeira página de memória e vai varrendo até encontrar a primeira lacuna suficientemente grande para armazenar o processo.

Best-fit: varre toda a memória e escolhe a página mais ajustada ao tamanho do processo.

Worst-fit: varre toda a memória e escolhe a página menos ajustada ao tamanho do processo.

Next-fit: segue a mesma idéia do **first-fit**, mas somente a primeira busca é iniciada na parte da memória, as outras iniciam onde terminou a última. Usa-se uma lista circular para permitir que, eventualmente, toda a memória seja percorrida.

Existem processos que tentem a necessitar mais tamanho de memória conforme são executados, para isso, nada melhor que alocar uma pequena memória extra para esses processos,

essa memória extra somente deve constar no momento em que o processo está na memória, quando ele retorna para o disco ele deve conter o tamanho real [12].

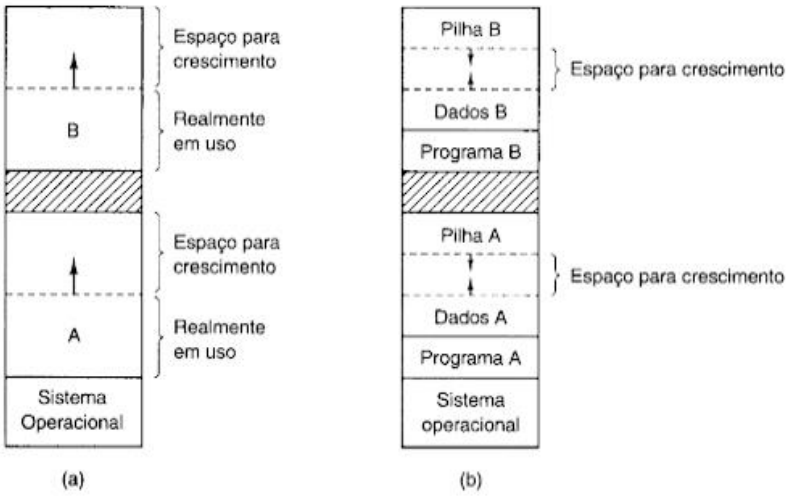
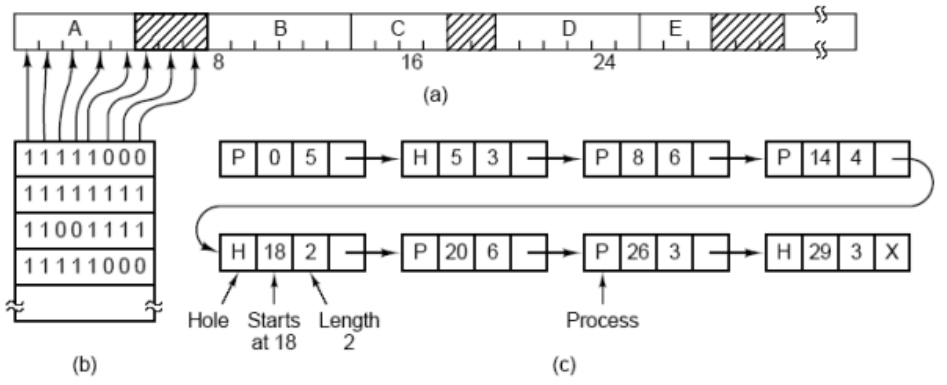


Figura 5 - Alocação de Espaço

(a) Alocando Espaço para um segmento de dados crescente.
(b) Alocando espaço para uma pilha e para um

9. Gerenciamento de Holes



Representação dos espaços de memória com mapa de bits e lista ligada. (a) 5 segmentos alocados a processos e três livres. (b) mapa de bits. (c) lista ligada.

Figura 6 - Gerenciamento de Holes

10. Conclusão

O correto gerenciamento e conhecimento dos tipos de memória é imprescindível para o bom funcionamento de um sistema operacional. Tendo uma visão correta da hierarquia de memória é possível estabelecer uma visão de custo, desempenho e capacidade das memórias da melhor forma, utilizando memórias registradoras para armazenamento temporário de dados pois são de alto desempenho, memória cache para o auxílio nos trabalhos da CPU e ao acesso a memória RAM, a memória RAM para o acesso e a leitura aos arquivos armazenados no computador e a memória de armazenamento em disco (HD) para o armazenamento dos dados.

Também é importante a definição da melhor estratégia a ser tomada na disponibilização de espaço de memória para a execução dos processos, pois a determinação da estratégia correta acarretará em melhor desempenho do sistema operacional.

11. Referências Bibliográficas

[1]https://pt.wikibooks.org/wiki/Sistemas_operacionais/Ger%C3%AAncia_de_mem%C3%B3ria - Acessado em 07/04/2019 – 19:32

[2]https://www.techtudo.com.br/noticias/noticia/2016/10/o-que-e-memoria-cache-entenda-sua-importancia-para-o-pc.html - Acessado em 07/04/2019 - 19:58

[3]https://www.techtudo.com.br/artigos/noticia/2012/02/o-que-e-memoria-ram-e-qual-sua-funcao.html - Acessado em 07/04/2019 – 20:50

[4] https://pt.wikipedia.org/wiki/Unidade_de_disco_r%C3%ADgido – Acessado em 07/04/2019 – 21:06

[5]Colleta,A.F.;EngenheirodeComputação – https://alexcoletta.eng.br/artigos/gerenciamento-de-memoria/ – Acessado em 01/04/2019 às 20:30.

[6]Tanenbaum, A. S. Sistemas Operacionais Modernos. 2ª Edição. Prentice Hall, 2003. – Acessado em 01/04/2019 às 21:30.

[7]Bartlett, Dave; Por dentro do Gerenciamento de Memória - As opções, trade-offs e implementações da alocação dinâmica - https://www.ibm.com/developerworks/br/library/l-memory/l-memory-pdf.pdf - 01/04/2019 às 21:00.

[8] https://www.miltonrocha.eng.br/hierarquia-de-memoria/ - Acessado em 08/04/2019 – 12:11

[9] https://www.gsigma.ufsc.br/~popov/aulas/so1/cap10so.html - Acessado em 08/04/2019 - 12:40

[10]http://hostel.ufabc.edu.br/~marcelo.nascimento/BC1518Q3/aulas/ufabc_SO_Aula7_Gerenciamento%20de%20memoria%20%5BModo%20de%20Compatibilidade%5D.pdf – Acessado em 16/04/2019 – 12:02

[11]https://pt.wikipedia.org/wiki/Sobreposi%C3%A7%C3%A3o_(programa%C3%A7%C3%A3o_de_computadores) – Acessado em 16/04/2019 – 12:30

[12] http://resumindoall.blogspot.com/2012/07/gerenciamento-de-memoria.html - Acessado em 16/04/2019 - 13:00