ACD

Grafos

DAVI LEMOS E HEDUARDO WITKOSKI

Introdução

Grafos são estruturas de dados compostas por vértices e arestas, representando relações ou dependências em conjuntos de dados.

- Suportam algoritmos eficientes para diferentes finalidades, por exemplo:
  - Busca: BFS (Busca em Largura) e DFS (Busca em Profundidade).
  - Ordenação: Ordenação Topológica.
  - Árvore Geradora Mínima: Algoritmos de Kruskal e de Prim.
  - **Caminho Mínimo (origem única): Algoritmos de Bellman-Ford e de Dijkstra.**
  - Caminho Mínimo (todos os pares): Algoritmo de Floyd-Warshall.
  - Fluxo Máximo: Algoritmo de Ford-Fulkerson.

Algoritmo de caminho mínimo de origem única

- Determinam o menor caminho de um vértice fonte para todos os outros em grafos ponderados, sendo usados em redes, logística e sistemas de navegação.
  - **Algoritmo de Bellman-Ford**: Permite arestas de peso negativo no grafo de entrada e produz uma resposta correta detectando a existência de ciclos negativos.
  - **Algoritmo de Dijkstra:** Supõe que todos os pesos das arestas no grafo de entrada são não negativos. Ex: mapa rodoviário.

Implementação de Grafos

- Representações computacionais que facilitam a manipulação de vértices e arestas, como lista e matriz de adjacência.
  - **Lista de Adjacência:** Armazena, para cada vértice, uma lista dos vértices adjacentes. É eficiente em memória e ideal para grafos esparsos.
  - **Matriz de Adjacência:** Usa uma matriz V×V para indicar conexões entre vértices. Consome mais memória e é eficiente para verificar a existência de arestas em grafos densos.

Objetivos

"Na implementação de estruturas de dados para Grafos, as representações de um grafo G por Lista de Adjacência (LA) e por Matriz de Adjacência (MA) proporcionam desempenhos semelhantes (em relação às médias de tempo de execução) para processamento de grafos densos (D) ou esparsos (E), especificamente na execução de implementações dos algoritmos de menor caminho de fonte única de Bellman-Ford e de Dijkstra?"

Objetivos

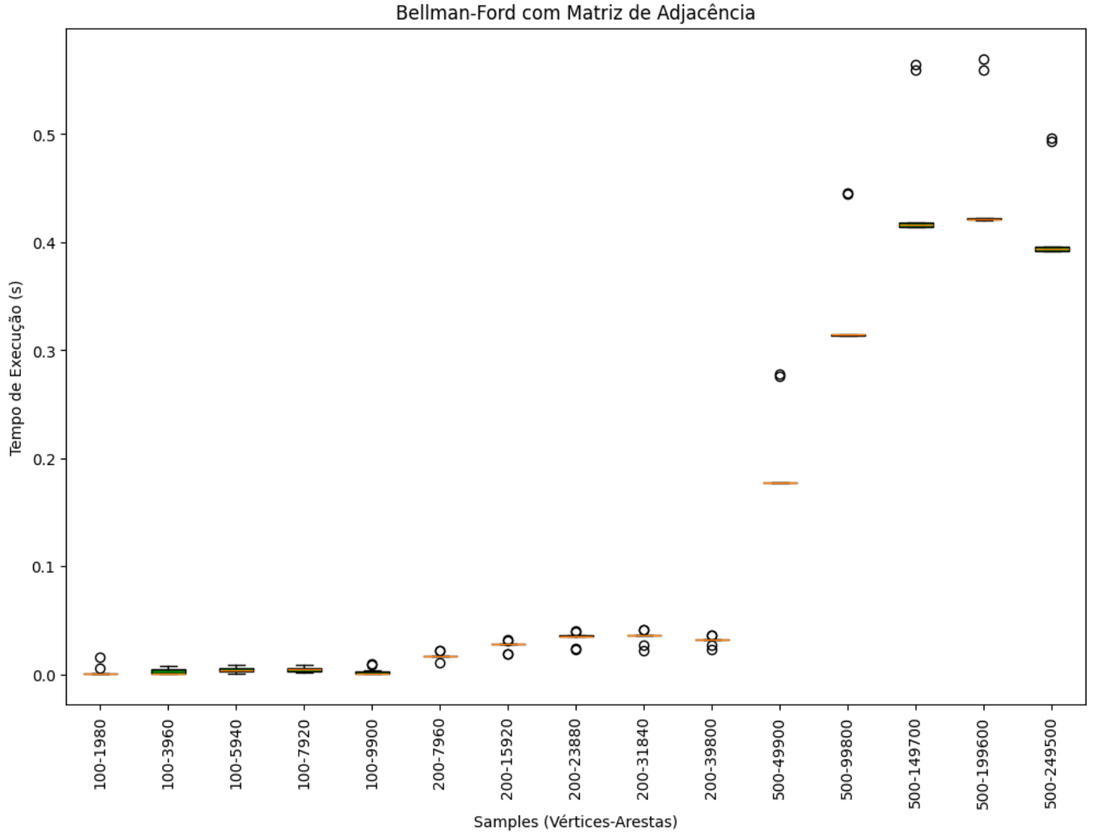
Avaliar e analisar, em termos de tempo de execução:

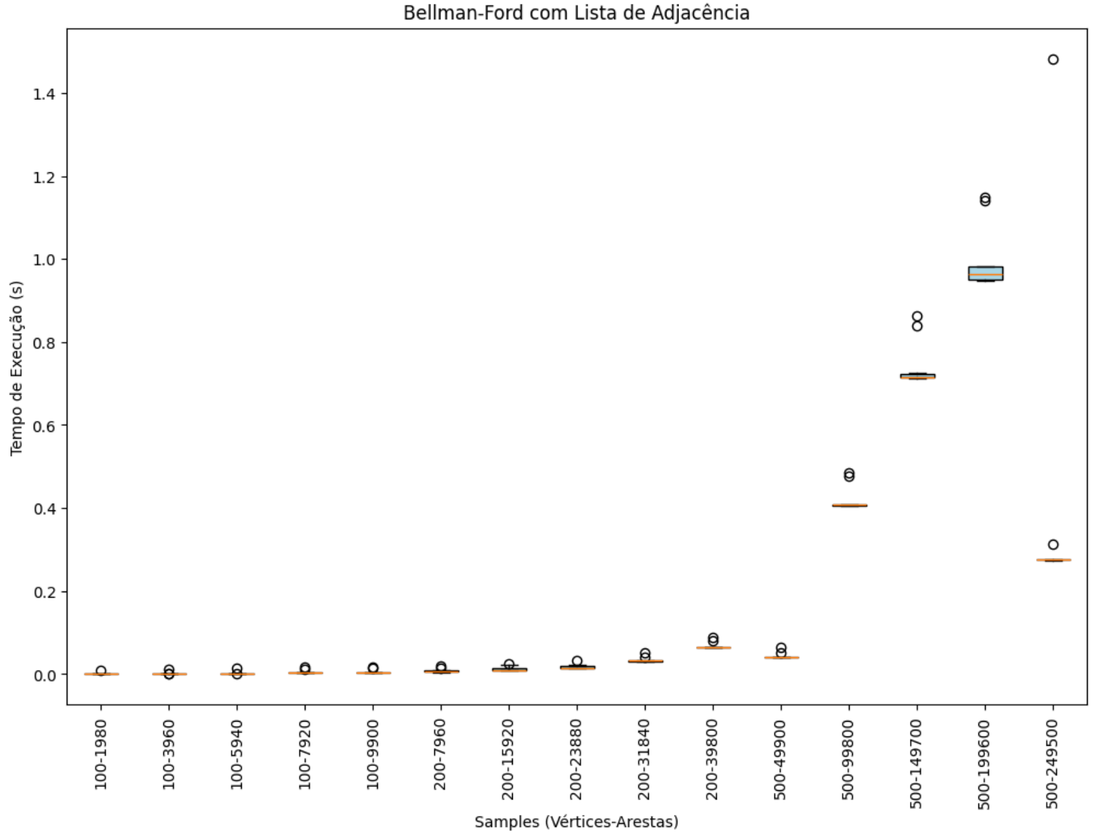
- O desempenho dos algoritmos de Bellman-Ford e Dijkstra;
- O impacto das representações de grafos (lista e matriz de adjacência) na eficiência dos algoritmos.
- A escalabilidade e o custo computacional dos algoritmos em grafos de diferentes tamanhos e densidades.

