



**Algoritmos em Grafos - GCC 218**

**Docente** Vinicius Vitor dos Santos Dias

**Discente** Davi Hermógenes Siqueira - 202120102

**Mini-Projeto** : Determinação de Vértices Cruciais em  
Redes Neurais usando Algoritmos de Caminho Mais Curto.

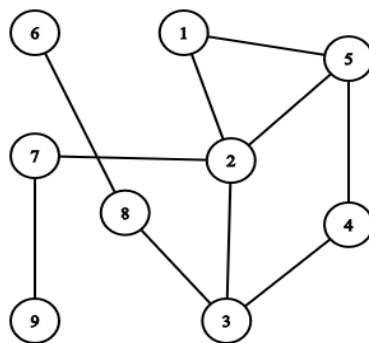
## 1. Definição e Apresentação da Base de Dados

A base de dados utilizada, "fly-drosophila-medulla-1", representa um grafo que descreve a conectividade entre neurônios na medula de **Drosophila** (mosca da fruta), um organismo frequentemente usado em pesquisas em biologia e neurociência. O grafo é conexo, não possui atributos, é **não-direcionado** e **não-ponderado**. Vértices representam neurônios e Arestas conexões. Aqui estão os dados principais associados ao grafo:

- **Vértices** : 1.8 mil nós, que representam neurônios individuais na medula de Drosophila.
- **Arestas** : 33.5 mil arestas, indicando conexões entre pares de neurônios na rede.
- Outros dados são: Densidade
- **Densidade**: 0.0211395; **Grau máximo**: 16.2 mil; **Grau mínimo**: 1; **Grau médio**: 37; **Assortatividade**: -0.325174.

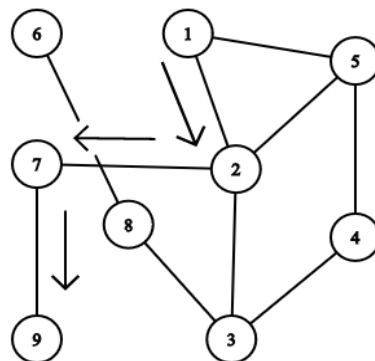
## 2. Limitação do Escopo e Definição do Problema

- Este projeto tem como objetivo encontrar os vértices cruciais para a comunicação eficiente da rede neural da espécie Drosophila. O problema a ser solucionado envolve a compreensão de redes neurais, visando obter informações referentes à análise de neurônios críticos para **caminhos mais curtos** e consequentemente para a comunicação eficiente entre toda a rede. Os vértices que **mais aparecem nos caminhos mais curtos**, **são os essenciais para a comunicação eficiente** entre toda a rede neural. Para melhor compreensão, presuma o seguinte grafo G :



Fonte : autoria própria

Agora, presuma o seguinte caminho mínimo de G :  $1 \rightarrow 9$ , sendo  $\{1, 2, 7, 9\}$

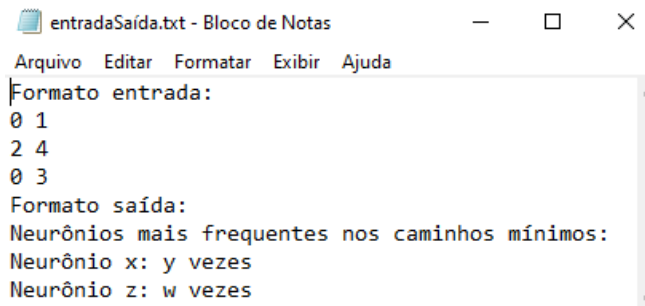


Fonte : autoria própria

Note que o vértice **2** é essencial para o caminho mais curto, sem ele, o caminho  $1 \rightarrow 9$  não existiria. Em termos simples, tome outro exemplo como base:  $1 \rightarrow 3$ , o caminho mais curto neste caso é  $\{1, 2, 3\}$ , porém, sem a presença do 2, o caminho seria  $\{1, 5, 4, 3\}$ .

Dito isso, o objetivo é **encontrar vértices similares ao exemplificado**, que tornam a comunicação neural mais eficiente, isto é, os essenciais.

Abaixo se encontra um exemplo de entrada e saída esperada, respectivamente. Ambos exemplificados utilizando arquivos texto.



```
Formato entrada:
0 1
2 4
0 3
Formato saída:
Neurônios mais frequentes nos caminhos mínimos:
Neurônio x: y vezes
Neurônio z: w vezes
```

fonte : autoria própria

### 3. Minha Solução Utilizando Algoritmos:

- A ideia fundamental é utilizar uma versão modificada do algoritmo [Floyd-Warshall](#), na qual todas as arestas têm peso 1, simulando um grafo não-ponderado. Este algoritmo calcula o caminho mais curto de todos para todos. Após isso, é efetuada uma busca, a fim de encontrar os vértices mais frequentes nos caminhos mais curtos. O algoritmo foi implementado utilizando matriz de adjacências e os dados de entrada foram obtidos de um arquivo texto comum.

