**Introdução**

O principal objetivo de um sistema de computação é executar programas. Esses programas, juntamente com os dados que eles accssam, devem estar (pelo menos parcialmente) na memória principal durante a execução. Para melhorar a utilização da CPU e a velocidade de resposta aos usuários, o computador deve manter vários processos na memória. Os muitos esquemas distintos de gerência de memória rcflctem várias abordagens; a eficácia de determinado algoritmo depende de cada situação específica. Como a memória principal geralmente é pequena demais para acomodar todos os dados e programas de forma permanente, o sistema de computação deve fornecer armazenamento secundário como apoio. A maioria dos sistemas de computação modernos utilizam discos como o principal meio de armazenamento online para informações (programas e dados). O sistema de arquivos fornece o mecanismo para armazenamento online e acesso aos discos. Os dispositivos que se acoplam a um computador variam em múltiplas dimensões. Os dispositivos transferem um caractere ou bloco de caracteres de cada vez. Podem ser acessados de forma unicamente sequencial ou aleatória. Transferem dados síncrona ou assincronamente. São dedicados ou compartilhados. Podem ser somente de leitura ou de leitura e escrita. Variam muito em termos de velocidade. De várias formas, também são os componentes mais lentos de um sistema de computação. Como a variedade é muito grande, o sistema operacional deve fornecer uma ampla gama de funcionalidades para permitir que as aplicações controlem todos os aspectos dos dispositivos.

**Estrutura de armazenamento em massa**

A memória é fundamental para a operação de um sistema de computação moderno. A memória consiste em um grande vetor de palavras ou bytes, cada qual com seu próprio endereço. A CPU busca instruções da memória de acordo com o valor do contador de programa. Essas instruções também poderão causar carga e armazenamento em endereços de memória específicos. Um ciclo típico de execução de instrução, por exemplo, primeiro busca uma instrução da memória. A instrução é então decodificada e poderá fazer com que operandos sejam buscados na memória. Depois que a instrução tiver sido executada sobre os operandos» os resultados poderão ser armazenados de volta na memória. Observe que a unidade de memória vê apenas um fluxo de endereços de memória; não sabe como eles são gerados (o contador de instrução, indexação, indireção, endereços literais e assim por diante) ou para que servem (instruções ou dados). Da mesma forma, podemos ignorar como um endereço de memória é gerado por um programa.

**Mapeamento de endereços**

Geralmente, um programa reside em disco como um arquivo executável binário. O programa deve ser levado à memória e colocado cm um processo para ser executado. Dependendo do tipo de gerência de memória em uso, o processo pode ser movido entre o disco e a memória durante sua execução. A colcção de processos no disco que está esperando para ser levada para a memória para execução forma a fila de entrada. O procedimento normal é selecionar um dos processos na fila de entrada e carregar esse processo na memória. A medida que o processo é executado, ele acessa instruções e dados da memória. Por fim, o processo termina e seu espaço de memória é declarado disponível. A maioria dos sistemas permite que um processo de usuário resida em qualquer parte da memória física. Assim, embora o espaço de endereçamento do computador comece em 00000, o primeiro endereço do processo de usuário não precisa ser 00000. Esse arranjo afeta os endereços que o programa de usuário pode usar.

Na maioria dos casos, um programa de usuário passará por várias etapas (algumas das quais podem ser opcionais) antes de ser executado (fig. 1). Os endereços podem ser representados de diferentes formas durante essas etapas. Os endereços no programa-fonte são geralmente simbólicos (tais como count). Em geral, um compilador vai fazer a associação desses endereços simbólicos com endereços relocávcis (tais como "14 bytes a partir do início desse módulo"). O carregador ou linkeditor, por sua vez, vai associar esses endereços relocáveis aos endereços absolutos (como 74014). Cada associação é um mapeamento de um espaço de endereçamento para outro.

(reservado a Fig. 1 “As várias etapas do processamento de um programa de usuário”)

A associação de instruções e dados com endereços de memória pode ser feita em qualquer etapa da vida de um programa.

