Atividade III - Grafos

Caio Ferreira Cardoso - 22200352
Gabriel Reimann Cervi - 22204117
Lucas Brand Samuel Martins - 22202622

Execução

Para rodar o programa, o usuário deve rodar em seu terminal:

```
python main.py nome_do_arquivo numero_da_questao
```

O número da questão deve estar entre 1 e 3. O nome do arquivo deve ser igual ao nome do arquivo presente na pasta “Testes”.

Questão 1 - Fluxo Máximo

Nesta questão, é utilizado o algoritmo de Fluxo Máximo apresentado no livro da matéria, utilizando busca em largura e Edmonds-Karp.

A BFS é usada para encontrar um caminho aumentante do vértice fonte (source) ao vértice sumidouro (sink) no grafo residual.

Durante a BFS, são utilizadas listas para marcar os vértices visitados (visited), para armazenar os ancestrais dos vértices (ancestor), e uma fila (queue) que auxilia na lógica de visitação dos vértices.

Um grafo residual é construído inicialmente copiando o grafo original. São adicionadas arestas invertidas (com capacidade 0) ao grafo residual para permitir a possibilidade de "retornar" fluxo em passos posteriores do algoritmo.

O algoritmo executa iterações sucessivas de BFS para encontrar caminhos aumentantes. Para cada caminho aumentante encontrado, o fluxo máximo ao longo desse caminho é determinado. As capacidades das arestas ao longo do caminho são ajustadas no grafo residual para refletir o fluxo que foi enviado.

Questão 2 - Emparelhamento

Nesta questão é apresentada uma implementação do algoritmo de Hopcroft-Karp conforme o apresentado no livro da disciplina, retornando a quantidade de emparelhamentos e quais são eles.

Existem duas funções que auxiliam na execução:

- **BFS:** busca em largura, utilizada para verificar se existe caminho aumentante alternante no grafo.
- **DFS:** busca em profundidade, usada para ver se há um caminho aumentante alternante a partir de um dado vértice v.

O algoritmo cria duas listas dos pares dos vértices, uma para os vértices do grupo X e outra para os do grupo Y, nas quais o valor na qual o valor em pares[i] será o index do vértice com o qual o vértice i está pareado, exceto no index 0, que representa o vértice nulo. Cria-se também uma lista das distâncias, que é usada para ver se há caminhos aumentantes.

É feita uma busca em largura para ver se tem caminho aumentante, então busca-se um vértice livre e nesse vértice fazemos uma busca em profundidade para ver se há um caminho aumentante alternante que sai dele. Isso se repete até que não haja mais nenhum caminho aumentante.

Um detalhe importante e limitador da nossa implementação é que os grupos X e Y não são inicialmente definidos, apenas assume-se que dada uma quantidade de vértices |V| = 2n, os primeiros n vértices pertencem a X e os n vértices restantes pertencem a Y. Isso acaba por deixar o algoritmo limitado a apenas grafos que seguem essa regra, não funcionando para grafos onde |X| =/= |Y|.

Questão 3 - Coloração de Grafos

A implementação segue a apresentada no caderno de anotações da disciplina sobre coloração de grafos com a adição de um método responsável por gerar os subconjuntos de dos vértices de dado grafo.