Universidade Federal de Lavras

Ciência da Computação

Prof. Douglas Aquino T. Mendes

Algoritmos em Grafos

Atividade de Implementação

1. **Objetivo:**

   Implementar algoritmos para análise de propriedades de grafos, avançando gradualmente à medida que a disciplina progride.

2. **Descrição:**

   a. Os alunos serão desafiados a implementar um **sistema modular** para análise de propriedades de grafos armazenados em arquivos de texto. **A implementação será progressiva**, com novas funcionalidades adicionadas conforme o avanço da disciplina. A estrutura básica consistirá em um menu interativo onde o usuário poderá selecionar a propriedade do grafo que desejam analisar.

   b. Os alunos são livres para decidir a linguagem de programação a ser utilizada.

   c. Os alunos possuem 60 dias para implementar as funções sugeridas, e são livres para adicionar qualquer outra a qual considere relevante/interessante.

   d. A avaliação se dará em cima da porcentagem de funções requeridas implementadas.
      - Para cada função requerida, pode-se escolher se a implementação utilizará matriz ou lista de adjacência.
      - Para cada função requerida, pode-se escolher qual o método/algoritmo a ser utilizado.

- Funções extras ou a opção de resolver o mesmo problema com algoritmos diferentes (exemplo: gerar árvore geradora mínima com Kruskal ou Prim?) ultrapassam o valor total do trabalho.
- A nota total do trabalho não excede 113%.

e. A atividade poderá ser realizada em trios, duplas ou sozinho.

f. Todo código desenvolvido deverá ser enviado para um repositório de códigos online, **GitHub**, com o devido *commit* resumindo o que foi implementado na versão atual.
   - https://github.com/git-guides/git-commit
   - Os commits serão utilizados para averiguação de participação dos alunos. Alunos que não participarem na implementação do código, não receberão pontuação.
   - O usuário SrAquino (e-mail: douglasaquino4@gmail.com) deverá ser adicionado como colaborador do projeto.

g. Todas as funções devem possuir um comentário javadoc
   - /** Essa função recebe a entrada em tal formato e responde tal coisa */
     - https://pt.stackoverflow.com/questions/6733/para-que-serve-o-c%C3%B3digo-na-linguagem-java

h. O grafo a ser analizado estará em um arquivo .txt, e deve ser representado como no exemplo a baixo, seguindo as regras de chaves '{' '}' para representar o conjunto, e finalizar com ponto e virgula ';', as arestas só precisam ser representadas uma vez em caso de grafos bidirecionais:

V = {a,b,c,d}**;** A = {(a,b,-2),(b,c,3),(c,d,-4),(d,a,5)}**;**

Onde V são os vértices e A é o conjunto de arestas ponderadas do grafo, no formato (vértice de partida, vértice de chegada, peso). Grafos não ponderados apenas não possuem o terceiro termo. Essa padronização será importante para os testes de avaliação.

3. **Etapas:**
    a. Implementação básica:
        - O programa deve perguntar se o grafo é direcionado ou não.
            - Caso seja não direcionado, o programa deve adicionar as arestas.

        - O programa deve fazer a leitura de um arquivo .txt
            - Identificar os vértices e arestas.
            - Retornar erro caso algo esteja incorreto na representação do grafo no arquivo
                - Exemplos:
                    i. Chaves sem fechamentos
                    ii. Arestas sem parenteses ou sem virgulas separando os vértices.
                    iii. Arestas indo para vértices que não pertencem ao grafo.

        - O programa deve ser modular, e ao chamar cada função disponível no menu, o programa deve mostrar o tempo que levou para executar a função.
            - https://www.youtube.com/watch?v=9GaDSKZ3sM4

Menu:

1. Verificar
    a. Quantidade de vértices
    b. Quantidade de arestas
    c. Conexo
    d. Bipartido
    e. Euleriano
    f. Hamiltoniano
    g. Cíclico
    h. Planar

2. Listar
    a. Vértices
    b. Arestas
    c. Componentes conexas
    d. Um caminho Euleriano
    e. Um caminho Hamiltoniano
    f. Vértices de articulação
    g. Arestas ponte

3. Gerar
    a. Matriz de adjacência
    b. Lista de adjacência
    c. Árvore de profundidade
    d. Árvore de largura
    e. Árvore geradora mínima
    f. Ordem topológia (Esta função não fica disponível em grafos não direcionado)
    g. Caminho mínimo entre dois vértices (Esta função não fica disponível em grafos não ponderados)
    h. Fluxo máximo (Esta função não fica disponível em grafos não ponderados)
    i. Fechamento transitivo (Esta função não fica disponível em grafos não ponderados)

Cronograma (26/04 à 26/06):

- A progressão na atividade será verificada a cada 10 dias:
  - ~~06/05 (Inicialização do projeto e representações do grafo)~~
  - ~~16/05 (Remoções e inserções)~~
  - ~~26/05 (Verificações e Árvores)~~
  - **Aguardar retorno da greve.**
  - ~~24/06 à 29/06 (Apresentação dos trabalhos)~~

Horários de atendimento: (Agendar)

Quartas das 17:00 às 19:00

Quintas das 17:00 às 21:00

E-mail: douglas.mendes@ufla.br