Análise de Desempenho

MergeSort e QuickSort - RPC e Socket TCP

Centro de Informática UFPE
Universidade Federal de Pernambuco

Camila Xavier (cxm) e Vituriano Xisto (vox)

Objetivo

Comparar o desempenho de uma aplicação cliente/servidor usando dois mecanismos de comunicação diferentes (GoRPC e Socket). Os algoritmos implementados foram os de ordenação *QuickSort* e *MergeSort*.

Serviços do sistema

- Servidor GoRPC:
  - Serviço de ordenação de array com MergeSort;
  - Serviço de ordenação de array com QuickSort;
- Servidor Socket:
  - Serviço de ordenação de array com MergeSort;
  - Serviço de ordenação de array com QuickSort;
- Cliente GoRPC;
- Cliente Socket.

Métricas de Desempenho

- Tempo de execução de uma requisição ao serviço, medido no Cliente;
- Cliente executa N (10) invocações ao servidor.

Parâmetros

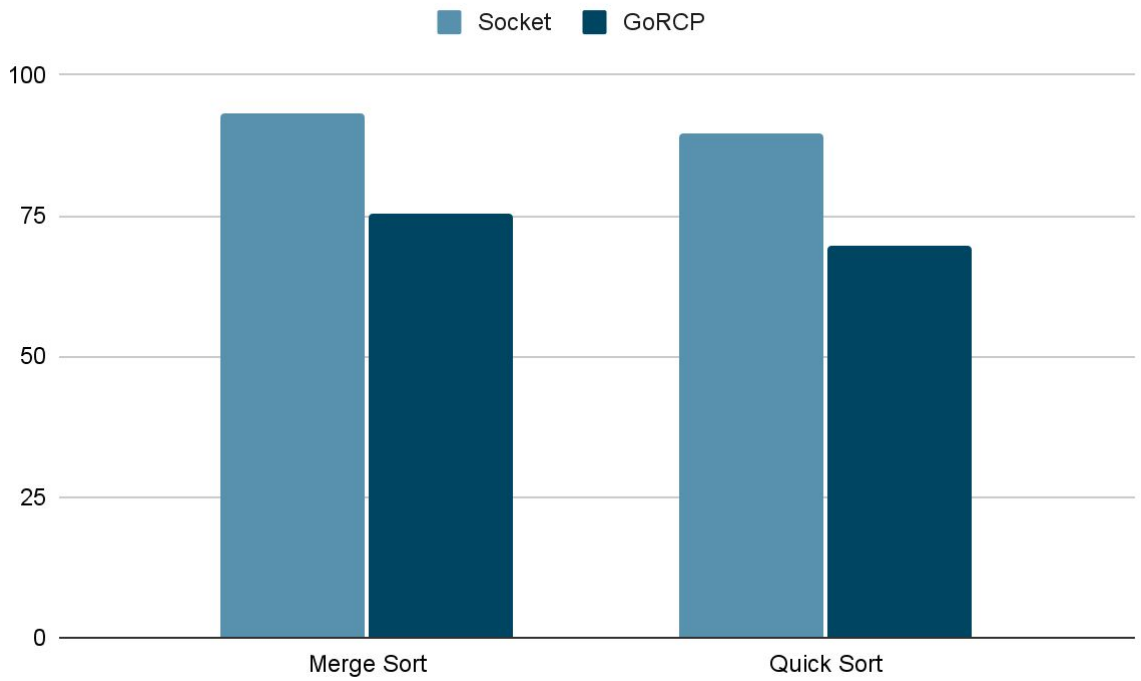
Parâmetro do Sistema	Valor
Hardware	MacBook M2 Pro, 16GB
Sistema operacional	macOS (Sonoma 14.1.1)
Linguagem de programação	Go
Interfaces de rede	Desligadas
Fonte de alimentação	Rede Elétrica
Processos em execução	Apenas os estritamente necessários à realização do experimento

Fatores

Fator	Nível
Algoritmo de Ordenação	MergeSort, QuickSort
Mecanismo de Comunicação	GoRPC, Socket

Resultados

- **N:** 10
- RTT Médio **Socket**:
  - Merge Sort: 93.214 µs
  - Quick Sort: 89.637 µs
- RTT Médio **GoRPC**:
  - Merge Sort: 75.477 µs
  - Quick Sort: 69.579 µs



Interpretação de resultados

Esses resultados indicam que o Quicksort tende a ser mais eficiente que o Mergesort em ambos os cenários, e que o uso de GoRPC reduz substancialmente o tempo de resposta médio em comparação com a comunicação via socket, tornando-se uma escolha preferível para aplicações que exigem alta performance de comunicação.