Instituto Federal Catarinense
Campus Videira
Estrutura de Dados II

Projeto Final - Sistema Inteligente de Logística de Transporte

Aluno: Hygor Albert Fernandes Marques

junho
2025

Instituto Federal Catarinense
Campus Videira
Estrutura de Dados II

Projeto Final - Sistema Inteligente de Logística de Transporte

Relatório do Projeto Final da matéria Estrutura de Dados II do Curso de Ciência da Computação do Instituto Federal Catarinense Campus Videira, como requisito parcial para finalização da matéria.

Aluno: Hygor Albert Fernandes Marques

junho
2025

Conteúdo

1 Resumo

Os algoritmos estudados na disciplina de Estrutura de Dados II permitem soluções otimizadas para diversos contextos, sendo a logística de transportes um dos principais. Isso se deve ao fato de que grafos representam bem a estrutura de mapas reais, possibilitando a criação de algoritmos para encontrar o melhor caminho com base em diferentes critérios. Por exemplo, em uma situação onde o tempo é prioridade, o algoritmo busca a rota mais rápida, mesmo que mais cara; em outro caso, pode-se priorizar segurança ou economia, aceitando um trajeto mais longo. Neste trabalho, foi desenvolvido um simulador de sistema logístico de entregas que, embora não represente uma solução completa, demonstra de forma funcional o comportamento de algoritmos de decisão de rotas aplicados sobre grafos. O mapa é gerado ao iniciar o programa, com cidades conectadas por caminhos fictícios de diferentes tamanhos e pesos. A partir da seleção de origem e destino, o algoritmo traça a melhor rota, que é então exibida visualmente na interface gráfica construída com a biblioteca Qt. Em seguida, o usuário escolhe os itens da entrega, e o sistema calcula o custo e o tempo estimado com base em variáveis simplificadas, como distância e peso da carga, desconsiderando fatores mais complexos como clima, temperatura ou trânsito. O projeto utiliza conceitos de grafos, orientação a objetos e algoritmos clássicos de caminho mínimo, como Dijkstra, além de estruturas de apoio como árvores de decisão. O sistema tem como objetivo principal demonstrar, de forma prática, a aplicação das estruturas de dados estudadas na resolução de problemas reais.

2 Introdução

Sistemas logísticos modernos enfrentam desafios cada vez mais complexos, exigindo soluções eficientes para planejamento de rotas, previsão de custos e otimização de entregas. O uso de estruturas de dados avançadas e algoritmos de caminhos mínimos é essencial nesse contexto, pois permite representar cidades, conexões e variáveis operacionais de forma estruturada. A disciplina de Estrutura de Dados II oferece uma base sólida para compreender e aplicar esses conceitos em problemas reais. Neste trabalho, desenvolvemos uma aplicação que simula um sistema logístico de entregas entre cidades, com foco na escolha da melhor rota a partir de diferentes critérios. O projeto combina estruturas como grafos, árvores e algoritmos de decisão para oferecer soluções variadas ao usuário, que pode optar por caminhos mais curtos, mais seguros ou mais econômicos, conforme o contexto.

3 Problema e Definição dos Algoritmos

O problema que nosso projeto visa resolver é a escolha do melhor caminho para uma entrega entre cidades, utilizando um sistema inteligente de transporte. Nossa empresa fictícia de entregas atende uma pequena região composta por cinco cidades — **Cidade A**, **Cidade B**, **Cidade C**, **Cidade D** e **Cidade E** — todas interligadas entre si.

Aplicamos quatro algoritmos e estruturas estudadas na disciplina:

- **Grafos**, para organizar as cidades e suas conexões.
- **Árvore**, para montar a estrutura da entrega com a rota e os produtos.
- **Dijkstra**, para buscar o menor caminho em termos de distância.
- **Variação de Dijkstra**, para buscar o caminho mais seguro.

O projeto foi desenvolvido em **C++**, utilizando a ferramenta **Qt (Cute)** para a interface gráfica. O planejamento inicial previa o uso de *pesquisa operacional* para decidir os melhores valores. No entanto, devido ao tempo, foi adotada uma simplificação com valores absolutos, que não levam em consideração variáveis como tipo de pneu, veículo, peso máximo, temperatura ou clima.

