Escola de Matemática Aplicada
Fundação Getulio Vargas

Projeto final - Mini-Waze

Fernanda Luísa Silva Gomes
João Lucas Duim
João Pedro Giordani Donasolo
Professor: Jorge Luís Poco Medina

28 de junho de 2021

Resumo

O relatório visa descrever o projeto final da disciplina de Estruturas de Dados e Algoritmos da Escola de Matemática Aplicada, FGV. O objetivo do projeto é implementar um Mini-Waze que calcula o menor caminho entre dois pontos do Rio de Janeiro informados pelo usuário.

Desenvolvimento

Para o desenvolvimento do web-aplicativo, a Fernanda e o João Pedro ficaram responsáveis pelo back-end, implementado em C++, e o João Lucas pelo front-end, implementado em HTML e JavaScript. O João Pedro baixou os dados, com auxílio da biblioteca *OSMnx* do Python, serializou os dados, implementou a fila de prioridade e estabeleceu a conexão entre o back e o front-end. A Fernanda implementou os algoritmos e escreveu o relatório. O João Lucas foi responsável pela

implementação da interface, ou seja, exibir o mapa, marcar os pontos de origem e destino informados pelo usuário e mostrar o menor caminho entre os nós informado pelo back-end.

Com o objetivo de explicar o projeto de modo simples e prático, o grupo gravou um vídeo apresentando o projeto e o desenvolvimento obtido. O vídeo pode ser obtido aqui. O código referente ao projeto pode ser encontrado aqui. O repositório possui três pastas iniciais: data, scr e test. Na pasta data, encontram-se os dados usados no web-aplicativo e o arquivo Python utilizado para baixar os dados. Na pasta scr, estão os códigos, separados em .vscode, back-end e front-end. Já na pasta test, temos um arquivo Python que realiza testes aleatórios para o algoritmo Dijkstra e A*. O script imprime no console o tempo médio e o desvio padrão dos tempos gastos pelos algoritmos para retornar o resultado.

Para executar o código, é necessário ter as bibliotecas microhttpd e jsoncpp de C++. Maiores detalhes de como rodar o código, obtendo a interface ou pelo console, estão no README do GitHub.

O grupo baixou e serializou os dado. Definiu uma classe Node, no C++. A classe possui como propriedades um id, a latitude e a longitude e um ponteiro apontando para um vetor. Nesse ponteiro, é possível obter o id, latitude, longitude e tamanho da rua que conecta os nós. Implementaram os algoritmos Dijkstra e A*, utilizando-se como fila de prioridade uma estrutura heap. De modo a simplificar o código e não implementar dois algoritmos similares, o algoritmo que calcula o menor caminho recebe como argumento um booleano que informa se o algoritmo a ser utilizado é o Dijkstra ou o A*.

Enquanto os integrantes responsáveis pelo back-end desenvolviam os algoritmos e a fila de prioridade, o membro responsável pelo front-end construía a interface. Inicialmente, mostrou-se o mapa da cidade do Rio de Janeiro e colocou marcadores para que o usuário clicasse nos pontos de origem e destino e informasse qual algoritmo gostaria que fosse utilizado. Posteriormente, trabalhou-se para que o front-end recebe um arquivo Json e exibisse o caminho entre os pontos marcados. Por último, a interface e o back foram integrados.

Resultados

O repositório que contem o desenvolvimento do trabalho possui 26,6 MB. Para saber o tempo de execução do algoritmo, realizaram-se testes aleatórios escolhendo-se dois nós do conjunto baixado com mais de 240 mil nós. Na Figura 1, temos dois plots. O primeiro é referente ao tempo de execução do Dijkstra e o segundo, ao A*. dados serializados: 8 mega

