Estrutura de Dados

Atividade Avaliativa 03

Guilherme Altmeyer Soares, Igor Correa Domingues de Almeida

Cascavel, 20/03/2025

O código implementa um grafo não direcionado, onde os vértices representam locais e as arestas indicam conexões entre eles. Cada conexão tem três atributos: tempo, preço e distância. O objetivo do programa é encontrar o melhor caminho entre dois pontos, considerando cada critério separadamente, utilizando o algoritmo de Dijkstra.

A estrutura do grafo é definida pela classe grafo, que usa um vetor de listas adjacentes para armazenar as conexões. Cada aresta é representada por um par: o primeiro elemento é o vértice de destino, e o segundo é um objeto da classe node, que guarda as informações da conexão. O uso de listas de adjacência permite que o programa gerencie eficientemente as conexões entre os vértices, otimizando o tempo de busca por caminhos.

O método aresta adiciona conexões ao grafo, recebendo os vértices de origem e destino, além dos atributos tempo, preço e distância. Como o grafo é do tipo não direcionado, a conexão é registrada nos dois sentidos, garantindo que a busca por caminhos possa ocorrer independentemente da direção inicial escolhida.

Para calcular o caminho mais eficiente, o código implementa três variações do algoritmo de Dijkstra:

1. djikstrapreco(int source, int dest): encontra a rota com menor custo financeiro.
2. djikstratempo(int source, int dest): determina o trajeto mais rápido.
3. djikstradistancia(int source, int dest): busca a menor distância percorrida.

Essas funções utilizam uma fila de prioridade (priority_queue) para processar os vértices na ordem de menor custo acumulado. A cada iteração, os vizinhos do vértice atual são analisados e atualizados se for encontrado um caminho mais eficiente. Os pais dos vértices são armazenados para reconstruir a rota ao final. O uso da priority_queue garante que o vértice de menor custo seja sempre processado primeiro, otimizando a busca pelo menor caminho.

Cada variação do algoritmo de Dijkstra manipula um atributo diferente (preço, tempo ou distância) na avaliação dos caminhos. A estrutura de repetição percorre os vizinhos de um vértice, atualizando a distância acumulada se um caminho mais curto for encontrado. Dessa forma, ao final da execução, o vetor de distâncias armazena o custo mínimo para alcançar cada vértice a partir da origem.

A função main inicia o grafo com até 10 vértices e adiciona as conexões conforme os dados fornecidos. Após a inserção das arestas, o programa aguarda a entrada do usuário para definir os pontos de origem e destino. Para facilitar a entrada de dados, o programa converte letras maiúsculas e minúsculas em índices numéricos correspondentes. Após a conversão, as três versões do algoritmo de Dijkstra são chamadas em sequência, retornando os caminhos e os custos calculados.