Consideramos apenas a distância entre cidades e o material das estradas. No mapa, temos dois tipos: **estrada de terra** e **estrada de asfalto**. A estrada de terra é mais curta, porém menos segura. Essa distinção serve para demonstrar as diferenças entre os algoritmos implementados.

3.1 Grafos

Os grafos são utilizados para representar as cidades e suas conexões no sistema. Cada cidade é representada como um vértice, enquanto as estradas que as ligam são as arestas do grafo, carregando informações como a distância e o tipo do pavimento.

3.2 Árvore

A árvore é usada para organizar as entregas e os produtos associados a cada rota de forma hierárquica. Nela, cada nó representa um conjunto de dados relacionados, como uma rota calculada ou um produto cadastrado, permitindo a estruturação dos registros de forma ordenada e eficiente para consulta e visualização. A árvore facilita o armazenamento das informações dos cálculos realizados, como distância, custo e tempo.

3.3 Algoritmo de Dijkstra

Esse algoritmo calcula o caminho mais curto entre os dois pontos, levando em consideração as distâncias das conexões entre as cidades. Ele começa atribuindo uma distância inicial infinita para todos os pontos, exceto a cidade de origem, que recebe distância zero. Em seguida, o algoritmo percorre o mapa, atualizando as distâncias para os pontos vizinhos sempre que encontra um caminho mais curto. No final, ele reconstrói o caminho ideal, calculando a distância total, o tempo estimado (com base na velocidade média) e o custo da entrega, que é proporcional à distância percorrida.

3.4 Varição de Dijkstra

Essa abordagem funciona de maneira semelhante ao Dijkstra, mas com uma diferença importante: ela pode aplicar regras específicas para escolher a rota, como evitar estradas de terra, que podem ser mais lentas ou prejudiciais para o transporte. Ao ignorar essas conexões, o sistema consegue sugerir uma rota que prioriza segurança ou qualidade do trajeto, mesmo que o caminho não seja o mais curto. O resultado também inclui a distância, tempo e custo estimados para a entrega.

4 Funcionamento do Sistema

O programa foi projetado para conter três telas principais: uma para o mapa, uma para os veículos e outra para as entregas. Nelas, o usuário poderia escolher a rota desejada, o veículo responsável pela entrega e, ao final, visualizar todas as informações relacionadas, incluindo o custo total.

Contudo, devido ao tempo limitado, o sistema foi simplificado para conter apenas duas telas: **Mapa** e **Entrega**. Assumiu-se, nesta versão, que qualquer veículo pode realizar qualquer rota, independentemente do tipo de carga transportada.

A implementação começa pela tela do mapa. Nela, o usuário pode selecionar cidades clicando diretamente sobre o mapa ou pesquisando pelo nome. Ao lado do mapa, há campos para inserir a cidade de origem e destino, além de um botão que aciona o cálculo da rota. Abaixo desses campos, são exibidas as informações resultantes: distância, custo e tempo da viagem.

Acima do mapa, o usuário pode escolher o algoritmo a ser utilizado para o cálculo da rota — **Dijkstra** ou **Dijkstra(Customizado)** — os quais determinam os valores exibidos nos campos mencionados. Essa escolha permite comparar os dois critérios distintos de decisão: menor distância ou maior segurança.

A geração do mapa é realizada por uma função principal que cria as cidades e desenha os caminhos entre elas. Essa função não recebe parâmetros; ela apenas realiza chamadas a funções auxiliares. Uma delas é responsável por criar as elipses representando as cidades e atribuir seus nomes; outra cria os caminhos (retos ou curvos). Essas funções utilizam coordenadas no *QGraphicsScene* do Qt e adicionam os elementos à *scene*, que é responsável por renderizar graficamente todos os objetos do sistema.

