Colégio Técnico de Campinas
Universidade Estadual de Campinas
Departamento de Processamento de Dados

Estrutura de Dados II

Caminhos de Trem

2° Informática - Matutino - 2021

Fabrício Onofre Rezende de Camargo - 20130
Ligia Keiko Carvalho - 20143

Campinas
2021

Sumário

1. Introdução

O presente projeto visa o desenvolvimento de aplicação que auxilie o usuário a calcular os caminhos de trem das cidades de Portugal e Espanha utilizando estruturas aprendidas na disciplina de Estrutura de Dados II. Essas estruturas de dados possibilitam maior rapidez no processamento, neste projeto serão utilizados uma árvore e um grafo. Para o grafo será utilizado o método de *Dijkstra*, que calcula o menor trajeto de um grafo baseando-se nos pesos de cada aresta. E para armazenar os dados de vértice, será utilizada uma árvore binária.

2. Problema

Como desenvolver uma aplicação que permita calcular a rota mais curta entre cidades de Portugal e Espanha, além de adicionar cidades e rotas?

3. Objetivo

Utilizar os conhecimentos de estrutura de dados adquiridos durante a disciplina para desenvolver uma aplicação Windows Forms na linguagem de programação C#, que permita fazer a manutenção de cidades localizadas em Portugal e Espanha. Para realizar tal ação, será utilizado algoritmos e técnicas que que permitam facilitar e agilizar os processos. Tais quais como a partição de vetores para balanceamento da árvore binária, além do algoritmo de *Dijkstra.*

4. Desenvolvimento

- **23/11/2021**

Iniciamos o projeto, primeiramente adicionando e criando as classes necessárias, além de ajustar aquelas que já pegamos prontas. Também foram feitos os métodos para ler e gravar os registros dos arquivos textos e a opção de adicionar e remover cidades. Criou-se o design do formulário e o método "*Load.*"

- **26/11/2021**

Desenvolveu-se o método de inclusão de rotas entre as cidades na classe do formulário dentro do método "Load" da classe do formulário.

- **30/11/2021**

Nessa data foram feitos os métodos de ler os arquivos textos de cidades e rotas. Na classe de Cidade foi implementado o método para o ler o arquivo de cidades e criar os vértices do grafo, e um outro método para ler o arquivo com rotas.

- **01/12/2021**

Foi feita uma classe para desenhar a árvore que contém as cidades e uma alteração no método que lê os registros do arquivo de rotas para que ele insere uma cidade na árvore, o qual se localiza na classe “GrafoTrem”.

- **02/12/2021**

Percebeu-se que a classe “ArvoreDeBusca” já possuía um método para desenhar a árvore. Então foi trocado a PictureBox que mostraria o desenho da árvore para um Panel, para esse panel criou-se um evento do formulário, evento Paint, que chama o método “DesenharArvore” da classe “ArvoreDeBusca”. Ao desenvolver esse método, o desenho da árvore estava sendo mostrado no Panel, no entanto, a árvore sempre pendia para direita, pois o vetor estava ordenado. Para solucionar esse problema, teria que ser feita uma leitura particionada do vetor de cidades. Após particionar o vetor, o programa foi capaz de desenhar a árvore corretamente, assim como mostra a Figura 1.

Ao testar a busca do menor caminho entre duas cidades, o programa sempre retornava o erro indicando que o vetor que armazena os vértices estava *null.* Esse erro era apontado no método “Caminho”, o qual busca o menor caminho em grafo, e no método obtém o índice de um vértice.

