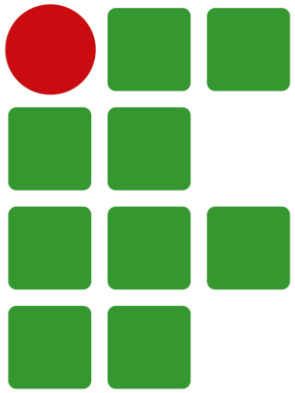

INSTITUTO FEDERAL
Pará

Campus Belém

Arquitetura e Organização de Computadores

Papel do desempenho

- Aula.04
- Curso Tecnologia em Análise e Desenvolvimento de Sistemas
- 1º semestre
- Prof. Mauro

19/11/2024

Objetivo

- Entender o que é desempenho em sistemas computacionais e como diferentes fatores podem influenciá-lo. Além disso, aprender as principais métricas de desempenho, os fatores que afetam a performance e as técnicas de otimização.

INSTITUTO FEDERAL
Pará
Campus Belém

Medidas de Desempenho

- O desempenho de um sistema computacional pode ser avaliado com base em várias métricas.

As três mais comuns são:

➢Tempo de Execução:

O tempo de execução é o tempo total que um programa ou processo leva para ser executado, desde o início até a conclusão.

- **Exemplo:** Um programa que leva 10 segundos para processar 1000 registros.
- **Importância:** Reduzir o tempo de execução é fundamental para melhorar a experiência do usuário e aumentar a produtividade do sistema.

➢Throughput (Taxa de Transferência):

Se refere ao número de unidades de trabalho que um sistema pode processar em um determinado período de tempo. Pode ser medido em unidades como transações por segundo, requisições por segundo, ou dados processados (em MB ou GB).

•**Exemplo:** Um servidor web que pode processar 500 requisições por segundo.

•**Importância:** Quanto maior o throughput, mais tarefas o sistema consegue realizar em um dado intervalo, o que é essencial para sistemas que lidam com grandes volumes de dados ou tráfego.

➢Latência:

É o tempo necessário para que uma solicitação seja recebida, processada e a resposta seja enviada. É, em essência, o "tempo de espera".

Exemplo: O tempo que leva para uma requisição HTTP chegar ao servidor, ser processada e retornar ao usuário.

Importância: Baixa latência é crucial para sistemas interativos, como jogos online, streaming de vídeo e aplicativos em tempo real.

Fatores que Afetam o Desempenho

- Vários fatores podem influenciar o desempenho de um sistema. Os mais comuns são:

Hardware

- O hardware em que o sistema está rodando tem um impacto direto no desempenho. Isso inclui o processador (CPU), memória (RAM), disco rígido (HD ou SSD) e a rede.
  - Exemplo: Um processador mais rápido ou mais núcleos pode reduzir o tempo de execução de tarefas pesadas.
  - Importância: Escolher a infraestrutura certa é crucial para garantir que o sistema tenha capacidade para lidar com a carga de trabalho.

Algoritmos e Estruturas de Dados

- A eficiência dos algoritmos e das estruturas de dados utilizadas também afeta diretamente o desempenho. Algoritmos com complexidade $O(n^2)$ podem ser muito mais lentos do que algoritmos com complexidade $O(n \log n)$.
  - Exemplo: Usar uma lista ordenada ($O(n \log n)$) para buscar um elemento em vez de uma busca linear ($O(n)$).
  - Importância: Escolher o algoritmo certo pode reduzir drasticamente o tempo de execução, especialmente em grandes volumes de dados.

Concurrency e Paralelismo

- A capacidade de executar múltiplas operações simultaneamente também pode melhorar o desempenho.
- Sistemas que suportam concorrência (vários processos simultâneos) e paralelismo (executando múltiplas tarefas ao mesmo tempo) conseguem aproveitar melhor os recursos do hardware.
  - Exemplo: Sistemas multicore podem executar processos em paralelo, diminuindo o tempo de execução de tarefas que podem ser divididas.
  - Importância: Maximizar a utilização dos recursos do sistema pode melhorar tanto o throughput quanto reduzir a latência.

INSTITUTO FEDERAL
Pará
Campus Belém

Capacidade de Armazenamento e Acesso a Dados

- O tempo que o sistema leva para acessar e recuperar dados pode ser um dos maiores gargalos de desempenho.
- O uso de discos SSD em vez de HDs tradicionais pode melhorar significativamente a velocidade de leitura e escrita de dados.
  - Exemplo: Leitura de dados de um banco de dados em memória (RAM) é mais rápida do que ler de um disco rígido.
  - Importância: Manter os dados em caches rápidos ou memória pode reduzir a latência e melhorar o throughput.