**Em tempo de compilação**: Se a posição de memória onde o processo residirá for conhecida em tempo de compilação, então, um código absoluto poderá ser gerado. Por exemplo, se soubermos a priori que um processo de usuário reside na posição R, o código compilado iniciará naquela posição e se estenderá a partir daí. Se, mais tarde, a posição de início mudar, então será necessário recompilar o código. Os programas de formato .COM do MS-DOS são código absoluto associados em tempo de compilação.

**Em tempo de carga**: Se durante a compilação não estiver determinado onde o processo residirá na memória, o compilador deverá gerar um código relocável. Nesse caso, a associação final c\* retardada até o instante de carga. Se o endereço de início mudar, basta recarregar o código de usuário para incorporar o valor alterado.

**Tempo de execução**: Se o processo puder ser movido durante sua execução de um segmento de memória para outro, a associação deverá ser retardada até o tempo de execução. Hardware especial deve estar disponível para que esse esquema funcione. A maioria dos sistemas operacionais de uso geral utiliza esse método.

**Discos Magnéticos**

Discos magnéticos representam a maior parte do armazenamento secundário em computadores modernos Giram de 60 a 200 vezes por segundo, podem ser fixos ou removíveis, são conectados ao computador através de barramentos de E/S. O controlador hospedeiro no computador utiliza um barramento para se comunicar com o controlador do disco incluindo no próprio drive de disco. Barramentos variam: EIDE, ATA, SATA, USB, FC (Canal de Fibra), SCSI.

* Taxa de transferência é a taxa na qual os dados fluem entre o disco e o computador.
* Tempo de posicionamento (tempo de acesso aleatório) é o tempo para mover a cabeça de leitura para o cilindro desejado (tempo de busca) somado ao tempo necessário para que o setor desejado passe pela cabeça de leitura (latência rotacional)
* Head crash (colisão do cabeçote) acontece quando a cabeça de leitura entra em contato com a superfície do disco
* That’s bad! Geralmente resulta em perda irreparável do disco.

**Estrutura de disco**

Drives de disco são endereçados como arrays unidimensionais de blocos lógicos, onde um bloco é a menor unidade de transferência.

O array unidimensional de blocos lógicos é mapeado de forma sequência em setores do disco, o setor 0 é o primeiro setor da primeira trilha do cilindro mais externo o mapeamento procede em ordem através da trilha, depois através das outras trilhas do mesmo cilindro e depois através dos demais cilindros, do mais externo para o mais interno.

**Conexão de disco**:

- **Local**

Discos conectados localmente ao computador são acessados através de portas de E/S se comunicando com barramentos de E/S, os computadores desktop utilizam barramentos mais simples como o SATA. Servidores de alto desempenho utilizam barramentos mais eficientes como SCSI ou FC.

**SCSI** é uma arquitetura de barramento que permite conectar até 16 dispositivos em um único cabo, possui um dispositivo Iniciador no lado do computador e até 15 Alvos para armazenamento cada alvo pode representar até 8 unidades lógicas (discos conectados a controladores).

**FC** é uma arquitetura serial de alta velocidade, estrutura de chaveamento com espaço de endereçamento de 24 bits: base para as storage area networks (SANs) nas quais vários hospedeiros se conectam a varias unidades de armazenamento Loop arbitrado: pode endereçar até 126 dispositivos.

- **Rede**

Armazenamento conectado a rede (NAS) é uma forma de acessar um dispositivo de armazenamento através de uma conexão de rede ao invés de um barramento de E/S NFS e CIFS são protocolos comuns implementado, utilizando chamadas remotas de procedimento (RPCs) entre o cliente e o armazenamento.

**Storage Area Network**, comum em grandes ambientes de armazenamento (mais comuns a cada dia), múltiplos hospedeiros conectados a múltiplos dispositivos de armazenamento, extremamente flexível.

**Escalonamento de Disco**

 O sistema operacional é responsável pela utilização eficiente do hardware — para os discos, isso significa ter baixos tempos de acesso e boas largura de banda, o tempo de acesso tem dois componentes.

