RPC - Computação Escalável

Fernanda Luísa Silva Gomes

Igor Patricio Michels

Laura Chaves Miranda

Marcos Antônio Alves

Professor: Thiago Pinheiro de Araújo

23 de maio de 2023

Solução arquitetural

Durante alguns dias, o grupo pesquisou sobre a implementação do stub servidor em C++ e não conseguiu progredir. Ocorreram erros até mesmo na execução do exemplo fornecido pela biblioteca *gRPC*. Assim, pela facilidade na implementação, optou-se por gerar os stubs cliente e servidor em Python.

As vantagens dessa abordagem são: simplicidade na implementação, maior material de apoio e possibilidade de manter esse código no próximo trabalho. Já as desvantagens são: necessidade de configurar e acessar o banco de dados em Python e em C++ e possível demora na exibição das análises, pois dependemos de um banco de dados além do RPC.

Decisões de projeto

A seguir estão as principais decisões tomadas.

- Utilizou-se *subprocess* na paralelização das simulações realizadas pelo mock. Isso permite que sejam executadas várias simulações ao mesmo tempo sem ter que abrir diversos terminais.
- No trabalho desenvolvido na A1, o sistema legado que consultava as informações do carro (nome, modelo e ano) utilizava um arquivo gerado pelo mock, em Python. Como nesse avanço as consultas são realizadas pelo servidor, essas informações também foram enviadas para o servidor via RPC.
- O grupo não conseguiu estabelecer a conexão com o banco de dados pelo C++. Para que fosse possível realizar os testes, decidiu-se armazenar os dados recebidos pelo servidor Python em arquivos. Assim, o stub servidor gera arquivos similares aos elaborados na primeira parte do

trabalho, mas com os dados recebidos via RPC, e o C++ irá manter seu comportamento, lendo os arquivos e processando os mesmos.

- Por fim, para possibilitar os cálculos de tempo total de processamento foram feitas pequenas adaptações nas structs elaboradas para a A1, bem como na saída das funções, juntando todos os resultados para serem impressos numa mesma função. Além disso, foi alterado o programa para que as threads se valham de toda a capacidade computacional do computador.

A implementação final pode ser encontrada aqui.

Resultados

Os resultados obtidos, com o tempo total de processamento desde a emissão até o final do processamento, pode ser observado na Figura 1.

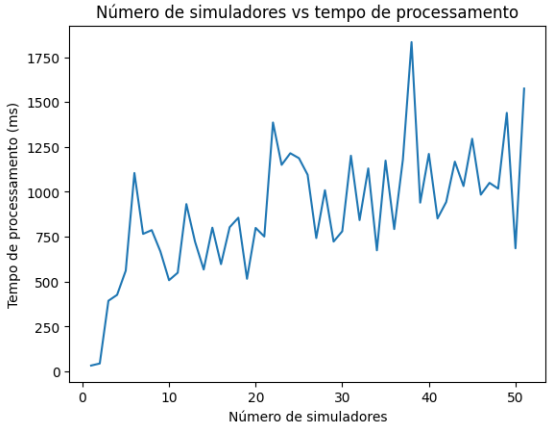

Figura 1: Resultados observados.

Podemos notar uma tendência crescente conforme o número de simulações aumenta. Esse comportamento se dá pelo aumento das operações de I/O em virtude do aumento no número de simuladores rodando simultaneamente.

Além disso, ocorreram alguns problemas durante o teste do servidor. Por estarmos trabalhando com várias máquinas ao mesmo tempo, com o tempo de emissão sendo tomado em uma máquina e o de cálculo em outra, há anomalias que não foram tratadas, pensando numa melhor simulação de "vida real". Essas anomalias se caracterizam por meio de tempo final negativo, ocorrido quando o horário do servidor está adiantado em relação ao do cliente. A opção por não tratar essas anomalias se deu pelo entendimento que o objetivo era a forma do gráfico, com sua tendência crescente.