UNIVERSIDADE FEDERAL DE SANTA CATARINA

INE5413 - GRAFOS

RELATÓRIO DA ATIVIDADE 2

BRENDA SILVA MACHADO

2023.1

1. Código Fonte

A linguagem de programação escolhida para a realização desta atividade foi Python, voltada para a Orientação Objetos. Não foi necessário o uso de nenhuma biblioteca. O código fonte da atividade está disponível no GitHub, no link: https://github.com/Brenda-Machado/grafos

2. Componentes Fortemente Conexas

A primeira questão pede um algoritmo para identificar as componentes fortemente conexas de um grafo. Para o desenvolvimento deste algoritmo, foram utilizadas estruturas de classe e variáveis do tipo lista, uma vez que apresentam simplicidade e facilidade de manipulação.

Para os métodos relacionados ao DFS, foi criada uma classe própria, a qual tem variáveis tipo lista para armazenar os vértices visitados e tipo dicionário para as variáveis tempo, fim e antecessores, graças à estrutura de key, que permite identificar a qual vértice pertence o dado armazenado.

3. Ordenação Topológica

A segunda questão pede um algoritmo que realize a ordenação topológica de um grafo. Para isso, a variável O, que possui a ordenação, foi implementada como lista, a qual foi invertida ao final para os prints, para simular o mecanismo de fila. A variável T foi implementada como dicionário, para permitir a consulta de keys, o que facilitou a implementação do método visit_ot().

Visitados também foi uma lista na implementação deste algoritmo, uma vez que apenas armazena o rótulo dos vértices como uma variável de controle.

4. Árvore Geradora Mínima

A terceira questão pedia a implementação de um algoritmo que retornasse árvores geradoras mínimas, dado um gráfico de entrada. O algoritmo escolhido como base para a implementação foi o algoritmo de Prim.

As variáveis antecessores e chave foram implementadas como dicionários, justamente pelo mecanismo de chave já mencionado acima. Chave como dicionário facilitou a implementação, uma vez que o dicionário já provém vários mecanismos de consulta necessários a essa variável ao longo do funcionamento do algoritmo.

A estrutura de prioridade foi implementada como heap queue (fila de prioridade). Essa estrutura foi escolhida graças à facilidade de manipulação, assim como ao fato de se organizar como uma árvore binária, o que permite um mecanismo de busca de menor complexidade. Esse heap recebeu o peso e a aresta considerada, e o argumento de menor peso foi facilmente encontrado com o método heappop.