UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA

MAXIMUS BORGES DA ROSA

Laboratório 1

Porto Alegre
2025

Tarefa

A tarefa consiste em realizar uma avaliação experimental do algoritmo de Dijkstra, medindo seu desempenho com diferentes tamanhos de grafos e quantidades de arestas e vértices, utilizando a estrutura de dados Min-Heap e comparando os resultados observados com a complexidade teórica esperada.

Solução

Implementei o algoritmo de Dijkstra utilizando uma fila de prioridade com um heap k-ário. Foram coletadas as métricas: tempo de execução, uso de memória e número de operações.

No nível do heap, foram contadas as operações elementares *sift-up* e *sift-down*, a fim de calcular os valores r médios para cada operação, dados por:

$$r_{\text{operação}} = \frac{\#\text{sifts}}{\log_k n}$$

No nível do algoritmo, foram contadas as operações elementares `insert`, `deletemin` e `update`, a fim de compará-las em relação ao número de vértices e arestas dos grafos.

Para isso, adaptei a implementação para contabilizar essas operações internamente durante a execução do algoritmo. Os arquivos com esses dados estão disponíveis em anexo ao envio deste trabalho.

Todos os testes foram feitos com valores calculados através de grafos gerados aleatoriamente e $k = 8$. A lógica de geração dos grafos aleatórios foi a seguinte:

- **Variação de vértices:** Para analisar o impacto da variação no número de vértices (n), foram gerados 15 grafos com um número fixo de arestas $m = 2^{12}$, enquanto o número de vértices n variou segundo a fórmula:

  $$n = \left(\sqrt{2}\right)^i = 2^{i/2}, \quad \text{com } i \in [13, 27]$$

  Essa abordagem permite observar o comportamento do algoritmo à medida que a densidade do grafo (razão entre arestas e vértices) diminui.

- **Variação de arestas:** Para a análise da variação no número de arestas (m), foram gerados outros 15 grafos com o número de vértices fixado em $n = 2^{15}$, enquanto o número de arestas m variou de acordo com:

  $$m = \left(\sqrt{2}\right)^i = 2^{i/2}, \quad \text{com } i \in [31, 45]$$

  Essa variação permite avaliar o desempenho do algoritmo conforme o grafo se torna mais denso.

Em ambos os casos, o crescimento exponencial controlado das variáveis foi escolhido para permitir uma análise mais fina do impacto de n e m separadamente sobre o

tempo de execução do algoritmo de Dijkstra. Os dados de desempenho (como tempo de execução normalizado) foram coletados em 30 execuções para cada par (n, m), e a média foi utilizada como métrica principal.

Resultados

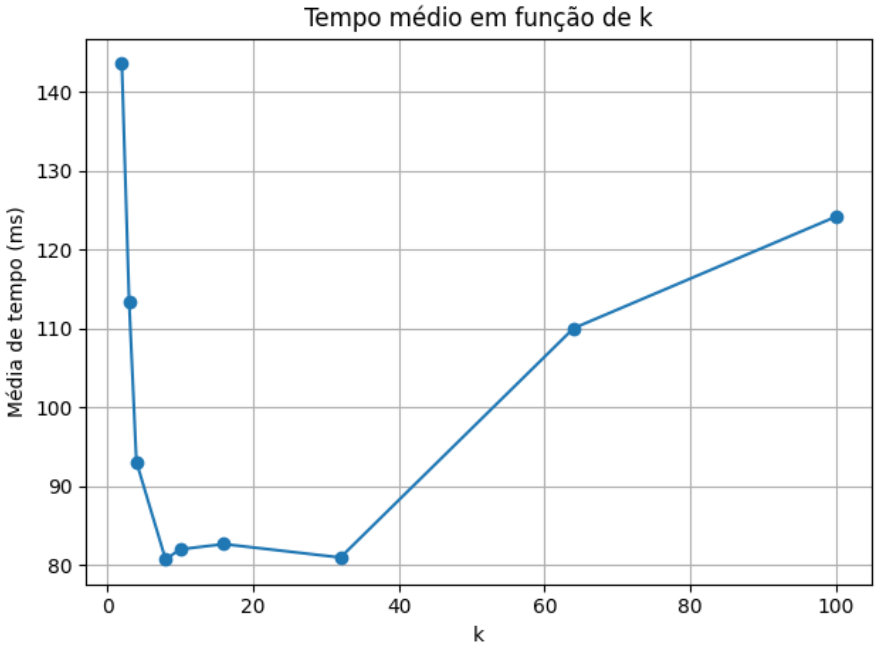

Figura 1.1 – Tempo médio para diferentes valores de k.