Metodologia

- Implementação: Algoritmos em Java com medição de tempo via nanoTime().
- Grafos: 15 grafos sintéticos variando em tamanho e densidade, testados 10 vezes cada.
- Processamento e visualização dos dados com Python.
- Ambiente de execução:
  - Microprocessador multicore AMD Ryzen 7 5700U 3,3 GHz, 8 núcleos com 16 threads;
  - 12 GB de memória DDR4;
  - Distribuição Linux Ubuntu 24.04 LTS;

Resultados e discussões: *Boxplots* -Bellman-Ford

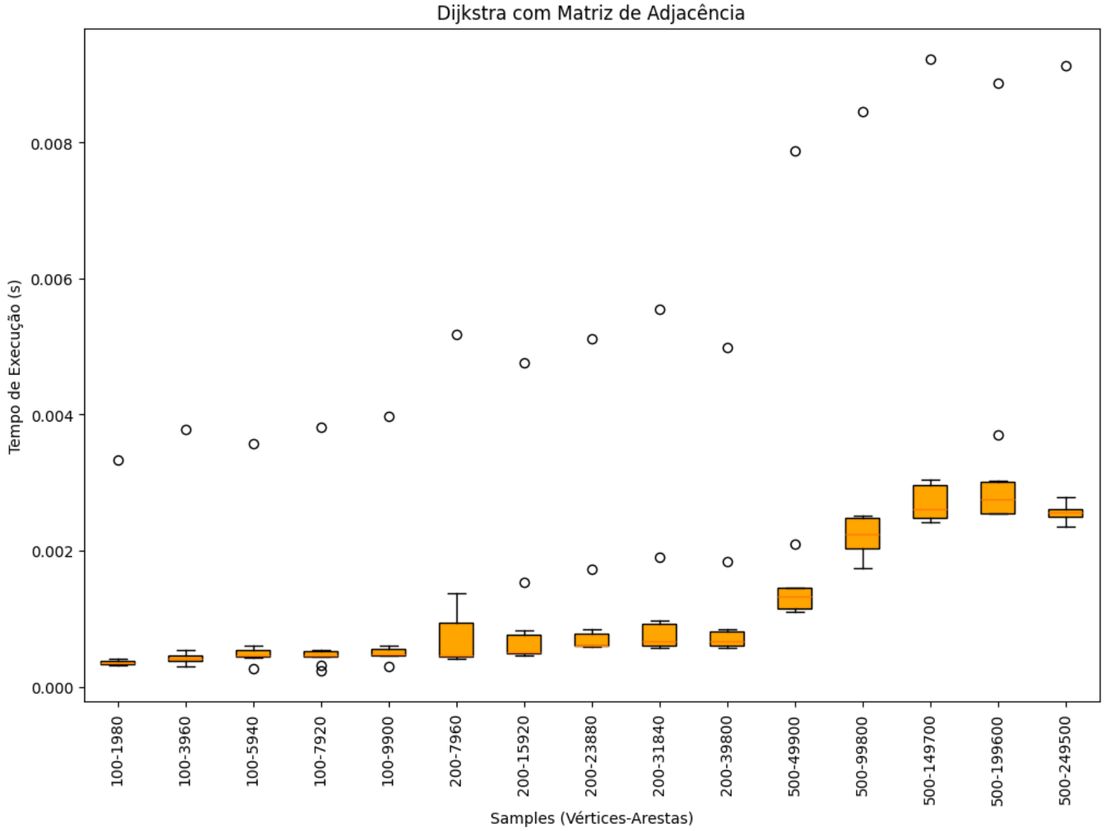


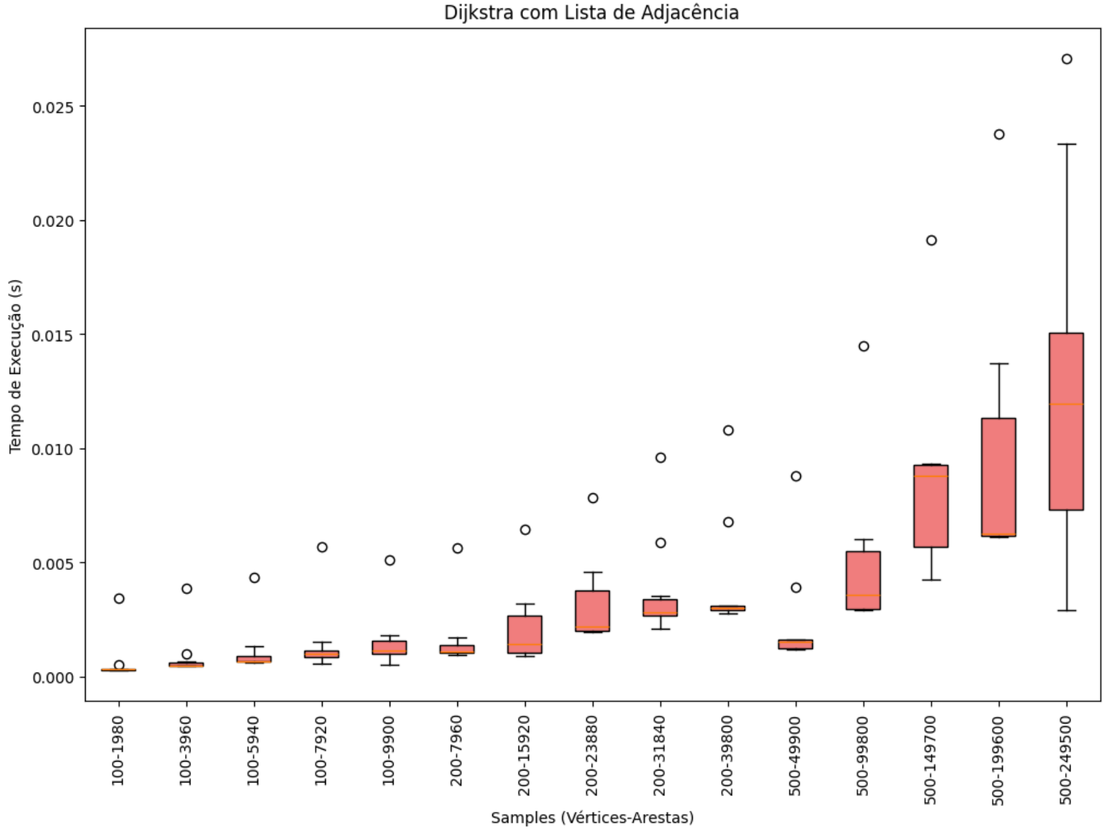


MA

LA

Resultados e discussões: *Boxplots* - Dijkastra

