RPC - Computação Escalável

André Costa
Emanuel Bissiatti
João Lucas Duim
Rafael Portácio
Victor Bombarda

22 de Maio de 2023

1 Introdução

Com o uso do gRPC no sistema operacional Fedora. Desenvolvemos um sistema de mensagens em que um programa escrito em Python envia uma string em formato de JSON para um servidor em C++ que será enfileirada numa queue e convertida para um objeto JSON. Foram feitas modificações no código Python para que ele obtivesse os dados do simulador e os enviasse para o servidor por meio do gRPC. O servidor em C++ foi criado para ficar com a porta aberta e aguardar a chegada de mensagens dos clientes.

2 Modelagem

Em resumo, nossa modelagem da primeira avaliação correspondia na troca de arquivos JSONs, salvos em disco, entre o python e o C++. Entretanto, isso não era tão eficiente. Dessa forma, optamos por desenvolver um sistema de troca de mensagens eficiente entre o python como cliente e o C++ como servidor por meio do gRPC.

3 Instalação e problemas

Sabíamos que instalar o gRPC no C++ não seria uma tarefa fácil, mas pareceu quase impossível. A princípio, tentamos instalar a biblioteca no Ubuntu LTS, entretanto, o CMAKE era incapaz de compilar o simples Hello World do gRPC. Passamos alguns dias quebrando a cabeça com isso até que, por sorte, um dos integrantes tinha o Fedora Linux instalado e seguindo o mesmo tutorial do Ubuntu foi possível instalar. Entretanto, a nossa alegria durou pouco, o gRPC fora dos exemplos do gRPC não compila em nenhuma hipótese. Estávamos quase desistindo, até que surgiu a brilhante ideia de transformar a única coisa que funcionava (o exemplo do gRPC para o C++) em nosso servidor. Assim, iniciou-se de fato o desenvolvimento.

4 Desenvolvimento

Para iniciar, implementamos um sistema de mensagens tal que o programa em Python atua como cliente. Assim, carregamos os arquivos gerados pelo gRPC para o antigo simulador.py, modificando-o para salvar como cliente.

No lado do servidor, desenvolvemos uma aplicação em C++ que utilizava o gRPC para receber as mensagens dos clientes. O servidor ficava com a porta aberta e aguardava a chegada de novas mensagens. Ao receber uma mensagem, o servidor adicionava as mensagens em uma fila, que seriam processadas posteriormente por threads dedicadas. Com isso, conseguimos uma comunicação rápida e direta entre o cliente e o servidor.

Além disso, houve a necessidade de modificar o processo de Extração, Transformação e Carga (ETL) para que ele processasse as informações provenientes dessa lista de mensagens em ordem de chegada. Isso garantiu que os dados fossem tratados sequencialmente, evitando problemas de concorrência e inconsistências nos resultados.

5 Conclusão

Após a dificuldade incial de instalar e compilar o gRPC no Fedora conseguimos avançar com boa agilidade na conversão. Não foi preciso modificar a lógica do simulador nem a do ETL, apenas tivemos que cuidar que a funções de leitura do ETL trabalhassem com o novo sistema de mensagens.

Além disso, como era desejado, o programa Python passou a enviar os arquivos em formato string para o servidor em C++ de forma mais eficiente, sem o gargalo do disco. E, como o ETL já estava trabalhando com paralelismo, não houve perda de desempenho ao dedicar uma thread para manter o servidor gRPC ativo. Logo, acreditamos que concluímos o trabalho com sucesso.