Após o mapa ser gerado, o usuário pode selecionar as cidades clicando nas elipses ou digitando os nomes nos campos. Essa seleção é gerenciada por uma função que torna as elipses interativas: ao serem clicadas, mudam de cor e têm seus nomes atribuídos aos campos de cidade de origem e destino. Uma pequena lógica garante que a cidade de origem seja sempre selecionada antes da cidade de destino, evitando inconsistências no cálculo.

Com as duas cidades selecionadas, o usuário pode clicar no botão de calcular rota. Isso aciona outra função auxiliar, semelhante à função de geração do mapa, porém voltada exclusivamente para o caminho calculado pelo algoritmo. Se o algoritmo escolhido for o de Dijkstra, o sistema calculará a menor distância possível entre as cidades, armazenando informações sobre cidades intermediárias para que o caminho visual seja corretamente traçado no mapa. Se o algoritmo selecionado for o de Árvore de Decisão, o cálculo considera o tipo de pavimento das estradas, evitando estradas de terra sempre

que possível. A rota calculada é desenhada com linhas ou curvas sobre o mapa, destacando o caminho seguido.

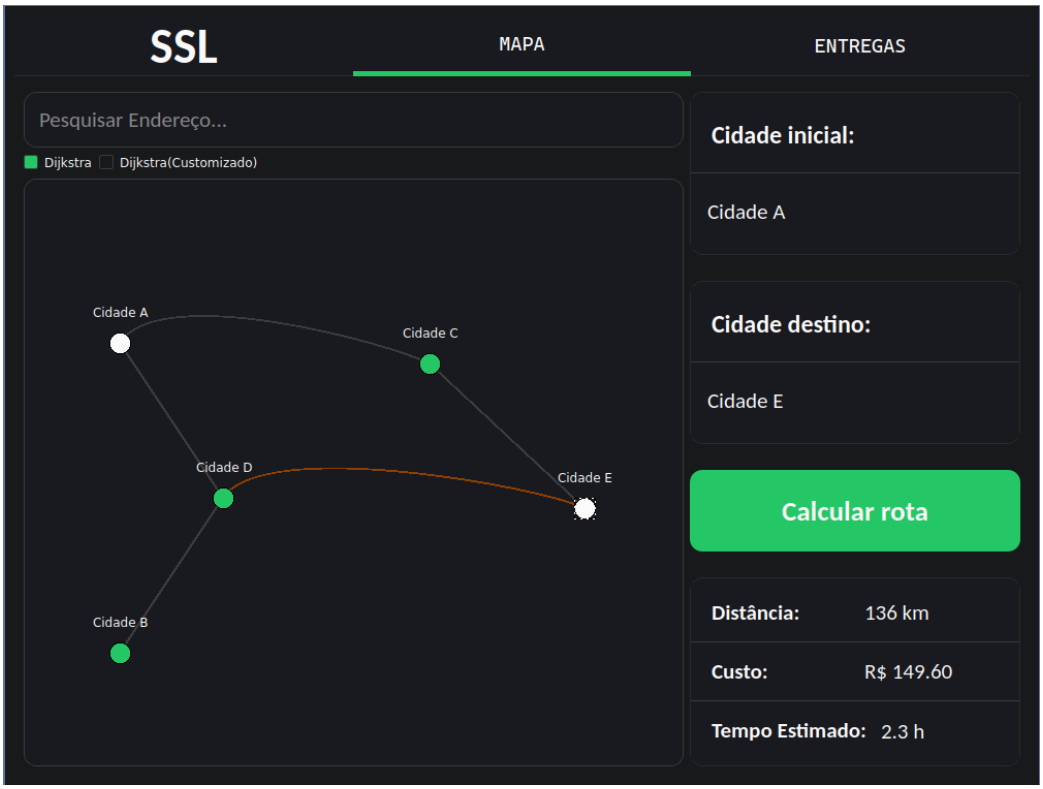

Figura 1: Tela Principal do Sistema Desenvolvido no Qt

Após o cálculo da rota, ela é adicionada à tela de entregas. Nela, é possível cadastrar um produto que será transportado, definindo seu nome e peso. Essa funcionalidade é simples, apenas para simular uma operação real. A tela de entregas permite calcular o total da entrega combinando os dados da rota com os do produto. Essa combinação é armazenada em uma árvore binária simples, que organiza os dados para visualização. A árvore permite armazenar múltiplas entregas e facilita a consulta e ordenação dos dados posteriormente.

