Introdução

A computação distribuída refere-se a um modelo de computação no qual componentes de software ou hardware, distribuídos em diferentes computadores conectados em rede, colaboram para alcançar um objetivo comum. Nesse paradigma, tarefas podem ser divididas entre múltiplos dispositivos, permitindo processamento paralelo e aumentando a eficiência.

Este relatório descreve o projeto de um sistema de busca distribuída baseado em sockets, apresentando conceitos teóricos, arquitetura, algoritmo de busca e formatos de dados utilizados.

O que é Computação Distribuída

A computação distribuída é uma abordagem de desenvolvimento de sistemas na qual várias unidades de computação colaboram para realizar uma tarefa maior. Diferentemente da computação centralizada, onde todas as operações ocorrem em um único ponto, a computação distribuída divide a carga de trabalho entre diferentes nós (computadores), promovendo eficiência, escalabilidade e robustez.

Características

- **Descentralização:** Não há um único ponto de falha, pois várias unidades contribuem para o processamento.
- **Paralelismo:** Permite o processamento simultâneo de diferentes partes de uma tarefa.

Conceitos de Escalabilidade e Tolerância a Falhas

Escalabilidade

Escalabilidade refere-se à capacidade de um sistema em lidar com um aumento no volume de trabalho ou número de usuários. No projeto, o design permite a adição de novos servidores para processar mais dados sem afetar o desempenho do sistema.

Tolerância a Falhas

Tolerância a falhas é a capacidade do sistema em continuar operando corretamente mesmo na presença de falhas em um ou mais componentes. No projeto, a comunicação cliente-servidor é projetada para ser resiliente, detectando desconexões e tentando reconexões.

Vantagens e Desvantagens de Utilizar a Computação Distribuída

Vantagens

- **Desempenho Melhorado:** O processamento é dividido entre vários nós.
- **Confiabilidade:** O sistema pode continuar funcionando mesmo que um nó falhe.
- **Escalabilidade:** Novos servidores podem ser adicionados conforme necessário.

Desvantagens

- **Complexidade:** O desenvolvimento e manutenção são mais complicados devido à natureza descentralizada.
- **Latência de Comunicação:** Pode haver atrasos devido à troca de informações entre os nós.
- **Gerenciamento de Falhas:** É necessário um sistema robusto para lidar com falhas de componentes.

Arquitetura da Solução

Diagrama da Comunicação

(Client) --(Socket)--> (Server A) --(Socket)--> (Server B)

Formato dos Dados Trafegados

Os dados trafegados entre os servidores e o cliente utilizam o seguinte formato JSON:

Formato de Requisição:

{

"id": "5000"

}

Formato de Resposta:

{

"label": "astro-ph",

"title": "Weak Gravity Conjecture, Black Hole Entropy, and Modular Invariance"

}

Explicação do Algoritmo de Busca

1. O cliente envia um identificador (ID) ao Servidor A.
2. O Servidor A verifica o arquivo `data_A.json` para encontrar informações relacionadas.
3. Caso os dados não estejam no Servidor A, a requisição é encaminhada ao Servidor B.
4. O Servidor B consulta o arquivo `data_B.json` e retorna a resposta ao Servidor A.
5. O Servidor A combina as respostas locais e do Servidor B, enviando o resultado ao cliente.

Justificativa: O algoritmo de busca foi escolhido por sua simplicidade e eficiência na comunicação distribuída. Ele utiliza JSON como formato de troca de dados devido à sua leveza e ampla compatibilidade com diversas linguagens.

Referências Bibliográficas

1. Coulouris, G., Dollimore, J., & Kindberg, T. (2005). *Distributed Systems: Concepts and Design*. Pearson.
2. Tanenbaum, A. S., & Van Steen, M. (2017). *Distributed Systems: Principles and Paradigms*. Pearson.
3. "JSON - JavaScript Object Notation". Disponível em: https://www.json.org
4. "Socket Programming in Java". Oracle Documentation. Disponível em: https://docs.oracle.com/javase/tutorial/networking/sockets/index.html