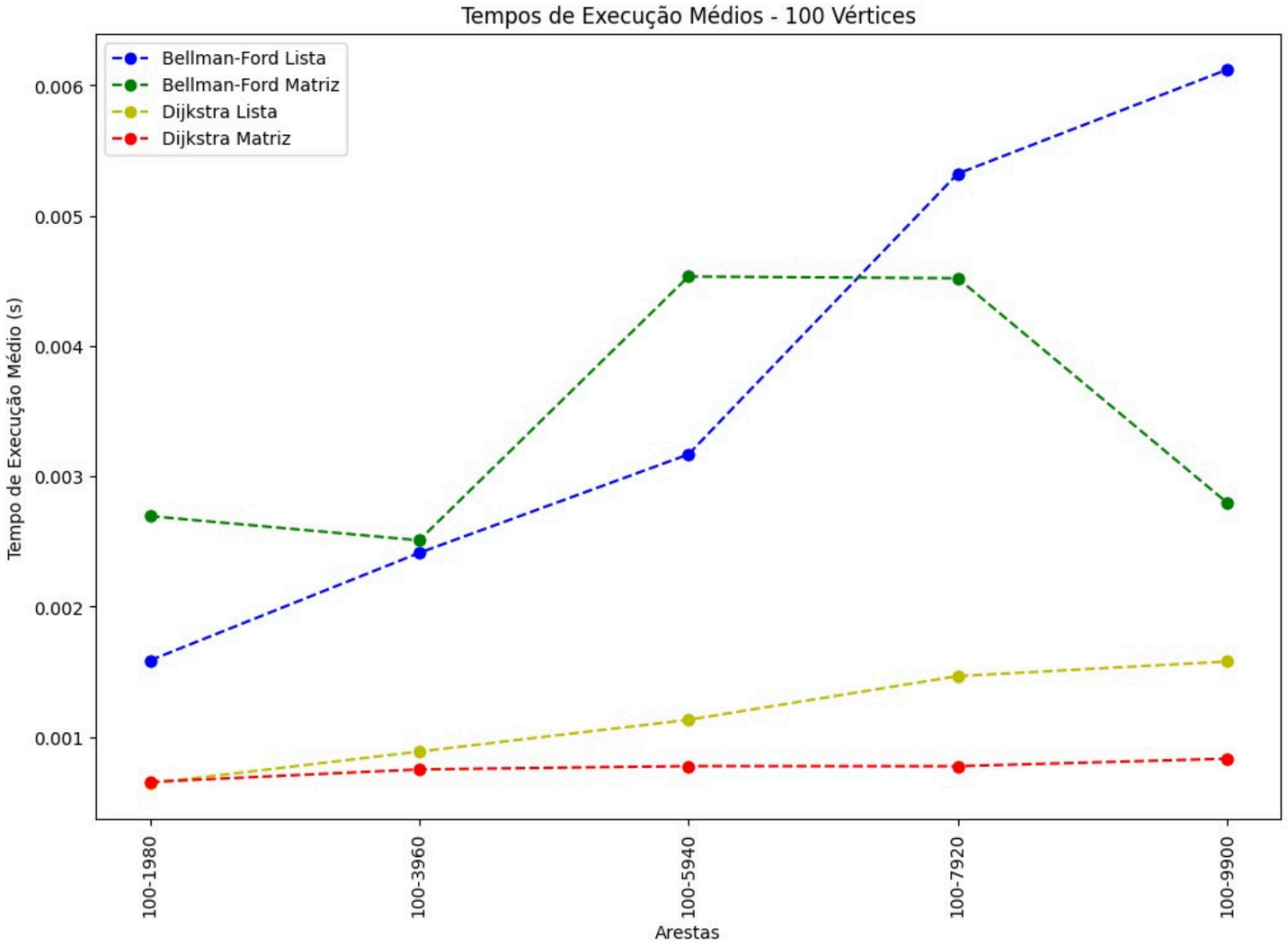




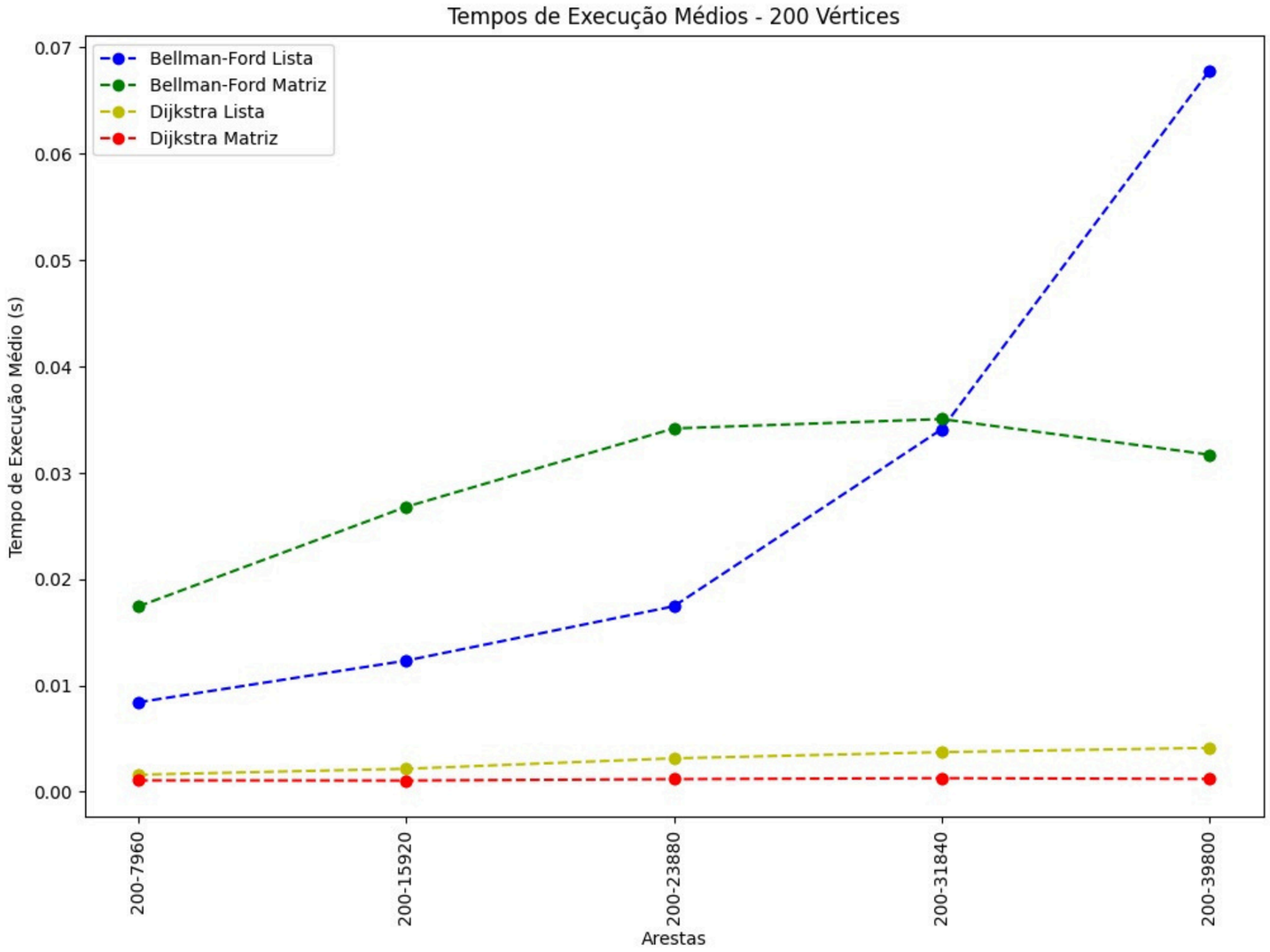
MA

LA

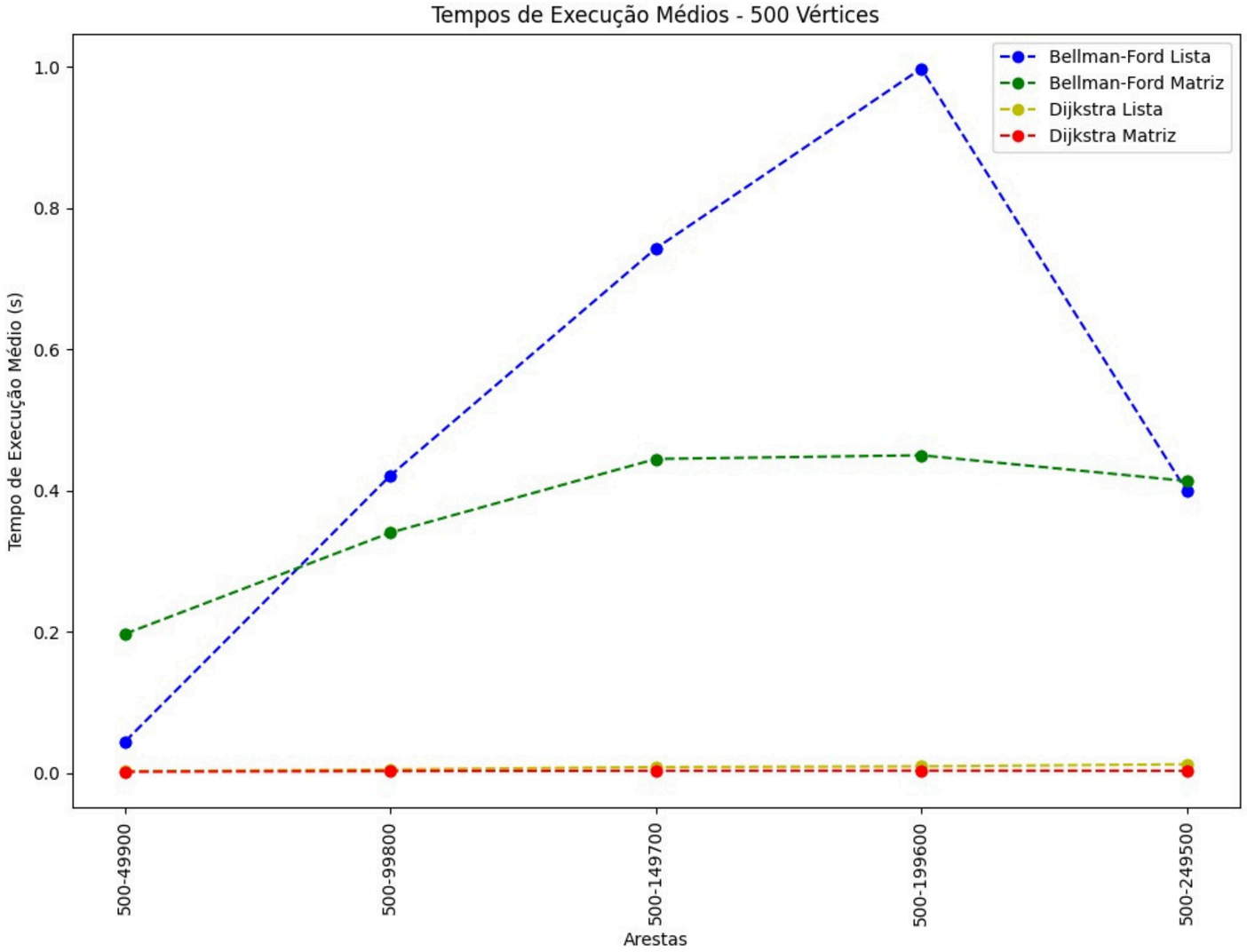
Resultados e discussões: Comparativo das implementações



Resultados e discussões: Comparativo das implementações



Resultados e discussões: Comparativo das implementações



Resultados e discussões: Comparativo das implementações

- A escolha entre Lista de Adjacência e Matriz de Adjacência impacta o desempenho dos algoritmos Bellman-Ford e Dijkstra.
- Bellman-Ford tem melhor desempenho com LA em grafos esparsos e MA em grafos densos.
- Dijkstra é mais eficiente com MA em grafos densos e LA em grafos esparsos.
- A densidade do grafo afeta diretamente os tempos de execução, com LA mais eficiente em grafos esparsos e MA em grafos densos.

OBRIGADO!

davilemos.aluno@unipampa.edu.br
heduardorocha.aluno@unipampa.edu.br