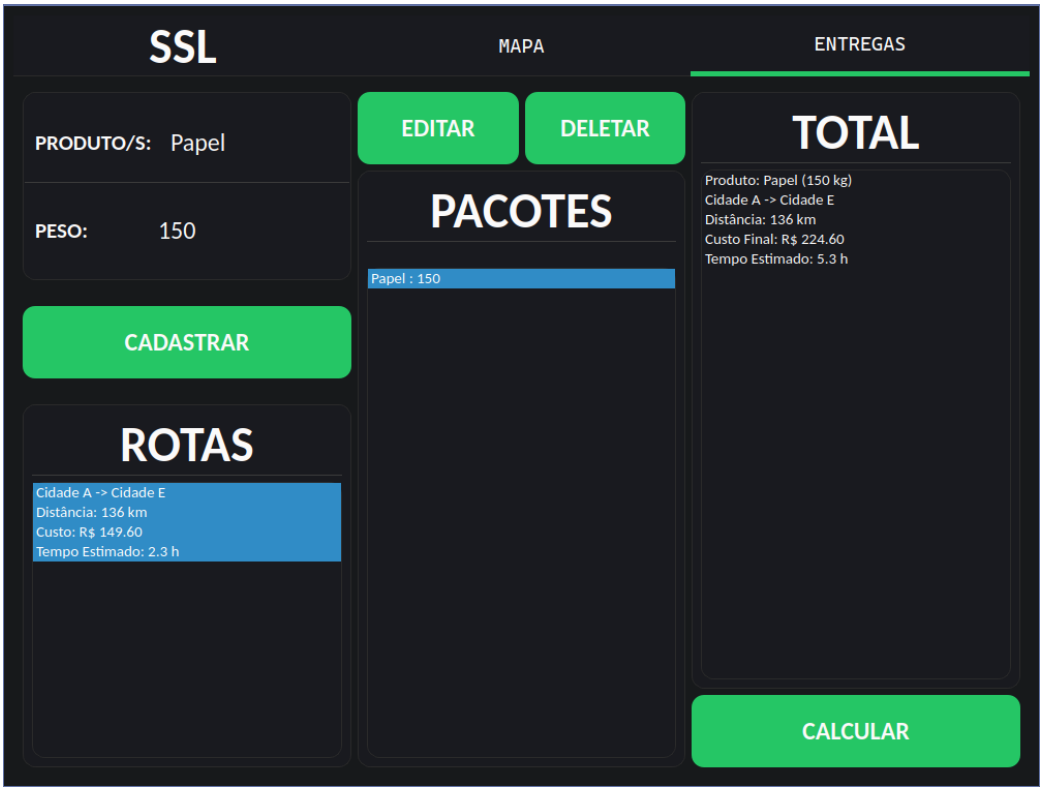

Figura 2: Tela Entregas do Sistema Desenvolvido no Qt

Esse é o processo completo de uma entrega no sistema. O usuário pode calcular várias rotas e cadastrar diversos produtos. A cada novo cálculo total, o campo é limpo e preenchido com os novos dados selecionados, permitindo que múltiplas entregas sejam simuladas de forma sequencial, de maneira clara e funcional.

5 Análise dos Resultados

Com todos os algoritmos e funções implementados, utilizando a biblioteca *Qt*, foi possível montar um sistema simples, com uma interface moderna e funcional. O objetivo principal — simular um ambiente de logística e observar o funcionamento integrado dos algoritmos — foi alcançado com êxito. A seguir, são apresentados alguns testes realizados e os resultados obtidos.

