Busca do Menor Caminho entre Bairros de São Luís usando Grafos e Algoritmo de Dijkstra

Antônio José Lobato Nogueira[1], Felipe Murilo Ribeiro Ribeiro[1]

Departamento de Sistemas e Computação
Instituto Federal do Maranhão (IFMA) – Campus Monte Castelo– São Luís – MA – Brazil

antonio.jose@acad.ifma.edu.br, felipe.murilo@acad.ifma.edu.br

Abstract. *This work models urban routes between São Luís neighborhoods as a weighted graph, with vertices (neighborhoods) and edges (roads in km). A generic Graph ADT (adjacency lists) and Dijkstra algorithm for shortest path were implemented. The system reads an input file and outputs the shortest path and distance for pairs like cohama–centro (13.7 km) and ipase–aeroporto (23.0 km). Results demonstrate applicability in urban route analysis.*

Resumo. *Este trabalho modela rotas urbanas entre bairros de São Luís como grafo ponderado, com vértices (bairros) e arestas (ligações viárias em km). Implementamos um TAD Grafo genérico (listas de adjacência) e o algoritmo de Dijkstra para caminho mínimo. O sistema lê arquivo de entrada e gera em saída o menor caminho e distância entre pares como cohama–centro (13.7 km) e ipase–aeroporto (23.0 km). Os resultados mostram a aplicabilidade em análise de rotas urbanas.*

1. Introdução

A determinação eficiente de rotas mais curtas entre pontos em uma malha viária é essencial para aplicações práticas como sistemas de navegação GPS, planejamento logístico e otimização de transporte urbano na cidade de São Luís, onde nos deparamos diariamente com desafios de mobilidade entre bairros como Cohama, Centro e Aeroporto. Neste trabalho, desenvolvido como atividade da disciplina Algoritmos e Estruturas de Dados II do IFMA Campus Monte Castelo, modelamos o problema de rotas entre 36 bairros da capital maranhense como um grafo ponderado não direcionado, representando vértices pelos bairros e arestas pelas ligações viárias com pesos em quilômetros.

Nosso objetivo principal foi implementar um Tipo Abstrato de Dados (TAD) Grafo de forma genérica e independente do domínio, utilizando listas de adjacência, e sobre ele aplicar o algoritmo de Dijkstra para resolver o problema do caminho mínimo de fonte única em grafos com pesos não negativos. O protótipo desenvolvido lê o arquivo entrada.txt contendo 66 conexões viárias e gera no saida.txt as menores rotas para casos práticos locais, como de Cohama ao Centro (13.7 km) e Ipase ao Aeroporto (23.0 km), validando a implementação em um cenário real da nossa cidade.

O restante deste artigo está organizado da seguinte forma: a Seção 2 apresenta a fundamentação teórica de grafos e Dijkstra; a Seção 3 detalha o problema real e sua modelagem; a Seção 4 descreve a implementação do TAD e do algoritmo em Java; a Seção 5 apresenta os experimentos com resultados obtidos; e a Seção 6 traz conclusões e sugestões para trabalhos futuros.

2. Objetivos

Objetivo principal: Implementar um TAD Grafo genérico baseado em listas de adjacência e o algoritmo de Dijkstra para resolver o problema do caminho mínimo de fonte única, aplicado à modelagem de rotas urbanas entre bairros de São Luís-MA, lendo conexões de arquivo de entrada e gerando rotas otimizadas em arquivo de saída.

Objetivos específicos:

- Modelar a malha viária local como grafo ponderado não direcionado (vértices = bairros, arestas = ligações em km).
- Desenvolver API genérica para grafos com operações de inserção/remoção de vértices e arestas.
- Implementar Dijkstra com PriorityQueue para calcular distâncias mínimas e reconstruir caminhos via mapa de pais.
- Validar com casos reais: rotas curta, média e longa entre bairros locais.

3. Fundamentação Teórica

Um grafo $G = (V, A)$ é composto por um conjunto finito de vértices V e um conjunto de arestas A que conectam pares de vértices, representando relacionamentos entre objetos. Neste trabalho utilizamos grafos ponderados não direcionados, onde cada aresta $(u, v) \in A$ possui um peso $p(u, v) \geq 0$ correspondente à distância em quilômetros entre bairros.

