

# Assignment 1

## Teoria de Grafos e Computabilidade

Eric Azevedo de Oliveira<sup>1</sup>, Felipe Nepomuceno Coelho<sup>1</sup>, Iyan Lucas Duarte Marques<sup>1</sup>

<sup>1</sup>Instituto de Ciências Exatas e Informática - Pontifícia Universidade Católica Minas Gerais (PUC-MG)

### 1. Problema

*"A Distribuidora Embel está abrindo 5 novos pontos de distribuição de seus produtos em Belo Horizonte. O abastecimento desses pontos é realizado por Jorge, que precisa entregar os produtos em todas as lojas. Qual seria o melhor percurso para Jorge"*

### 2. Representação

Considerando os pontos de distribuição marcados no mapa de Belo Horizonte<sup>1</sup> como os vértices, podemos modelar um grafo  $G = (V, E)$ , em que  $V$  é o conjunto de vértices e  $E$ , o conjunto de arestas. De tal forma que os pontos 0, 1, 2, 3 e 4 pertencem ao conjunto  $V$  e as arestas que se interligam pertencem ao conjunto  $E$ .

$$E = \{\{0, 1, 2, 3, 4\} \forall 0, 1, 2, 3, 4 \in V\} \quad (1)$$



---

<sup>1</sup>Mapa abaixo extraído do Google Earth

## 2.1. Não Direcionado, Não Ponderado

A representação do grafo com as arestas ligando cada vértice.



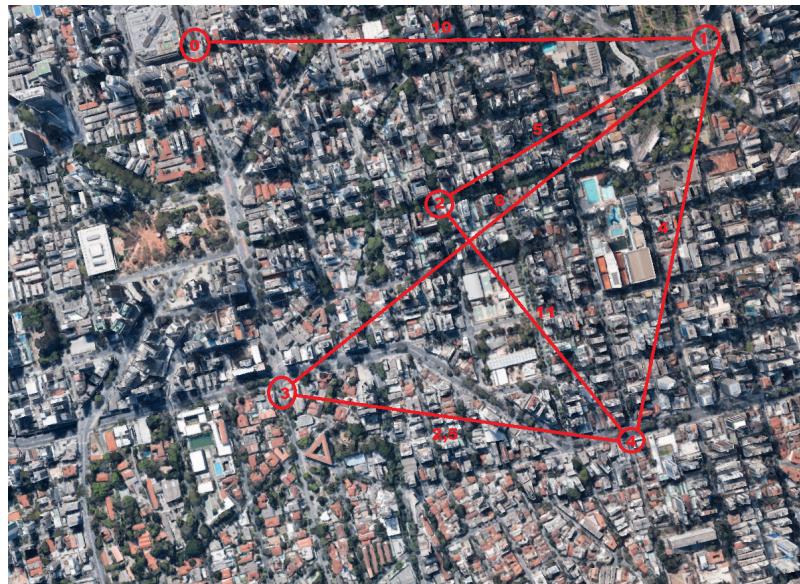
## 2.2. Direcionado, Não Ponderado

PA representação do grafo com as arestas representando a ligação entre os vértices e, as direções como o fluxo das vias de Belo Horizonte.



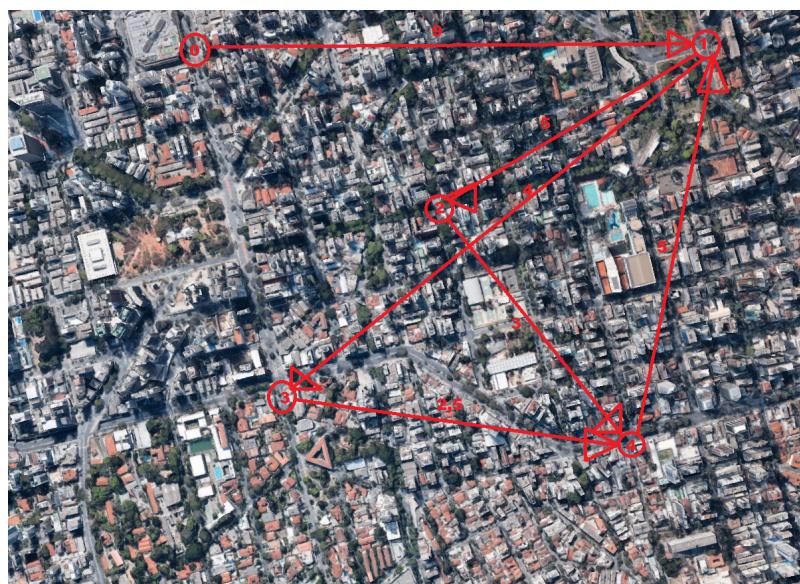
### **2.3. Não Direcionado, Ponderado**

A representação do grafo com as arestas representando as distâncias entre os vértices, desconsiderando o sentido de fluxo das vias de Belo Horizonte.



### **2.4. Direcionado, Ponderado**

A representação do grafo com as arestas representando as distâncias entre os vértices, considerando o sentido de fluxo das vias de Belo Horizonte.



