Universidade Federal de Uberlândia
Faculdade de Computação
FACOM32402 – Estrutura de Dados II
2025/1
Prof. Renato Pimentel

Trabalho 01 – Grafos (15 pontos)

Prazo final de entrega: 24/07/2025 23:59.

Objetivo e especificação

O objetivo do presente trabalho consiste em implementar uma ED para grafos, considerando-se a representação por lista de adjacências com encadeamento dinâmico, ou seja, dado um vértice, sua lista de adjacência será dada por uma lista dinâmica encadeada. O *array* dos vértices pode ser feito como no TAD visto em sala, mas pode ser aumentado sob demanda (veja a primeira função abaixo). A partir de tal implementação, devera ser analisada uma rede real com relação a aspectos como número de componentes conexas, vértice com maior grau, comprimento médio do menor caminho, e assim por diante.

A implementação consiste num TAD opaco – implementar em linguagem C – cuja interface deve disponibilizar, dentre outras, as seguintes funções:

- `int insereVertice(Grafo *G, char *label)`. Insere um vértice no grafo (usando alocação dinâmica – `malloc()` ou `calloc()` aumente em 50% o tamanho do *array* dos vértices caso o mesmo lote). Retorna 1 se inserção bem-sucedida ou 0, caso contrário.

- `int insereAresta(Grafo *G, int u, int v, float w)`. Insere a aresta (u, v) no grafo, com peso w, se o mesmo é ponderado. Se grafo não é dígrafo, trate internamente na função, sem recursão, fazendo a inserção também de u na lista de adjacências de v. Retorna 1 se inserção bem-sucedida ou 0, caso contrário.

- `int removeAresta(Grafo *G, int u, int v)`. Remove a aresta (u, v) do grafo, caso exista. Se grafo não é dígrafo, trate internamente na função, sem recursão. Retorna 1 se remoção bem-sucedida ou zero caso contrário.

- `int grau(Grafo *G, int v)`. Retorna o grau do vértice v, caso exista, ou -1, caso contrário.

- `float grauMedio(Grafo *G)`. Retorna o grau médio do grafo, ou seja, a média dos graus de todos os seus vértices, ou -1 se grafo vazio. Caso prefira, use `double` ao invés de `float`

- `int grauMax(Grafo *G, int *v)`. Similar à anterior, retorna o maior grau de todos os vértices da rede, armazenando em v qual o índice do vértice que possui tal grau.

Devem também ser implementadas 2 funções adicionais, para calcular:

- Qual o menor caminho médio do grafo. O menor caminho médio de um grafo é a média das distâncias entre todos os pares de vértices do grafo. A distância entre dois vértices é o menor caminho entre ambos. Para isso, podem ser usados algoritmos como o Dijkstra visto em aula. Se o grafo for ponderado, considerar o peso das arestas.

- Qual é a assortatividade da rede. **Assortatividade** é uma medida de grafos que representa a tendência ou não de vértices se conectarem ou não a outros vértices com grau semelhante. Mais detalhes em https://pt.wikipedia.org/wiki/Assortatividade.

A aplicação será usada para analisar uma rede real com pelo menos $n = |V| = 1000$ vértices, dentre as disponíveis no repositório https://networkrepository.com/index.php. As medidas calculadas acima devem corresponder aos valores mostrados no repositório, quando disponíveis.

Cada grupo deverá entregar, além do código e os arquivos de entrada utilizados, um relatório de no máximo 2 páginas A4, descrevendo a motivação para a escolha da(s) rede(s) utilizada(s), detalhes e dificuldades encontradas na implementação e uma breve seção de resultados. No relatório, incluir também o papel de cada membro do grupo em seu desenvolvimento.

Na entrega, submeter: o(s) arquivos-fonte (.c, .h), o relatório (**formato PDF somente)** e a base de dados (pode incluir um texto com o link para o grafo escolhido no repositório caso a rede seja muito grande). **Não zipar a entrega**.

Grupos e definição das redes escolhidas

Cada grupo deve conter no mínimo três e no máximo cinco integrantes. Até a data de 10/07, 23:59, devem ser divulgados, em resposta à postagem no Teams a ser criada pelo professor, qual(is) as rede(s) escolhida(s) por cada grupo, além de seus integrantes. **Os grupos formados deverão ser mantidos para o segundo trabalho prático.**

Entrega

Será criada uma tarefa no Teams, com base nos grupos definidos, e a entrega deverá ser feita, por um dos integrantes do grupo, usando tal tarefa, até a data de 24/07/2025, impreterivelmente. Não serão aceitos atrasos, sob qualquer justificativa. O grupo que não entregar no prazo, ou o grupo que, durante a avaliação pelo professor, seja constatado na implementação plágio ou uso de código pronto de outras fontes, terá sua nota de trabalho integralmente anulada.

Ponto extra: receberá um ponto extra o grupo que entregar a implementação que use o menor número possível de linhas de código, comparado aos demais. Atente-se que, além das rotinas acima, devem ser implementadas funções para inicializar o grafo, gerar o mesmo a partir dos arquivos da base do repositório e liberá-lo da memória ao final de sua utilização.