O problema do caminho mínimo de fonte única busca, dado um vértice de origem $s \in V$ o caminho mais curto de $a\ a\ v$ definido como $\delta(s, v) = min\{p(c): s \rightsquigarrow v\}$, onde $p(c)$ é a soma dos pesos das arestas no caminho c. O algoritmo de Dijkstra resolve este problema em grafos com pesos não negativos, utilizando uma fila de prioridade para sempre expandir o vértice u com menor distância estimada $d[u]$ e relaxar suas arestas adjacentes.

A complexidade temporal do Dijkstra com heap binário é $O((V + A)logV)$.

4. Descrição do Problema

O problema real abordado consiste em determinar o caminho mais curto entre pares de bairros de São Luís-MA em uma rede de ruas locais, simulando cenários práticos como planejamento de entregas, rotas de aplicativos de mobilidade ou navegação urbana. Cada bairro é modelado como vértice e cada ligação viária (rua ou avenida) como aresta ponderada pela distância real em quilômetros, lida diretamente do arquivo entrada.txt no formato bairro 1 e bairro 2 distância (exemplo: cohama cohafuma 3.2).

Por exemplo, o grafo gerado contém 36 vértices (bairros representativos como "cohama", "centro", "aeroporto", "ipase", "sao-cristovao", "divineia") e 66 arestas bidirecionais, refletindo a conectividade real da malha urbana da capital maranhense com densidade típica de redes de transporte (grafo esparso). O algoritmo de Dijkstra deve calcular, partindo de uma origem fixa, a menor rota e distância total até múltiplos destinos específicos, reconstruindo o caminho ótimo via estrutura de predecessores, sem

qualquer interação com usuário durante a execução, atendendo rigorosamente às restrições da atividade prática.

Esta modelagem permite simular consultas reais de roteamento, como "qual a rota mais curta de Cohama ao Aeroporto?" ou "melhor caminho de Ipase ao Centro?", gerando saídas padronizadas para análise posterior. A escolha de São Luís justifica-se pela familiaridade local dos autores e pela representatividade do grafo para cidades de porte médio brasileiro.

5. Implementação

A implementação do sistema foi guiada pela busca de eficiência e escalabilidade, qualidades fundamentais para a manipulação de redes urbanas complexas. O TAD Grafo foi desenvolvido em Java utilizando a estrutura de Map<String, List<Aresta>>. Isso garante que o acesso aos adjacentes de qualquer bairro ocorra em tempo constante ($O(1)$). A lógica de inserção de dados no grafo segue o procedimento abaixo:

```
None

inserirAresta(String u, String v, double peso)
      Se mapa_adjacencias não contém u: mapa_adjacencias.put(u, nova
Lista())
      Se mapa_adjacencias não contém v: mapa_adjacencias.put(v, nova
Lista())

      mapa_adjacencias.get(u).adicionar(nova Aresta(v, peso))
      mapa_adjacencias.get(v).adicionar(nova Aresta(u, peso))
Fim Procedimento
```

Usamos Dijkstra, que busca o menor caminho. Para otimizar a busca, utilizou-se uma PriorityQueue (fila de prioridade), garantindo que o algoritmo explore sempre o nó com a menor distância.

```
None

Algoritmo Dijkstra(Grafo G, Vértice origem)
    Para cada vértice v em G:
        distancia[v] ← ∞
        pai[v] ← nulo
```

```
    distancia[origem] ← 0
    FilaPrioridade Q ← inserir(origem, 0)

    Enquanto Q não estiver vazia:
        u ← extrair_minimo(Q)
        Para cada vizinho v de u:
            Se distancia[u] + peso(u, v) < distancia[v]:
                distancia[v] ← distancia[u] + peso(u, v)
                pai[v] ← u
                Q.atualizar_ou_inserir(v, distancia[v])
Fim Algoritmo
```

O algoritmo é o que permite a atualização dos custos de deslocamento entre os bairros de São Luís. Um diferencial relevante é a manutenção do mapa de pais (predecessores), estrutura que preserva a memória do trajeto e possibilita a reconstrução completa do itinerário para o usuário final.

A classe Main faz a leitura do arquivo entrada.txt, onde as 66 conexões viárias são processadas, seguida pelo cálculo das rotas para pares estratégicos como Cohama ao Centro e Ipase ao Aeroporto. Os resultados são exportados para o arquivo saida.txt, entregando dados prontos para uso sem necessidade de interação constante.

6. Experimentos e Resultados

Para validar a corretude e a eficiência da implementação do TAD Grafo e do algoritmo de Dijkstra, foram conduzidos experimentos utilizando a malha viária modelada a partir do arquivo entrada.txt, contendo 36 vértices (bairros de São Luís) e 66 arestas ponderadas bidirecionais, caracterizando um grafo esparso representativo de uma rede urbana real.