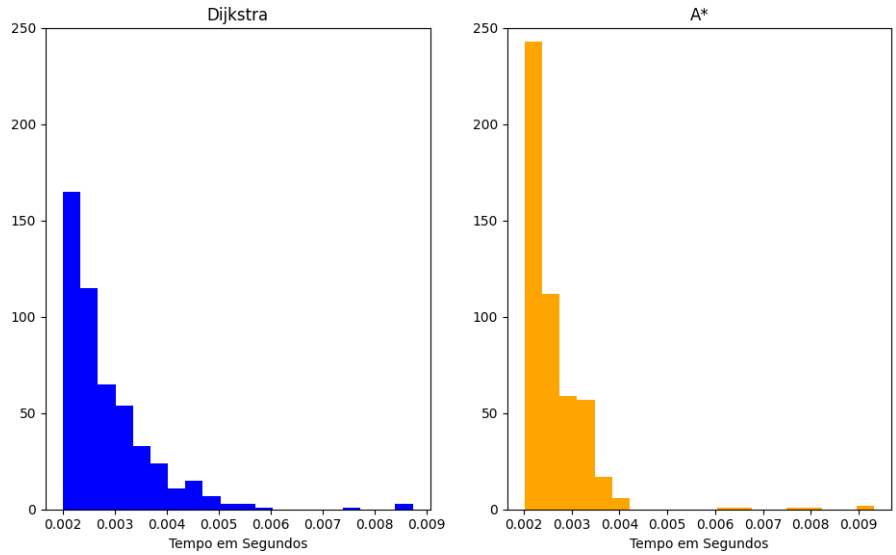

Figura 1: Testes aleatórios para os algoritmos Dijkstra e A*

Observa-se que ambos os algoritmos demoraram pouco tempo para retornar o resultando, sendo inferior a um décimo de segundo. Isso se deve a serialização realizada e a estrutura de dados utilizada na implementação.

Adicionalmente, nota-se que o algoritmo A* é mais rápido, como o esperado. Ao atualizar a distância dos nós, o A* adiciona uma heurística que corresponde a distância real entre os pontos no mapa.

Limitações e possíveis melhorias

O aplicativo possui limitações. Como trabalhou-se somente com os dados da Cidade do Rio de Janeiro, se o usuário pedir a rota entre dois nós em que um deles está a mais de 200 metros de um nó tratável, o usuário recebe a mensagem que não é possível calcular o menor caminho. Só é possível definir os pontos de origem e destino através de cliques na tela, sendo o primeiro clique a origem e o segundo o destino. Seria interessante poder digitar o endereço ou o nome do local desejado, por exemplo “FGV”. Além disso, para retirar os marcadores, a página é recarregada. Em relação ao back-end, o grupo utilizou heap como fila de prioridade, sendo fibonacci heap a melhor estrutura de dados para o algoritmo Dijkstra. Entretanto, devido a performance do aplicativo, acredita-se que

essa mudança não tenha sido prejudicial.

Desse modo, para evoluções e trabalhos futuros, a fila de prioridade pode ser alterada de heap para fibonacci heap, tornando o algoritmo mais eficiente. Pode modificar a interface de modo que seja possível digitar o endereço do local de origem e destino, bem como retirar os marcadores de outro modo sem ser recarregar a página. Esses aprimoramentos resultam em um web-aplicativo mais versátil e dinâmico, melhorando a experiência do usuário.

Conclusões

Por meio do projeto, foi possível desenvolver e melhorar diversas habilidades. Ampliou-se o conhecimento de diferentes linguagens de programação, tendo contato com C++, JavaScript e HTML. Pode-se aprender a lidar com um grande volume de dados, sendo similar a grandes aplicativos e projetos. Aprendeu-se a integrar o back-end e o front-end por meio de um servidor. Construiu-se uma interface similar ao Waze, sem algumas funcionalidades, que fornece o menor caminho entre dois pontos no mapa. Foram capazes de implementar algoritmos sabendo somente o modo como ele deveria funcionar.

Além de desenvolver habilidades técnicas, o projeto propiciou que os alunos melhorassem soft skills. Foi extremamente necessário o trabalho em grupo e colaboração. Ademais, os integrantes buscaram diversas fontes para resolver os problemas que tiveram ao longo do projeto.