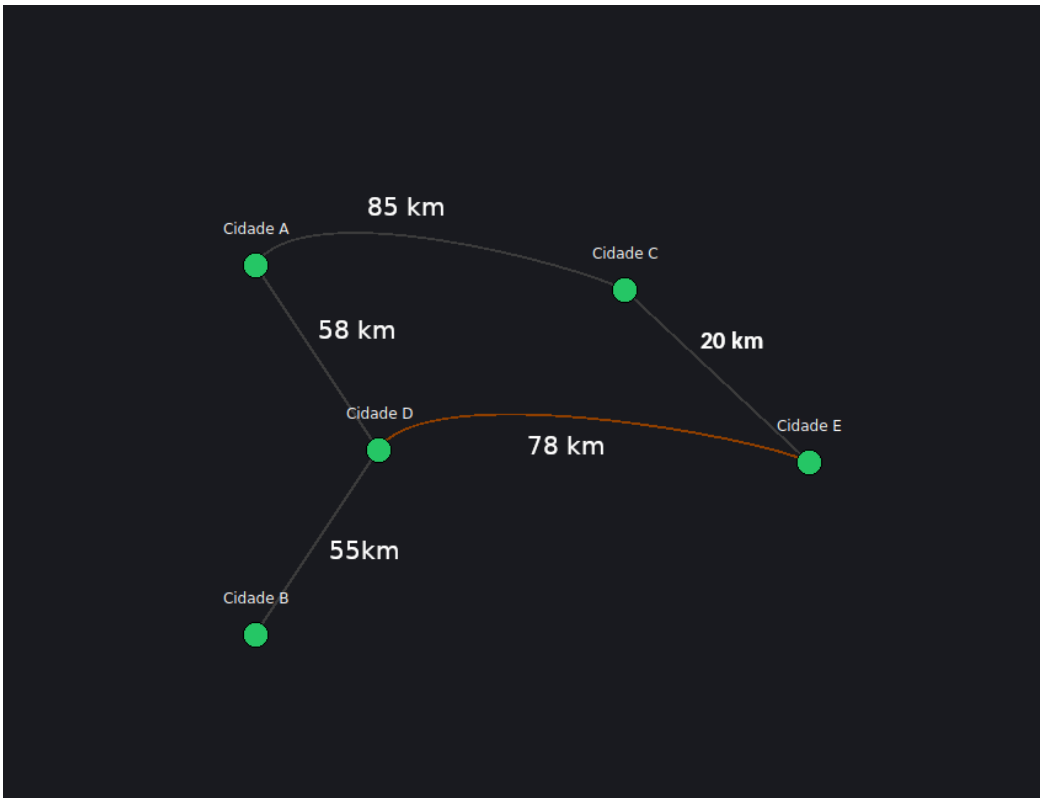

Figura 3: Mapa gerado com os pesos entre as cidades

Testes com estradas de terra permitidas (grafo completo)

Exemplo: Cidade A → Cidade E
Caminhos possíveis:

- Cidade A → Cidade D → Cidade E = 58 + 78 = **136**

- Cidade A → Cidade C → Cidade E = 85 + 57 = **142**

Melhor caminho: Cidade A → Cidade D → Cidade E = **136**

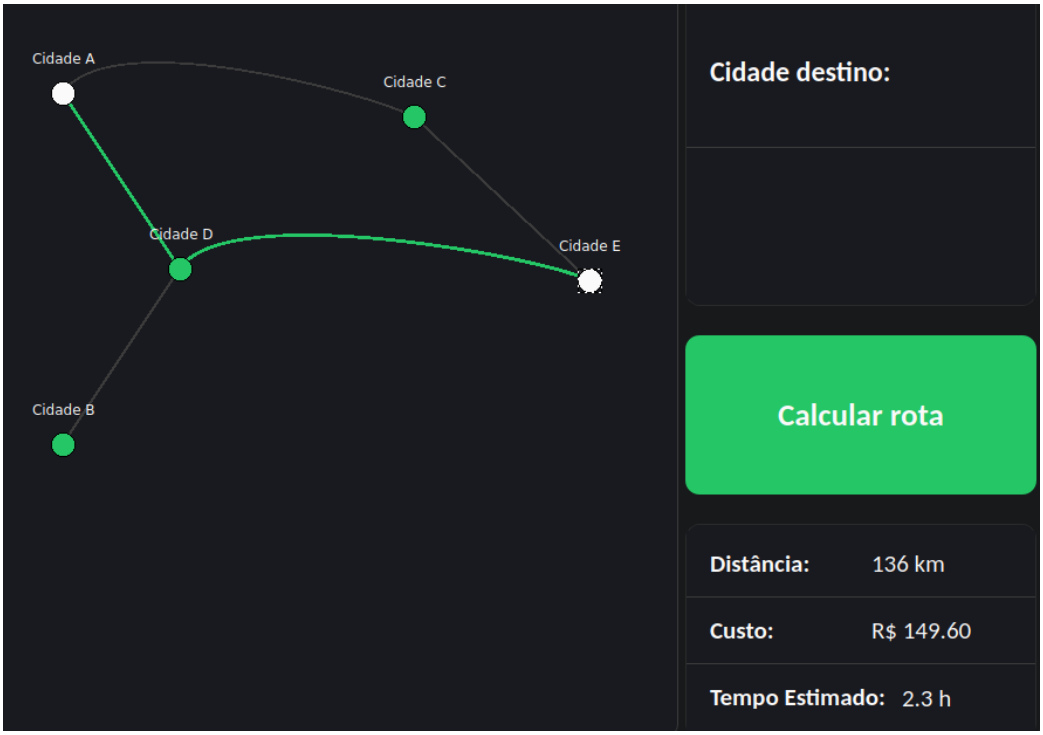

Figura 4: Caminho A → E com estradas de terra permitidas

Exemplo: Cidade B → Cidade E
Caminhos possíveis:

- Cidade B → Cidade D → Cidade E = 55 + 78 = **133**
- Cidade B → Cidade D → Cidade A → Cidade C → Cidade E = 55 + 58 + 85 + 57 = **255**

Melhor caminho: Cidade B → Cidade D → Cidade E = **133**

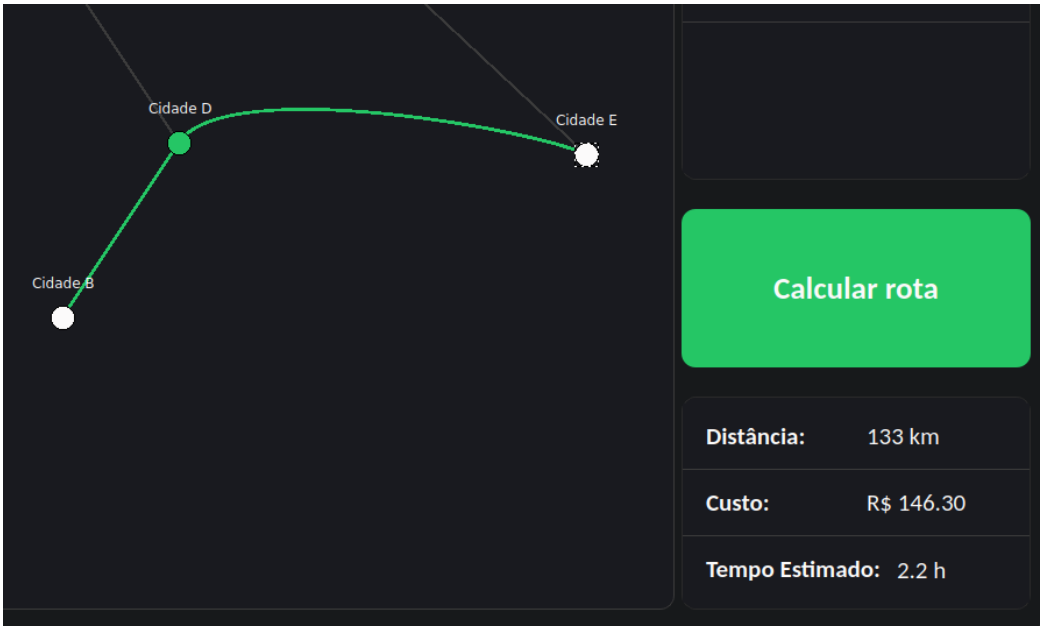

Figura 5: Caminho B → E com estradas de terra permitidas

Testes evitando estradas de terra

Neste cenário, foram desconsideradas as conexões que representam estradas de terra, como o trecho entre Cidade D e Cidade E. O sistema, ao utilizar o algoritmo de Árvore de Decisão, prioriza trajetos com pavimentação asfáltica.