Os experimentos tiveram como objetivo verificar: (i) a capacidade do algoritmo em encontrar caminhos mínimos corretos, (ii) a reconstrução adequada dos trajetos por meio do mapa de predecessores, e (iii) o comportamento das distâncias em rotas de diferentes extensões.

Foram selecionados quatro cenários de teste, abrangendo rotas curtas, médias, longas e um caso aleatório, conforme apresentado na Tabela 1.

Tabela 1: Rotas mais curtas calculadas entre bairros de São Luís

Caso	Origem	Destino	Distância (km)	Caminho
Curta	cohama	cohafuma	3.2	cohama → cohafuma
Média	cohama	centro	13.7	cohama → cohafuma → calhau → renascenca → centro
Longa	ipase	aeroporto	23.0	ipase → centro → monte-castelo → joao-paulo → coroado → tiririca l → aeroporto
Teste aleatório	sao-cristovao	divineia	14.2	sao-cristovao → tiririca l → coroado → turu → divineia

Os resultados obtidos demonstram que o algoritmo de Dijkstra identificou corretamente os caminhos mínimos em todos os cenários testados, sem ocorrência de ciclos e com distâncias crescentes proporcionais ao número de arestas percorridas. Observa-se que rotas curtas utilizam apenas uma ligação direta, enquanto rotas mais longas exploram múltiplos vértices intermediários, refletindo o comportamento esperado em redes urbanas reais.

A reconstrução dos caminhos, realizada por meio do mapa de predecessores, confirmou a integridade das rotas calculadas, permitindo a geração completa dos itinerários desde a origem até o destino final. Esse aspecto é essencial para aplicações práticas como sistemas de navegação e planejamento logístico.

Do ponto de vista estrutural, o grafo apresenta baixa densidade, com razão aproximada $\frac{|A|}{|V|^2} \approx 0.05$, característica típica de redes viárias urbanas. Nesse contexto, o desempenho do algoritmo mostrou-se eficiente, beneficiando-se do uso de listas de adjacência e de uma fila de prioridade baseada em *heap*, o que garante complexidade temporal assintótica $O((V + A) \log V)$.

7. Conclusão

Este trabalho atendeu plenamente aos objetivos propostos, implementando com sucesso um TAD Grafo genérico baseado em listas de adjacência e o algoritmo de Dijkstra para resolução do problema de caminho mínimo em grafos ponderados não direcionados. A modelagem da rede de ruas de São Luís como grafo com 36 vértices e 66 arestas permitiu calcular rotas realistas e otimizadas, como Cohama-Centro (13.7 km) e Ipase-Aeroporto (23.0 km), gerando resultados consistentes com distâncias reais, sem ciclos e com complexidade temporal $O(V + A) \log V$ adequada para redes urbanas esparsas.

A implementação demonstrou flexibilidade da API genérica (inserirAresta, obterAdjacentes), permitindo manipulação independente de vértices e arestas sem referência ao domínio específico, e eficiência do Dijkstra com PriorityQueue para relaxamento guloso. Os experimentos sistemáticos validaram a corretude algorítmica (caminhos mínimos reconstruídos via mapa de pais) e praticidade para cenários reais de navegação GPS, logística de entregas e planejamento urbano local.

8. References

VIEIRA, Raimundo Osvaldo. "Grafos – Parte 01: Conceitos Básicos, DFS e BFS". Algoritmos e Estruturas de Dados II, Instituto Federal do Maranhão – Campus Monte Castelo, São Luís, 2026.

VIEIRA, Raimundo Osvaldo. "Grafos – Parte 02: Ordenação Topológica e Árvores Geradoras Mínimas (Prim/Kruskal)". Algoritmos e Estruturas de Dados II, Instituto Federal do Maranhão – Campus Monte Castelo, São Luís, 2026.

VIEIRA, Raimundo Osvaldo. "Grafos – Parte 03: Problema do Caminho Mínimo e Algoritmo de Dijkstra". Algoritmos e Estruturas de Dados II, Instituto Federal do Maranhão – Campus Monte Castelo, São Luís, 2026.

CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford. Algoritmos: Teoria e Prática. Rio de Janeiro: Elsevier, 2012.

ZIVIANI, Nívio. Projeto de Algoritmos com Implementações em Pascal e C. 3. ed. São Paulo: Cengage Learning, 2011.