Trabalho 1 - Teoria dos Grafos

Arthur Wallace Silva Lopes, João Paulo Euzébio Paiva, Lucas Rodrigues Lopes

[1]Instituto Federal de Brasília (IFB)

1. Introdução

Este trabalho foi desenvolvido utilizando a linguagem de programação **Python** e a biblioteca **Networkx** para o tratamento e manipulação de grafos a fim de demonstrar de forma prática os conteúdos estudados em sala de aula. Inicialmente, foram disponibilizadas duas base de dados contendo os dados dos grafos a serem utilizados para a análise empírica (*As_graph.txt* e *collaboration_graph.txt*), com base nelas, foi desenvolvida uma biblioteca de funções capazes de manipular e salvar os dados relativos à essas execuções.

2. Estudo de Caso 1 - collaboration_graph.txt

2.1. Compare o desempenho em termos de quantidade de memória utilizada das duas representações do grafo. Determine a quantidade de memória (em MB) utilizada pelo seu programa quando você representa o grafo utilizando uma matriz de adjacência e lista de adjacência.

Como podemos observar no gráfico abaixo, para realizar a representação desse grafo por meio da geração da matriz de adjacência e da lista de adjacência foram consumidas quantidades bem distintas de memória. Enquanto a construção da lista de adjacência é uma tarefa relativamente simples, que não exige muito poder computacional para ser realizada, atingindo um pico de **4,51MB** de memória, a representação por meio da matriz de adjacência se mostrou uma tarefa complexa e que exige bastante poder computacional da máquina em que é executada, atingindo o pico de **8397,86MB** ou **8,39GB** de memória para ser executada.

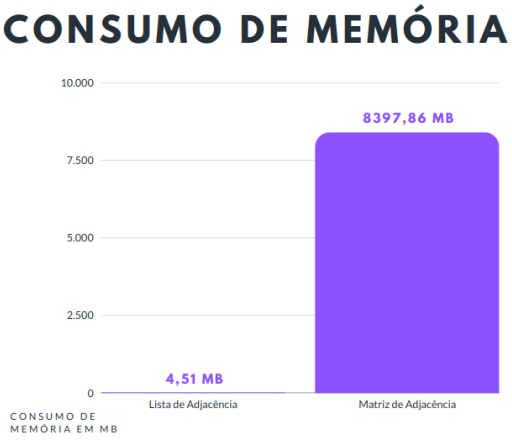

Figura 1. Gráfico de consumo de memória

2.2. Compare o desempenho em termos de tempo de execução das duas representações do grafo. Determine o tempo necessário para executar uma busca em largura em cada um dos casos (considere o pior caso).

Ao analisarmos os tempos de execução para a construção de cada uma das representações, também se torna perceptível a diferença de eficiência entre elas. Conforme apresentados no gráfico abaixo, os tempos de execução para a matriz de adjacência são amplamente superiores aos da lista de adjacência, levando, em média, **15min e 31seg** para se criar a matriz e **0,27seg** para se criar a lista de adjacências.

Figura 2. Gráfico de tempo de execução

2.3. Obtenha os componentes conexos do grafo. Quantos componentes conexos tem o grafo? Qual é o tamanho do maior e do menor componente conexo?

Esse grafo possui **8826** componentes conexos. Sendo que o menor deles possui **2** vértices e o maior deles possui **33533** vértices. O arquivo com os componentes conexos desse grafo estão disponíveis em: `https://github.com/ArthurWallaceIFB/teoriaGrafos_trabalho1/blob/main/graficos/componentes_collaboration.txt`

3. Estudo de Caso 2 - As_graph.txt

Considere o grafo de conexão das redes que formam a Internet (AS Graph). Utilizando este grafo, responda às perguntas abaixo.

3.1. Trace um gráfico com seu resultado. Qual é o maior grau do grafo? E o menor? Como isto se compara ao maior grau possível?

O maior grau do grafo é **2159** e o menor é **0**.

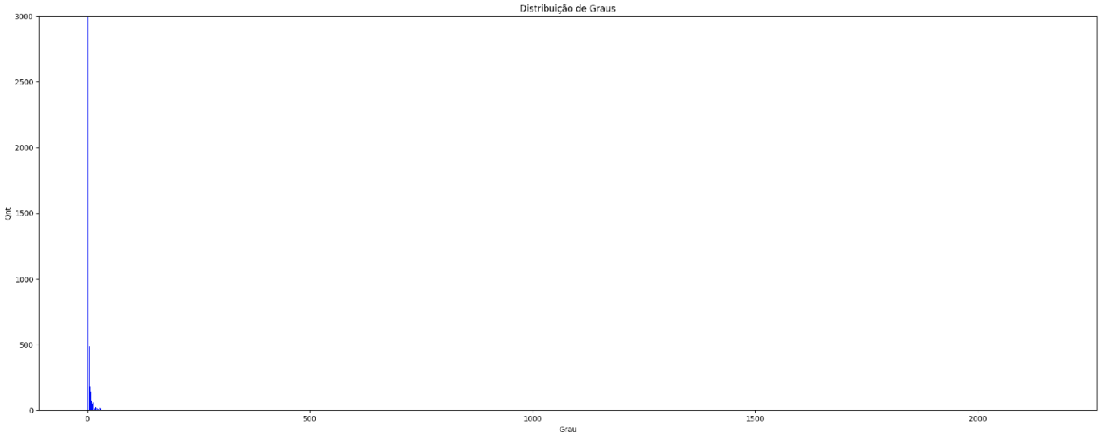

Figura 3. Gráfico de graus

3.2. Obtenha os componentes conexos do grafo. Quantos componentes conexos tem o grafo? Qual é o tamanho do maior e do menor componente conexo?

Este grafo possui apenas um componente conexo que contém os 32385 vértices existentes. Dessa forma, o maior componente conexo tem tamanho igual 32385. O arquivo contento todos os vértices desse componente pode ser consultado em: `https://github.com/ArthurWallaceIFB/teoriaGrafos_trabalho1/blob/main/graficos/componentes_as_graph.txt`

3.3. Faça uma busca em largura a partir do vértice 1. Neste caso, o maior nível da árvore geradora de busca representa a maior distância do vértice 1 a qualquer outro. Determine este valor. Repita este procedimento para outros vértices. O que você pode concluir?

Partindo do vértice '1', o maior nível tem valor igual a 32384 e é representado pelo vértice '29809'. Ao executar para os outros vértices, é possível concluir que o nível de maior valor será sempre o número de vértices totais menos 1, nesse caso: $(32385 - 1) = 32384$.

3.4. Determine o diâmetro da Internet. Lembrando que o diâmetro é a maior distância entre qualquer par de vértices do grafo (ou seja, o comprimento do maior caminho mínimo do grafo). Utilize a BFS para responder esta pergunta.

Tomando como base os resultados obtidos na execução anterior, é possível afirmar que o diâmetro desse grafo, ou seja a maior distância entre qualquer par de vértices do grafo, será de **32384**. Pois representa o último nível dessa busca por largura e profundidade.

O código para esse projeto está disponível em: `https://github.com/ArthurWallaceIFB/teoriaGrafos_trabalho1`