Exemplo: Cidade A → Cidade E

Caminhos possíveis:

- Cidade A → Cidade C → Cidade E = 85 + 57 = **142**

Melhor caminho: Cidade A → Cidade C → Cidade E = **142**

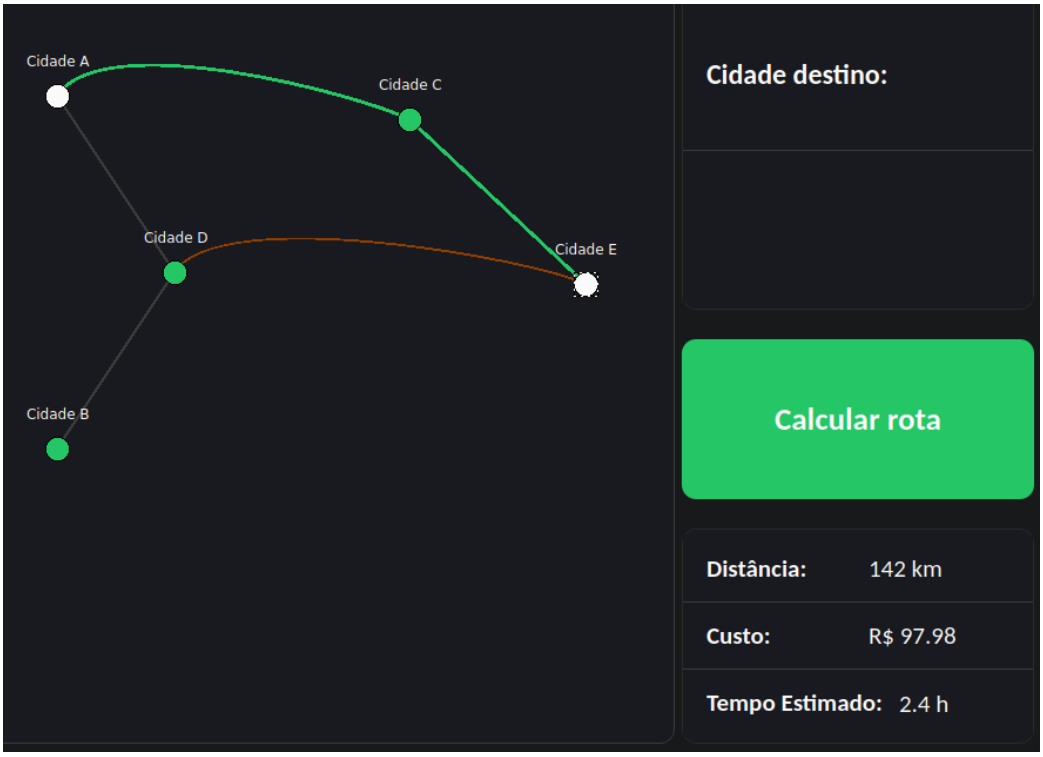

Figura 6: Caminho A → E evitando estradas de terra

Exemplo: Cidade B → Cidade E
Caminhos possíveis:

- Cidade B → Cidade D → Cidade A → Cidade C → Cidade E = 55 + 58 + 85 + 57 = **255**