INSTITUTO FEDERAL
Pará
Campus Belém

Técnicas de Otimização

- Agora que entendemos as métricas e os fatores que afetam o desempenho, vamos explorar algumas técnicas para otimizar os sistemas.

Caching

- Caching é a técnica de armazenar dados temporariamente em um local de acesso rápido (como a memória) para evitar acessos repetidos a fontes mais lentas, como bancos de dados ou sistemas de arquivos.
  - Exemplo: Um servidor web pode armazenar o resultado de uma consulta a um banco de dados em memória, para que não precise repetir a consulta em todas as requisições.
  - Importância: O uso de caches pode reduzir significativamente a latência e aumentar o throughput de um sistema, pois evita repetidos acessos a fontes lentas de dados.

Pipelining

- Pipelining é uma técnica que permite que múltiplas operações sejam executadas de forma sobreposta, sem esperar que a operação anterior seja concluída completamente.
  - Exemplo: Em processadores, enquanto uma instrução está sendo executada, outra pode ser decodificada e uma terceira pode ser carregada na memória.
  - Importância: Pipelining pode melhorar o throughput ao permitir que o sistema execute múltiplas operações ao mesmo tempo, sem precisar aguardar a conclusão de uma tarefa antes de iniciar outra.

Pipelining - Funionamento

- Imagine uma linha de montagem:
  - enquanto uma tarefa está sendo executada em uma etapa, outra pode começar na etapa seguinte, sem precisar esperar a conclusão da tarefa anterior.
  - Isso aumenta a eficiência, reduzindo o tempo total necessário para processar múltiplas operações.
  - Em processadores, por exemplo, isso significa que várias instruções podem ser processadas em diferentes estágios de execução ao mesmo tempo, melhorando o desempenho geral.

INSTITUTO FEDERAL
Pará
Campus Belém

Pipelining - Funionamento

- Imagina um processador que precisa executar a instrução "A + B" em um programa. Sem *pipelining*, o processador faria as seguintes etapas, uma por vez:

1. **Busca (Fetch)**: Recupera a instrução "A + B" da memória.
2. **Decodificação (Decode)**: Decodifica a instrução para entender o que fazer (somar A e B).
3. **Execução (Execute)**: Realiza a soma de A e B.
4. **Escrita (Write)**: Armazena o resultado de A + B na memória.

INSTITUTO FEDERAL
Pará
Campus Belém

Pipelining - Funionamento

Agora, com *pipelining*, essas etapas são divididas e podem ser sobrepostas. Enquanto o processador está no estágio de execução (somando A + B), ele pode buscar a próxima instrução, decodificá-la, e assim por diante, sem esperar cada uma terminar completamente. O processador tem várias instruções em diferentes estágios do pipeline ao mesmo tempo.

1. **Ciclo 1**: Busca a instrução "A + B".
2. **Ciclo 2**: Decodifica "A + B" e busca a próxima instrução (por exemplo, "C - D").
3. **Ciclo 3**: Executa a soma "A + B" e decodifica "C - D".
4. **Ciclo 4**: Escreve o resultado de "A + B" e executa "C - D".

Load Balancing (Balanceamento de Carga)

- Balanceamento de carga distribui as requisições ou tarefas entre várias instâncias de um sistema para evitar que um único servidor ou recurso fique sobrecarregado.
  - Exemplo: Em um servidor web com alto tráfego, um balanceador de carga distribui as requisições entre diferentes servidores para garantir que nenhum servidor fique sobrecarregado.
  - Importância: O balanceamento de carga pode melhorar o throughput e garantir que o sistema continue responsivo mesmo durante picos de tráfego.

Compressão de Dados

- A compressão de dados pode ser usada para reduzir a quantidade de dados transmitidos pela rede ou armazenados no sistema, o que pode reduzir tanto o tempo de transferência quanto o uso de armazenamento.
  - Exemplo: Comprimir imagens ou arquivos antes de enviá-los pela rede pode reduzir a latência e aumentar o throughput de uma aplicação web.
  - Importância: A compressão é uma técnica útil especialmente para sistemas que lidam com grandes volumes de dados.

INSTITUTO FEDERAL
Pará
Campus Belém

Revisão

- Para garantir um bom desempenho em sistemas computacionais, é essencial medir e entender o impacto de métricas como tempo de execução, throughput e latência.
- Vários fatores, como hardware, algoritmos e acesso a dados, afetam essas métricas.
- Técnicas como caching, pipelining, balanceamento de carga e compressão de dados podem ser empregadas para otimizar o desempenho de sistemas.