Trabalho Prático de Grafos

1 Modelagem

Para representar o grafo, foi utilizada a matriz de adjacências. Para a resolução do problema, são utilizados dois algoritmos: **Floyd-Warshall** e **Dijkstra**.

O primeiro problema apresentado é para encontrar o diâmetro do grafo, ou seja, a maior distância entre quaisquer dois pares de vértices, e também quais são seus vértices diametrais. Para isso, utilizamos o algoritmo de **Floyd-Warshall**, que nos permite calcular as distâncias para todos os pares de vértices. Essas distâncias são armazenadas em uma matriz, que depois é percorrida para encontrar a maior distância e quais são os vértices desta distância, chamados de vértices diametrais.

Após encontrar os vértices diametrais, deve-se encontrar o caminho mínimo entre eles. Como não existem pesos negativos no grafo, podemos utilizar o **Dijkstra** para isso, realizando uma pequena modificação para para armazenar o "pai" de cada vértice no caminho, para construirmos o caminho depois no *output* do programa.

2 Análise de Complexidade

As duas funções mais significativas do programa são descritas abaixo. Todas as outras funções são auxiliares para ler a matriz ou construir o *output*.

```
def floyd_warshall(G):
    # restante do codigo no programa
```

O algoritmo usado é o **Floyd-Warshall**. Sua complexidade, no pior caso, é de $O(|V|^3)$.

```
def dijkstra(G, source, target):
    # restante do codigo no programa
```

O algoritmo usado é o **Dijkstra**. Sua complexidade, no pior caso, é de $O(|V|^2)$, quando é utilizada a matriz de adjacências para representar o o grafo.

O tempo de execução do programa como um todo, então, é definido pela complexidade do algoritmo do **Floyd-Warshall**, que é $O(|V|^3)$.