Alunos: Caio Marquiafave Carregari / Clidenor Barbosa de Melo Neto

Relatório Tarefa 1 – Teoria de Grafos

A priori, os resultados obtidos na tarefa se demonstraram satisfatórios. Primeiramente, a IDE em que nosso projeto foi idealizado, foi o VScode e tudo foi compilado no terminal no Ubuntu. Em quesito ao código, em primeiro lugar, criamos uma struct denominada “flor”, para salvar cada característica presente no arquivo IrisDataset.csv (sepal.length, sepal.width, petal.length, petal.width). Após ser lido, criamos a função denominada “distância\_euclidiana” para realizar os cálculos de distância. Após isso, criamos a função “distância\_normalizada”, em que utilizamos de outra função “maior\_menor” para criar uma matriz com as distâncias normalizadas. Após passar por essas funções, nosso programa cria um arquivo denominado “TEG.csv” que representa a matriz de adjacência criada pelas funções (além do cabeçalho requisitado). Importante ressaltar que, o valor, inicialmente 0.3, foi alterado para 0.1, a fim de garantir uma matriz de confusão mais conexa e íntegra. Além disso, os valores das distancias euclidianas e distâncias normalizadas menores (denmenor e demenor) são zeradas pois as flores presentes nas linhas 103 e 144 são iguais.

Após a criação da matriz de adjacências, adaptamos um arquivo em Python, “grafo\_3D.py” para ler uma matriz de adjacências (ao invés de uma lista de adjacências) e, com isso, cria-se um documento em html, denominado “grafo\_3D.html”. Seguindo, para criar as clusters, desenvolvemos um programa de alocação dinâmica de memória capaz de separar as flores em seus devidos grupos (que é controlado pelo limiar definido na criação da matriz de adjacências). Dessa forma, o programa separa os grupos, calcula o centro do grupo 1 (Setosas) e grupo 2 (Não setosas) e, para os outros grupos criados (caso tenha), seus vértices são adicionados nos grupos 1 ou 2, recalculando seus novos centros novamente. Após isso, ao finalizar os cálculos dos centros, verificamos a acurácia dos grupos gerados e mostramos a matriz de confusão criadas, com os valores de TN, TP, FN e FP.

Figura 1: Grafo plotado com limiar 0.3

Uma imagem contendo Gráfico

Descrição gerada automaticamente

Figura 2: Grafo plotado com limiar 0.1 (escolhido)

Uma imagem contendo Gráfico

Descrição gerada automaticamente

Figura 3: Grafo plotado com limiar 0.08

Diagrama

Descrição gerada automaticamente com confiança média

Figura 4: Grafo plotado com limiar 0.05

Diagrama

Descrição gerada automaticamente

Para a conclusão, é de suma importância ressaltar que: com um limiar muito pequeno, ou limiar muito grande, a acurácia do nosso programa é bem afetada. Recomendamos valores entre 0.2 e 0.1 para obter um resultado mais satisfatório e, assim, gerar uma matriz mais precisa. Além disso, para executar nosso programa (Em C) recomendamos usar o terminal do Ubuntu, digitando:

Comando 1: gcc Trabalho.c -o Trabalho

Comando 2: ./Trabalho

Além disso, para executar o grafo plotado em Python, digita-se:

Comando 1: python3 grafo\_3D.py

(O programa vai gerar um HTML, importante rodar o programa num ambiente controlado onde seja fácil a manipulação de arquivos.)

Dessa forma, o trabalho se mostrou complexo, porém, motivou o aprendizado, tanto sobre programação, quanto sobre grafos e suas características.