**Tempo de busca**: tempo necessário para mover a cabeça de leitura para o cilindro contendo o setor desejado.

**Latência rotacional**: tempo necessário para que setor desejado passe abaixo da cabeça de leitura.

Objetivo: Minimizar o tempo de busca (tempo de busca ≈ distância de busca).

Largura de banda é a quantidade total de bytes transferidos dividida pelo tempo total entre a primeira requisição e a conclusão da última transferência.

**Escalonamento de disco**

Existem vários algoritmos de escalonamento para atender requisições de E/S de disco, ilustraremos o funcionamento destes algoritmos utilizando a seguinte fila de requisições de E/S para blocos dos cilindros (0-199). 98, 183, 37, 122, 14, 124, 65, 67 Cabeça de leitura inicialmente no cilindro 5314. FCFS possui movimento de 640 cilindros para a cabeça de leitura.

**SSTF**, seleciona a requisição com o menor tempo de busca a partir da posição atual da cabeça de leitura, o escalonamento SSTF é semelhante ao SJF para escalonamento de processos: pode causar starvation. SSTF possui movimento de 236 cilindros para a cabeça de leitura.

**SCAN**, a cabeça de leitura inicia em uma extremidade do disco e se move até a outra extremidade atendendo todas as requisições no caminho, ao chegar ao final o movimento é invertido e o atendimento continua também conhecido como algoritmo do elevador. SCAN (Algoritmo do elevador) possui movimento de 208 cilindros para a cabeça de leitura.

**C-**SCAN, apresenta um tempo de espera mais uniforme que o do SCAN, a cabeça de leitura vai de uma extremidade para a outra do disco atendendo as requisições, porém, ao chegar ao final, volta imediatamente para o começo sem atender nenhuma requisição, trata os cilindros como uma lista de circular.

LOOK e C-LOOK , são versões do SCAN e C-SCAN.

A cabeça de leitura se move apenas até a última requisição em cada direção, depois inverte o sentido.

C-LOOK, selecionando um algoritmo de escalonamento de disco.

SSTF é comum e tem um apelo natural, SCAN e C-SCAN tem melhor desempenho para sistemas que apresentam grande carga de operações de disco, o desempenho depende do número de requisições, as requisição ao disco podem ser influenciadas pelo método de alocação de arquivos o algoritmo de escalonamento de disco deve ser escrito como um modelo separado do sistema operacional, permitindo que seja substituindo por um algoritmo diferente caso seja necessário.

Tanto o SSTF quanto o LOOK são escolhas razoáveis para um algoritmo padrão.

**Gerenciamento de disco**

Formatação de baixo nível ou formatação física, dividir o disco em setores que possam ser lidos e escritos pelo controlador para ser utilizado, para gravar arquivos o sistema operacional ainda precisar gravar suas próprias estruturas de dados no disco.

Particionar o disco em um ou mais grupos de cilindros.

Formatação lógica: criação do sistema de arquivos, bloco de boot inicializa o sistema, o software de boot é armazenado em ROM.

Programa para carga do software de boot, métodos como setores de reserva são utilizados para lidar com bad blocks.

**Gerenciamento de espaço de swap**

Espaço de swap: a memória virtual utiliza espaço em disco como uma extensão da memória principal, o espaço de swap pode ser reservado no sistema de arquivos normal ou, mais comumente, ser alocado em uma partição de disco diferente.

Estruturas de dados para gerenciamento de swap em sistemas Linux.

**Estruturas de RAID**

RAID, múltiplos drives de disco para prover confiabilidade via redundância, RAID pode ser configurado em seis níveis diferentes. Muitas melhorias nas técnicas de utilização de discos envolvem a utilização cooperativa de múltiplos discos e os esquemas de RAID melhoram o desempenho e a confiabilidade do sistema de armazenamento armazenando dados de forma redundante, o espelhamento ou sombreamento mantém cópias de cada disco, a Paridade com blocos entrelaçados utiliza muito menos redundância.