Foram testados os seguintes valores de k: 2, 3, 4, 8, 10, 16, 32, 64 e 100, com $k = 8$ obtendo o melhor resultado (média 80,736) e $k = 2$ o pior resultado (média 143,562). A diferença percentual entre o pior e o melhor caso é de aproximadamente:

$$\frac{143.562 - 80.736}{143.562} \times 100 \approx \textbf{43,76\%}$$

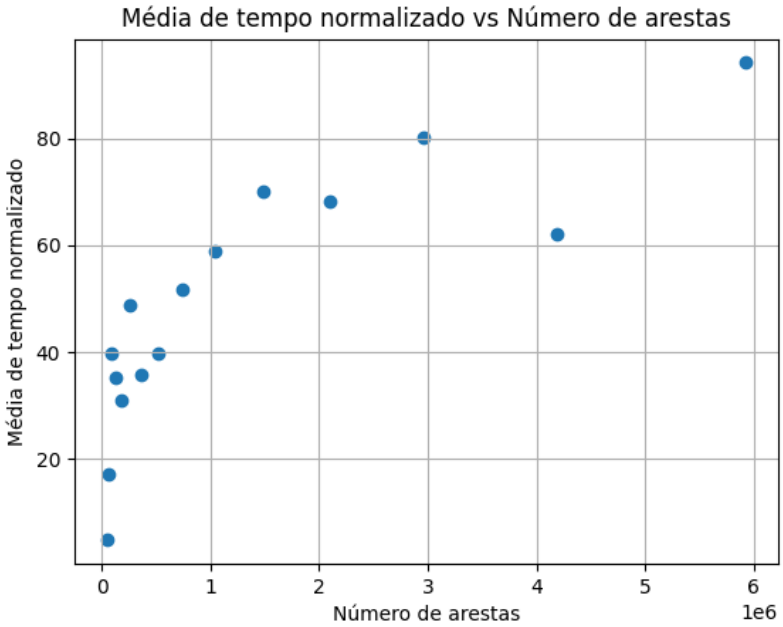

Figura 1.2 – Tempo de execução em função do número de arestas (m).

A curvatura dos pontos sugere comportamento logarítmico.

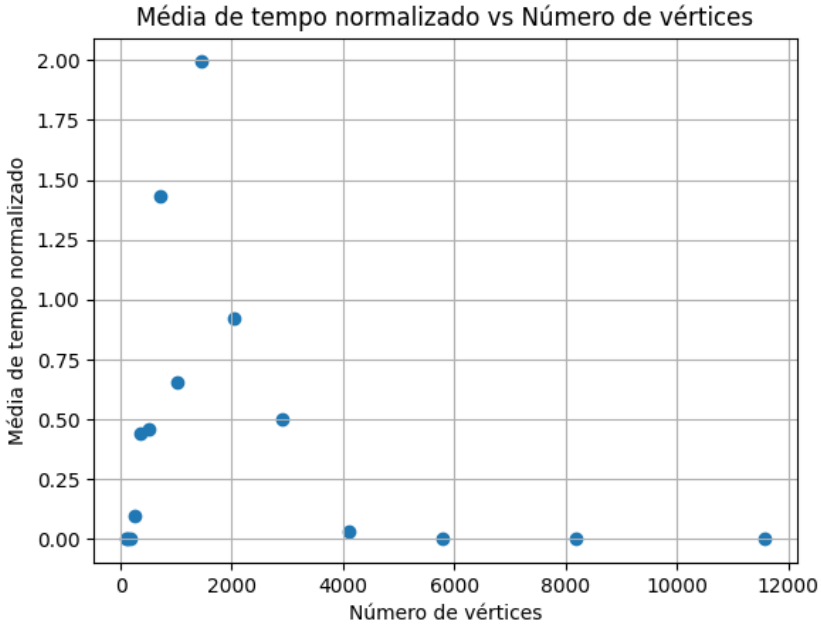

Figura 1.3 – Tempo de execução em função do número de vértices (n), com m fixo.

