Teoria dos Grafos

Trabalho de disciplina – Parte 1

César Tallys Henrique Duarte

Instituto Federal de Brasília – Campus Taguatinga

1. Introdução

Este trabalho tem como objetivo desenvolver uma biblioteca que seja capaz de manipular grafos. Para o desenvolvimento de algumas funções da biblioteca, houve o uso de algumas outras bibliotecas externas, como a **Networkx,** a fim de garantir maior praticidade. Os testes realizados foram feitos com uso de grafos disponibilizados contendo os dados dos grafos para análise (*as_graph.txt, collaboration_graph.txt*). Todos os arquivos desse projeto estão disponíveis no repositório do GitHub: https://github.com/CesarTHD/Teoria-dos-Grafos

2. Estudo de caso 1 – collaboration_graph

2.1. Comparação de desempenho em termos de memória entre lista e matriz de adjacência

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 simples que exige, relativamente, baixo poder computacional para ser realizada, atingindo um consumo total de 25.503MB (~25GB) 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 um consumo total maior que 60.000MB (~ 60GB) de memória para ser executada.

2.2. Comparação de desempenho em termos de tempo de execução das duas representações do grafo

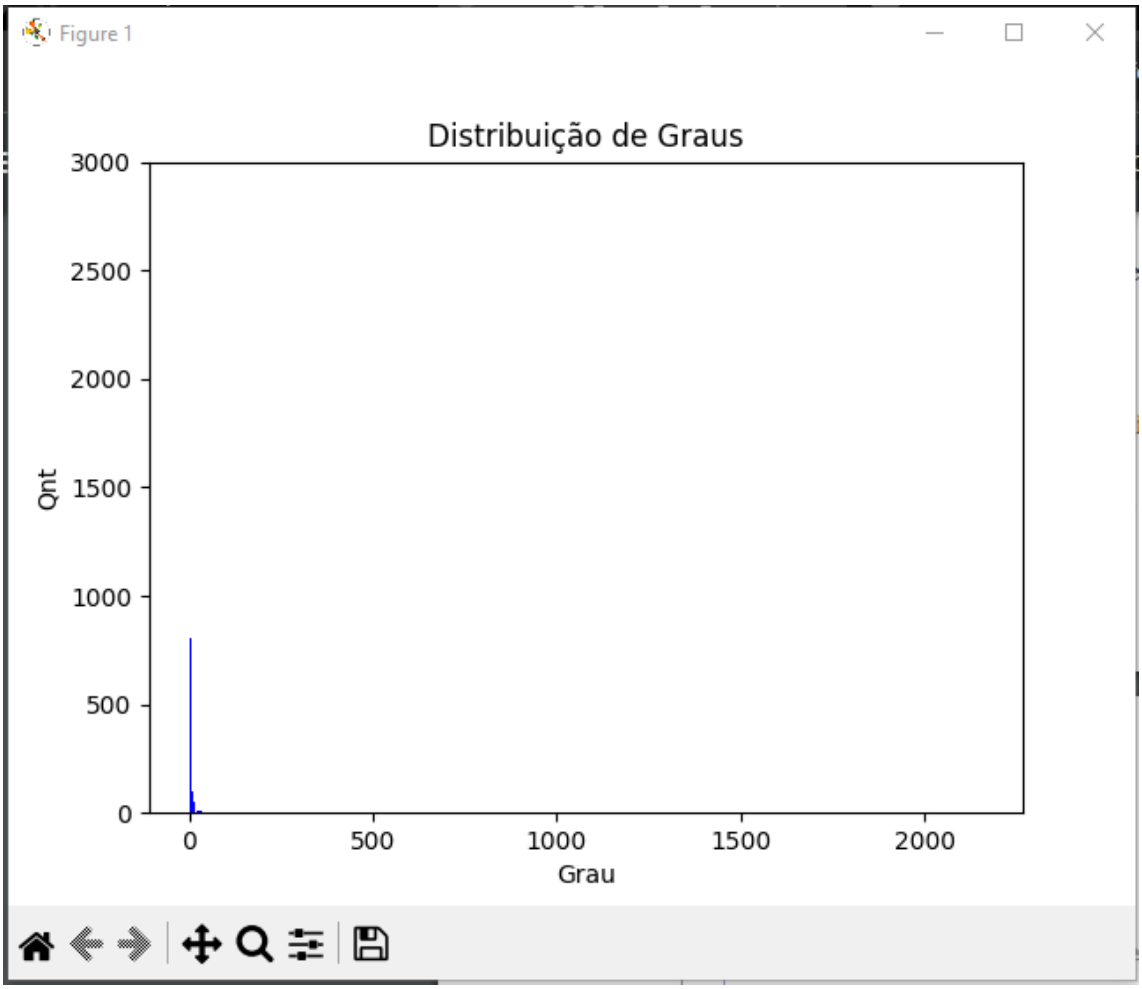
Ao analisarmos os tempos de execução para a construção de cada uma das representações, também se torna perceptível a diferença de eficiência entre elas. Conforme apresentados no gráfico abaixo, os tempos de execução para a matriz de adjacência são amplamente superiores aos da lista de adjacência, levando, em média, ´ 1 hora 15 minutos e 25 segundos para se criar a matriz e 0,27seg para se criar a lista de adjacências.

2.3. Componentes conexas

Este grafo tem **8826** componentes conexas. O menor deles possui **2** vértices, e o maior **33533** vértices. O arquivo com os componentes conexos desse grafo está no repositório do GitHub.

3. Estudo de caso 2 – As_graph

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



3.1. Componentes conexas

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 no repositório do GitHub.

3.2. Busca em largura

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.**

3.3. Diâmetro da internet

A partir dos resultados obtidos na execução anterior, pode-se concluir que o diâmetro desse grafo (maior distância entre qualquer par de vértices do grafo, será de **32384**. Pois representa o ultimo nível dessa busca por largura e profundidade.