UNIVERSIDADE ESTADUAL DE MARINGÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA

Trabalho I

Disciplina: Algoritmos em Grafos

Professor(a): Marco Aurélio Lopes Barbosa

Alunos:
Renan Augusto Leonel - 115138
Pedro Henrique de Melo Costa - 112653

1. Introdução

Este trabalho consiste no desenvolvimento de três algoritmos para gerarmos árvores aleatórias, baseados em: Passeio aleatório, algoritmo de Kruskal e algoritmo de Prim. Em cada algoritmo, produzimos uma árvore aleatória com n vértices, pois recebem como entrada um número n > 0.
Todos os algoritmos apresentam testes automatizados para todas as funções principais e auxiliares, com o intuito de evitar erros de implementação.

2. Desenvolvimento

Para implementarmos os algoritmos descritos anteriormente, foi utilizada a linguagem Python devido à sua sintaxe de simples compreensão.

Para representarmos os grafos e seus atributos, optamos por utilizar classes com listas de adjacências, com exceção somente na função `grafoCompletoPrim()`, onde optamos por utilizar matriz de adjacências para facilitar a implementação.

Também optamos por criar uma classe auxiliar para tratarmos apenas grafos com a estrutura necessária para o desenvolvimento do algoritmo de Kruskal, pois como o mesmo foi desenvolvido em etapas finais do trabalho, precisaríamos alterar grande parte do código já desenvolvido para modificarmos a estrutura `Grafo`, devido ao grande número de asserts para as funções anteriormente desenvolvidas.

3. Resultados

Todas as execuções foram realizadas em um computador com as seguintes configurações:

Processador Intel(R) Core(TM) i5-7500 CPU @ 3.40GHz 3.40 GHz
RAM 8,00 GB
GPU NVIDIA GeForce GTX 1060 6GB
Sistema operacional de 64 bits, processador baseado em x64

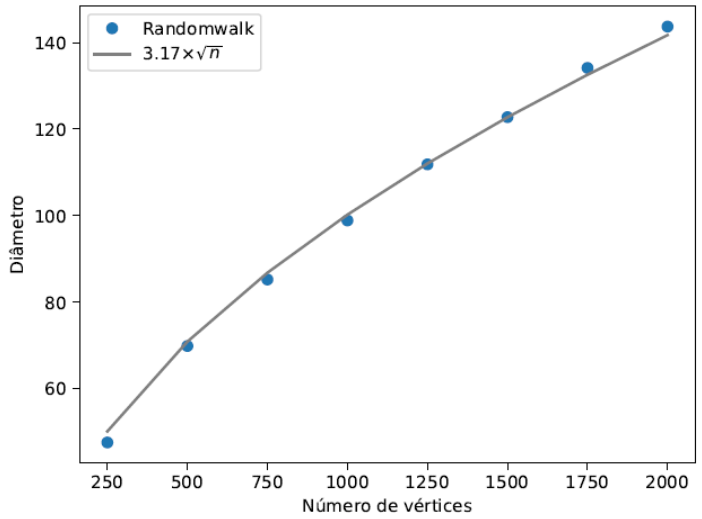
Priorizamos a execução de todo o código em apenas uma máquina para evitar divergências no tempo de execução. Obtemos, em média, os seguintes tempos de execução para as funções:

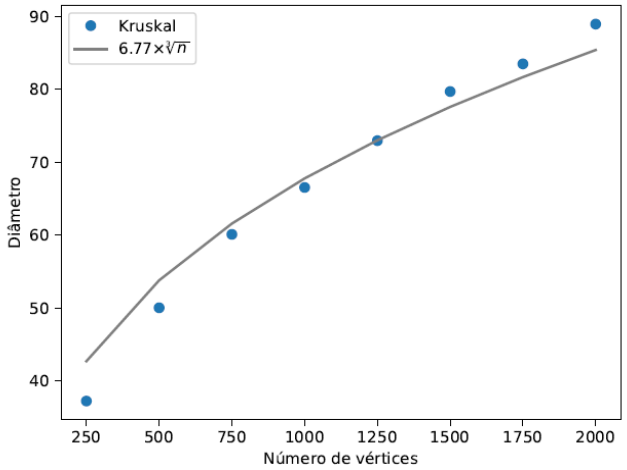
Random Walk: 67.88 segundos
Kruskal: 5578.03 segundos
Prim: 3922.41 segundos