O comportamento de declínio após $n = 2000$ com m constante sugere que o número de arestas é dominante na complexidade do algoritmo.

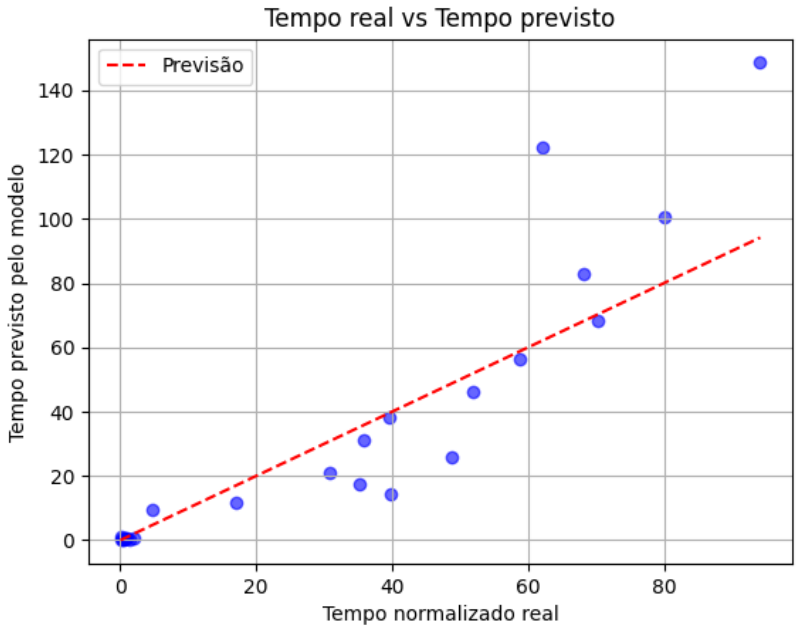

Figura 1.4 – Ajuste de regressão para o tempo de execução.

Tendo como base a regressão:

$$T(n, m) \approx 1.87449 \cdot 10^{-4} \cdot n^{0.464} \cdot m^{0.562}$$

com $R^2 = 0.5779$ e $RMSE = 1.92846 \cdot 10^1$, podemos ver que o crescimento do tempo se comporta de maneira relativamente próxima à análise assintótica esperada. Interessante notar que o termo m possui expoente de maior valor, o que reforça indícios de sua dominância.

Grafo	**n**	**m**	**Tempo médio (ms)**	**Memória (kB)**
New York	264346	733846	896.73	161196
East	598623	8778114	11991.33	245506.40
West	6262104	15248146	21962.50	421525.86

Tabela 1.1 – Média de tempo e memória para grafos que simbolizam estradas reais dos EUA.
Fonte: http://www.diag.uniroma1.it/ challenge9/download.shtml

Houve problemas técnicos ao executar o algoritmo no grafo completo dos EUA, por isso a escolha por utilizar "East"e "West".

Ambiente de Teste

Os testes foram executados em um processador **Intel Core i5-10210U** de 1.60 GHz com 12 GB de RAM, em um sistema **Windows 11** utilizando **WSL**. O código foi desenvolvido em **C++** utilizando a IDE **Visual Studio Code**.

Conclusão

Os resultados experimentais confirmam que o algoritmo de Dijkstra tem desempenho compatível com sua complexidade teórica, especialmente ao observar a relação entre tempo de execução e a função $(m + n) \log n$. A normalização do tempo sugere comportamento assintótico estável.

Além disso, a escolha de um *"k ideal"* se mostrou estar entre $[8, 32]$, tendo impacto significativo no desempenho do algoritmo, com uma diferença de aproximadamente **43,76%** entre o melhor caso ($k = 8$) e o pior caso ($k = 2$).

Por fim, os testes indicam que o número de arestas m é o fator dominante na complexidade do algoritmo de Dijkstra quando se utiliza uma estrutura de heap k-ária.