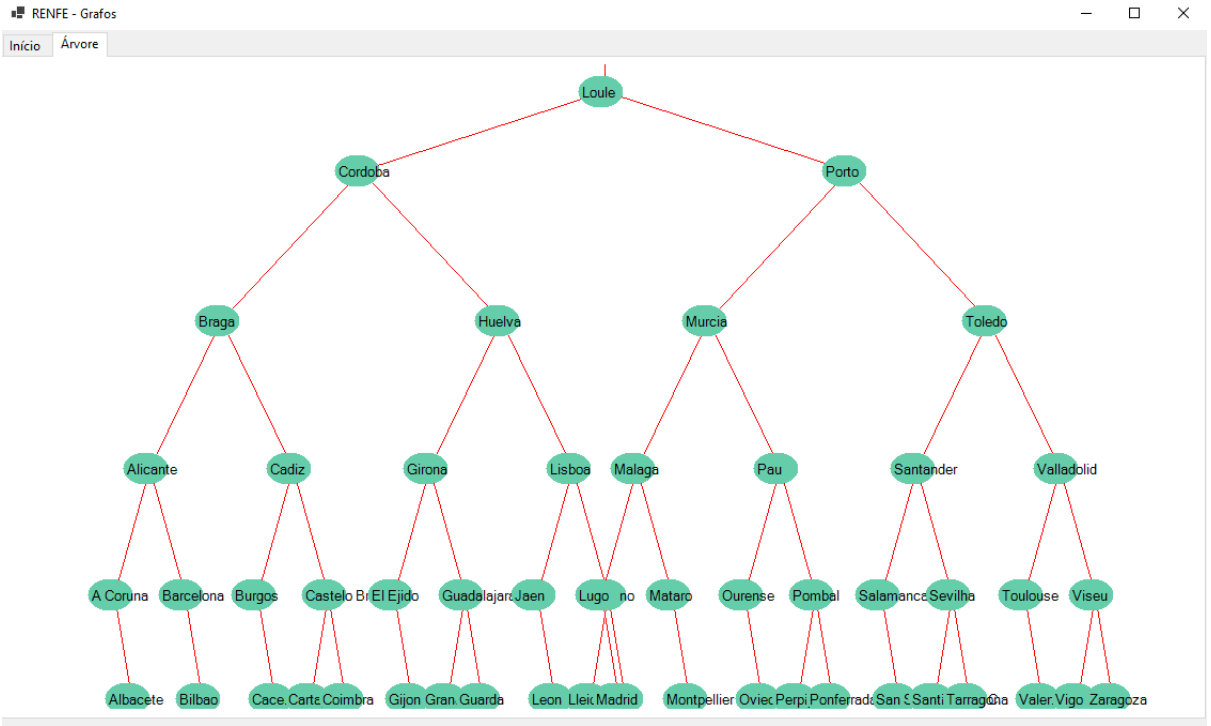

Figura 1. Desenho da árvore. Fonte: Autores.

- **03/12/2021**

Ao depurar o programa, observou-se que o vetor de vértices estava sempre nulo, pois um objeto da classe Grafo era instanciado mais de uma vez. Além desse erro, o método que obtém a posição de um vértice no vetor estava comparando o nome de uma cidade com outra, no entanto, o parâmetro da classe Cidade que guarda o nome da cidade possui espaços em branco no fim, dificultando a procura da cidade no vetor. Após consertar esses erros, o programa foi capaz de traçar o caminho mais curto entre duas cidades.

Também foi feito método que desenha as linhas do grafo em um PictureBox que possui o mapa de Portugal e Espanha. Para isso, foi colocado no método “ExibirPercurso” da classe Grafo uma pilha que guarda as cidades que estão na rota. Após isso, essa pilha é chamada em um método para desenhar as linhas nas coordenadas lidas do arquivo texto. Nesse método de desenhar o grafo, é lido as informações da cidade que saiu da pilha, e caso exista uma próxima cidade, variáveis auxiliares guardarão as informações da cidade anterior. Após obter a cidade de origem e a próxima cidade e suas respectivas coordenadas, uma linha é desenhada na PictureBox, assim como mostra a Figura 2.

Figura 2. Desenho do Grafo no mapa de Portugal e Espanha. Fonte: Autores

- **05/12/2021**

Desenvolvemos o método para excluir cidades. Além do método do formulário, a árvore deveria ser reorganizada e os índices atualizados, bem como o vetor. Uma grande dificuldade enfrentada durante a reorganização da árvore foi excluir um nó e depois substituí-lo por um filho, pois o método “ApagarNo” fazia com que a árvore duplicasse o filho direito para o nó que deveria ser excluído. Para corrigir o erro que o método originou, foi necessário alterar o método “ProcuraMenorDosMaioresDescendentes”, adicionando um *else* para caso a condição do nó direito ser diferente do sucessor.

Feito isso, iniciou-se o desenvolvimento do método que calcula o tempo, distância e preço das rotas. Para que isso fosse possível, criamos uma classe “InfoRota” a qual teria como parâmetros o preço, distância e tempo. Também foi alterada a classe “DistOriginal”, onde o construtor receberia um objeto do tipo “InfoRota”. Para que esses parâmetros fossem atribuídos às respectivas cidades, cada posição do vetor de “percurso” da classe Grafo possui um objeto de InfoRota. No entanto, ao executar a busca de rotas entre duas cidades, o programa era incapaz de traçar uma rota ou acaba excluindo uma cidade que fazia parte da rota certa.