Melhor caminho: Cidade B → Cidade D → Cidade A → Cidade C → Cidade E = **255**

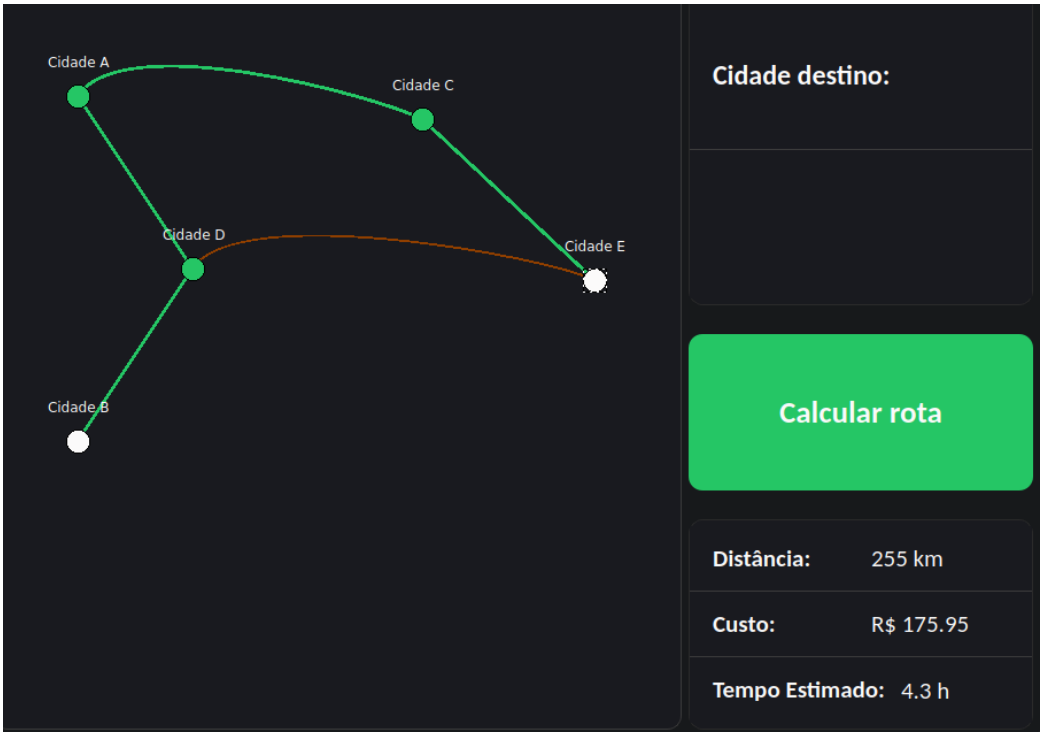

Figura 7: Caminho B → E evitando estradas de terra

6 Conclusão

O desenvolvimento deste sistema inteligente de logística de transporte possibilitou a aplicação prática e integrada dos conceitos abordados na disciplina de Estrutura de Dados II, especialmente o uso de grafos, árvores e algoritmos de caminho mínimo. Com a combinação entre a modelagem gráfica via *Qt* e os algoritmos de Dijkstra e de Dijkstra Modificado, o sistema foi capaz de calcular rotas eficientes ou seguras entre cidades, considerando diferentes critérios — como distância mínima ou qualidade do pavimento.

As funcionalidades desenvolvidas permitiram ao usuário interagir diretamente com o mapa, visualizar rotas calculadas dinamicamente e cadastrar produtos para simulação de entregas. Os testes realizados comprovaram o funcionamento correto do sistema tanto com estradas de terra permitidas quanto evitando-as, mostrando na prática como a escolha do algoritmo influencia o caminho final.

Embora o sistema tenha sido simplificado, assumindo, por exemplo, que qualquer veículo pode realizar qualquer entrega independentemente da carga, ele cumpriu seu objetivo principal: demonstrar a utilidade das estruturas de dados na resolução de problemas reais de forma visual, interativa e acessível.

A experiência obtida reforça a importância de uma boa modelagem dos dados e da escolha adequada dos algoritmos conforme o contexto. Em trabalhos futuros, o sistema pode ser expandido com a introdução de variáveis como capacidade de carga, tipos de veículos, condições climáticas e até restrições legais, além de técnicas de otimização da pesquisa operacional, visando soluções mais robustas e realistas.

Conclui-se, portanto, que o projeto foi bem sucedido, evidenciando o poder das estruturas de dados como ferramentas de modelagem e solução de problemas.