### 3. Implementação

#### 3.1. Execução

O nosso algoritmo é utilizado no processamento dos grafos, direcionais ou não direcionados e ponderados ou não ponderados. A base do mesmo consiste em sua entrada. A qual será inserida pelo usuário, o número de vértice, grau máximo e se é ponderado ou não. Com esses dados o algoritmo constrói uma coluna na memória que corresponderá aos vértices do nosso grafo. Tendo em cada linha, tamanho máximo relacionado ao grau máximo, o inicio e fim da aresta que estão ligadas aos vértices.

Após a alocação dinâmica, partiremos para a entrada do nosso algoritmo, que é constituída pelo primeiro e segundo vértice. Um atributo que define se ele é ou não direcionado e se a ligação tem algum peso. Com isto, nosso algoritmo analisará os vértices e os coloca dentro da alocação dinâmica.

Desta forma, será mostrado na tela o formato mais adequado para aquele grafo:

- **Grafo direcionado não-ponderado:** Neste tipo de grafo, utilizaremos a amostragem por meio da matriz de adjacência. Isto se deve pelo fato da mesma ser representada com uma linha e uma coluna para cada nó. Portanto, para este caso, esta amostragem se apresenta a mais efetiva pelo fato do grafo não ser ponderado e ser direcionado. Sendo assim, não será necessário demonstrar o peso, a entrada e saída da aresta. Desta forma, esta representação possibilita uma visualização mais clara do grafo. Segue o exemplo:

	0	1	2	3	4
0:	0	1	0	0	0
1:	0	0	1	1	0
2:	0	0	0	0	1
3:	0	0	0	0	1
4:	0	1	0	0	0

Na imagem acima, esta sendo caracterizada uma matriz adjacência, no qual o número 1 é representado como ligação entre os vértices e, o 0, como nenhuma ligação entre eles. Esta forma de representação, como não é necessário mostrar o peso das arestas, torna-se uma tarefa muito mais fácil e corrobora para o entendimento e visualização. Além disto, como é um grafo direcionado, ao se analisar a matriz, temos noção de onde as arestas estão conectadas com o 1.

- **Grafo não-direcionado não-ponderado:** Neste tipo de grafo, utilizaremos as amostragens por lista de adjacência. Isto é, uma lista de nós, em que cada nó aponta para a aresta de seu sucessor ou nó adjacente. Como não precisaremos utilizar pesos e não existe direcionamento, a forma da matriz de adjacência ficaria muito desorganizada, e sua leitura ficaria comprometida. Por isto, a lista será uma escolha mais adequada, tendo em si, uma representação com um entendimento mais fácil. E tendo em seu corpo, os nós e as arestas de forma descrita e não como binário, mesmo gastando mais memória, que a matriz neste caso é superior.

```

0: [1],
1: [0],[3],[2],[4],
2: [1],[4],
3: [1],[4],
4: [2],[3],[1],

```

Na imagem acima, temos a lista de adjacência com suas arestas e seus nós. Os mesmos estando no começo de cada linha, de 0 ate 4. E, suas arestas, as quais estão explícitas entre colchetes, após dos dois pontos. Ao vermos esta estrutura, podemos ver de uma forma muito mais clara e rápida todas as ligações que nosso grafo possui.

- **Grafo direcionado ponderado:** Neste tipo de grafo, o peso entre os vértices é fundamental. Por isso, utilizamos novamente a lista de adjacência, pelo fato dela conter as arestas descritas em cada espaço de colchete como o peso de cada uma. Da mesma forma, este tipo de apresentação custa muito em termos de memória, mas pelo fato do peso ser fundamental na apresentação, a perda de eficiência vale pelo resultado dado.

```

0: V[1] Peso(10.00),
1: V[3] Peso(6.00), V[2] Peso(5.00),
2: V[4] Peso(11.00),
3: V[4] Peso(2.50),
4: V[1] Peso(4.00),

```

Na imagem acima, temos a lista de adjacência sendo retratado um grafo direcionado. Na qual, o vértice está fazendo uma ligação e o peso de cada aresta, sendo eles descritos das formas v[].

- **Grafo não-direcionado ponderado:** Neste tipo de grafo, não direcionado, faz com que a lista de adjacência também seja uma excelente representação. Pelo fato supracitado, a qual sua representação de peso e de vértice será muito boa. Mesmo tendo perda de eficiência e grande uso da memória em si do computador, gerando uma representação bem clara e objetiva em relação ao grafo.

```

0: V[1] Peso(10.00),
1: V[0] Peso(10.00), V[3] Peso(6.00), V[2] Peso(5.00), V[4] Peso(4.00),
2: V[1] Peso(5.00), V[4] Peso(11.00),
3: V[1] Peso(6.00), V[4] Peso(2.50),
4: V[2] Peso(11.00), V[3] Peso(2.50), V[1] Peso(4.00),

```

Na imagem acima, o grafo não direcionado e ponderado está sendo representado pela lista de adjacência. A qual mostra de maneira clara e coesa o nó que estamos se referindo, os vértices que ele está fazendo a ligação e o peso que a aresta está vinculada.

### 3.2. Executando o Código

Para executar o código, basta compilar com o seguinte comando:

*gcc main.c Grafo.c -o exec*

E para executá-lo:

*/exec*