- **06/12/2021**

Continuamos as mudanças na classe “Grafo” para que ela pudesse mostrar o preço, distância e tempo da rota. No dia anterior, em alguns métodos trocamos “infinity” por *false*, pois estávamos tentando comparar um atributo *bool* da classe “InfoRota”, para verificar se os objetos eram rotas. No entanto, isso acabou acarretando em mais erros, então voltamos para o valor de *infinity* e comparamos os valores com o atributo “Distancia” de “InfoRota”. Ao arrumar esse e alguns erros de lógica da classe “Grafo”, o programa passou a reconhecer as rotas normalmente, porém quando a rota era buscada mais de uma vez, o programa fazia a ligação direta entre a cidade de origem e a cidade de destino, ignorando as outras cidades que deviam fazer parte do caminho.

Para corrigir esse erro, fomos depurando o programa e anotando os valores de saída das variáveis, além de compararmos os valores de saída do programa que estava funcionando sem a matriz de “InfoRota”, para ser capaz de identificar onde o valor estava sendo alterado de forma errada. Além disso, também foi necessário arrumar os *getters* e *setters* da classe “InfoRota” e corrigir o método “AjustarMenorCaminho”, para que fosse possível ajustar o vértice pai, preço, tempo e distância do caminho.

5. Conclusão

Dado o que foi aprendido no semestre e durante o desenvolvimento desse projeto, foi possível criar uma aplicação Windows Forms, na linguagem C#, a qual consegue identificar a rota entre duas cidades utilizando o método de Dijkstra e uma árvore binária de busca capaz armazenar, inserir, excluir e buscar as cidades que estão armazenadas nela.

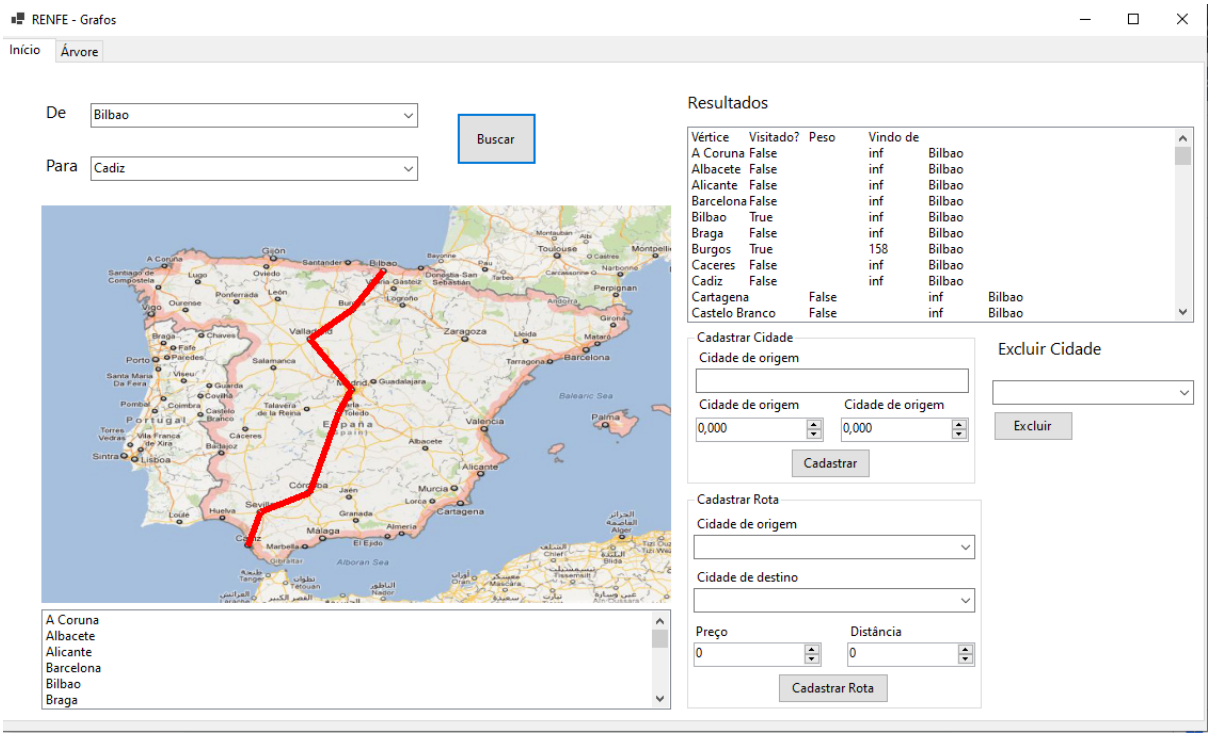

Figura 3. Programa finalizado. Fonte: Autores.