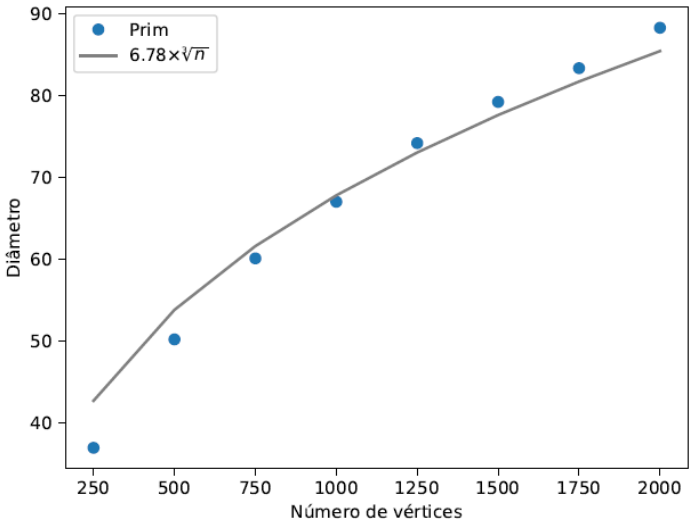
	250	500	750	1000	1250	1500	1750	2000
Random Walk	1.35s	3.34s	6.10s	7.79s	10.05s	12.53s	14.56s	15.94s
Kruskal	18.39s	90.04s	222.76s	424.86s	688.37s	967.36s	1522.42s	2083.06s
Prim	19.21s	84.56s	191.26s	326.50s	479.48s	707.76s	1001.49s	1254.04s

Com isso, podemos avaliar que o algoritmo de Prim executa mais rápido do que o algoritmo de Kruskal conforme o número de vértices aumenta.

Após executarmos os algoritmos, geramos os 3 gráficos a seguir:







Analisando os três gráficos em conjunto, observamos que a execução se manteve na curva média durante a maior parte da execução, porém para 250 vértices e 2000 vértices nos algoritmos de Prim e Kruskal, obtivemos uma divergência um pouco maior do que nos demais casos, onde se afastou bastante da curva desejada.

4. Descrição da experiência

Durante o desenvolvimento do trabalho, encontramos algumas dificuldades quanto à implementação, pois nunca havíamos trabalhado com a linguagem Python anteriormente, porém essa dificuldade foi superada rapidamente com algumas pesquisas de sintaxes e bibliotecas nativas da linguagem. Também tivemos dificuldades para reduzir o tempo de execução de alguns algoritmos e funções auxiliares, para que se aproximasse do tempo ideal. Para isso, procuramos alterar algumas funções como `pop()` para `popleft()`, pois com pequenas alterações como esta, podemos reduzir um número considerável de execuções, transformando funções lineares em constantes, etc. Por fim, vale ressaltar que tivemos dificuldades na implementação dos algoritmos de Prim e Kruskal, pois requer um nível alto de lógica de programação. Porém, conseguimos sanar estas dificuldades revendo as aulas gravadas da disciplina e concluir o trabalho.

Podemos destacar a importância de se atentar ao tempo de execução de funções simples, que usamos no cotidiano, mas que poderíamos substituir por outras de melhor desempenho e obter um ganho de tempo surpreendentemente alto, como o caso da função `popleft()` citada anteriormente.

Ao final do trabalho, podemos concluir que as implementações auxiliaram na compreensão da importância dos algoritmos em grafos para a computação como um todo, principalmente quanto ao tempo de execução de algoritmos. Por fim, gostaria de agradecer ao professor por ter sido flexível quanto às datas de entrega dos trabalhos, pois isso foi essencial para que o mesmo pudesse ser desenvolvido da melhor maneira possível e entregue na data desejada.