Em relação a complexidade do algoritmo :

- O algoritmo de Floyd-Warshall possui complexidade  $O(V^3)$ .
- A leitura do arquivo e a construção do grafo têm uma complexidade de  $O(E)$ , onde  $E$  é o número de arestas no grafo.
- A inicialização da matriz do grafo tem complexidade  $O(V^2)$ .
- A complexidade geral é dominada pelo **Floyd-Warshall**,  $O(V^3)$ .

Abaixo há um pseudocódigo do funcionamento geral, sem considerar funções auxiliares.

```
floydWarshall(dist, V, caminhos)
    PARA k DE 0 ATÉ V-1
        PARA i DE 0 ATÉ V-1
            PARA j DE 0 ATÉ V-1
                SE dist[i][j] > dist[i][k] + dist[k][j]
                    E dist[i][k] != INF E dist[k][j] != INF
                        dist[i][j] = dist[i][k] + dist[k][j]
                        caminhos[i][j] = k

main()
    dist = matriz de adjacência inicializada com INF
    caminhos = matriz de caminhos inicializada com -1
    Inicializar dist com 0 para distâncias próprias
    Ler arestas e atualizar dist
    chamar floydWarshall(dist, número de vértices, caminhos)
    chamar encontrarVerticesFrequentes(caminhos, V, dist)
```

**4. Ferramenta/Software:** Foi utilizada a linguagem de programação C++, versão do compilador 6.3.0. A arquitetura é baseada em funções modulares, sendo as mais importantes, 'floydWarshall' e 'EncontrarVérticesFrequentes'. Foi feito o uso da estrutura de dados 'Vector' da biblioteca padrão do C++, a fim de implementar de forma fácil vetores dinâmicos. Em relação a compilação, execução e entrada de dados :

- Utilize um compilador C++ padrão.
- O código e os arquivos texto necessários estão configurados corretamente utilizando o makefile..
- O software lê os dados da rede neural a partir de um arquivo de texto chamado "fly-drosophila medulla 1.txt", sem aspas. Já incluído no código.

**5. Resultados e Limitações :** O programa identificou os vértices (neurônios) que aparecem com mais frequência nos caminhos mínimos da rede neural. Esses vértices são pontos de convergência frequentes, sugerindo sua importância na conectividade global da rede. É importante reconhecer as limitações do programa, como a simplificação para grafos não ponderados e não direcionados. Essa abordagem é específica para identificar padrões de conectividade em redes simples, sem considerar nuances mais complexas. Os resultados podem ser relevantes em relação a compreensão de neurônios-chave em redes neurais complexas, no contexto da biologia e estudo da comunicação neuronal. Fornecendo uma abordagem prática para identificá-los. Em relação às limitações :

- **Complexidade Computacional :** o algoritmo de Floyd-Warshall utilizado tem uma complexidade de  $O(V^3)$ , o que pode limitar sua escalabilidade para redes neurais muito grandes.
- **Modelo de Ponderação Fixa :** A abordagem adotada assume uma ponderação fixa para todas as arestas, considerando-as igualmente importantes. Em redes neurais biológicas, a natureza das conexões pode variar, e a atribuição uniforme de pesos pode simplificar demais a realidade biológica.
- **Dependência da Base de Dados:** A implementação é específica para a base de dados utilizada, sendo necessário reavaliar o algoritmo a fim de usar uma base de dados maior. Visto que, como mencionado, a escalabilidade fica comprometida.

"[Eu] Pensava que nós seguíamos caminhos já feitos, mas parece que não os há. O nosso ir faz o caminho." - **C.S. Lewis**

## **Bibliografia**

**Mack, David. Finding shortest paths with Graph Neural Networks.**

**Medium. Jan 7, 2019. Disponível em:**

**<<https://medium.com/octavian-ai/finding-shortest-paths-with-graph-net-works-807c5bbfc9c8>>. acesso em : 4/12/2023**

**Fly-Drosophila-Medulla-1. NetworkRepository. Disponível em**

**<<https://networkrepository.com/bn-fly-drosophila-medulla-1.php>>. acesso em : 3/12/2023**

**Floyd Warshall Algorithm. GeeksforGeeks. Disponível em**

**<<https://www.geeksforgeeks.org/floyd-warshall-algorithm-dp-16/>>. acesso em : 4/12/2023**