UNIVERSIDADE FEDERAL DE ALAGOAS
INSTITUTO DE COMPUTAÇÃO

Teoria dos Grafos

Lista de Exercícios

Prof. Rian Gabriel Pinheiro

2024.1

Instruções

(a) As listas podem ser resolvidas em dupla, neste caso deve-se entregar apenas um trabalho!
(b) Mencione os teoremas e propriedade usadas para justificar usas afirmações.
(c) Qualquer modificação das duplas deve ser informada.
(d) As resoluções devem ser entregues escritas à mão no dia da prova.
(e) Qualquer tentativa de fraude implicará em nota *zero* na parte correspondente.
(f) Cada questão correta irá incrementar 0,05 ponto na respectiva prova.

Parte I.
1ª VA

1. Grafo e Subgrafo; Arvores e Conectividade

1.1 . Prove que para toda árvore com n vértice, o número de arestas $m = n - 1$.

1.2 . Mostre que há onze não-isomorfos grafos simples com quatro vértices.

1.3 . Quantas arestas possui o grafo K_n?

1.4 . Prove que:

- $|E(K_{n_1,n_2})| = n_1 n_2$;
- se G é um grafo simples e bipartido, então $|E(G)| \le n^2/4$

1.5 . Seja $d(v)$ o grau de um vértice v pertencente ao grafo $G = (V,E)$. Mostre que $\sum_{v\in V} d(v) = 2|E|$.

1.6 . Prove que em todo grafo, o número de vértices com grau ímpar é par.

1.7 . Prove que um grafo é bipartido se e somente se não possuir ciclo ímpar.

1.8 . Prove que se existe um (u,v)-passeio (sem repetição de arestas) no grafo G, então existe um (u,v)-caminho simples (sem repetição de vértices) em G.

1.9 . Prove que um grafo conexo G é uma árvore se e somente se toda aresta de G for um ponte.

1.10 . Encontre um grafo com 5 vértice na qual a clique máxima e o conjunto independente máximo são menores que 3.

2. Euleriano e Hamiltoniano; Clique e Conjunto Independente

2.1 . Considere as 21 peças do jogo de dominó que não são "bombas". Cada uma dessas peças corresponde a um subconjunto de cardinalidade 2 do conjunto $\{0;1;2;\ldots;6\}$. É permitido "encostar" uma peça $\{i,j\}$ numa peça $\{j,k\}$ de forma a produzir a sequência (i,j,j,k). Pergunta: É possível formar um "roda" que contenha todas as 21 peças? E se eliminarmos todas as peças que contêm "6"?

2.2 . Prove que o grafo de Petersen não é hamiltoniano.

2.3 . Mostre que, se um grafo conexo G não orientado for euleriano, seu conjunto de arestas poderá ser particionado em ciclos disjuntos.

2.4 . Mostre que K_{ij} possui um ciclo hamiltoniano se e somente se $i = j$.

2.5 . Prove que, em um grafo conexo G em que todos os vértices possuem grau par, não existe ponte em G.

2.6 . Seja G um grafo conexo não Euleriano, mostre que é possível adicionar um único vértice em G, juntamente com algumas aresta ligadas a este novo vértice, de modo a tornar G Euleriano.

2.7 . Mostre que qualquer grafo com no mínimo dois vértice contém no mínimo dois vértices com o mesmo grau.

2.8 . Qual a relação entre o problema da clique máxima e o problema do conjunto independente máximo? Como é possível usar um algoritmo que resolve um dos problemas para resolver o outro?

2.9 . Prove que todo grafo com mais que seis vértices contém no mínimo ou uma clique com no mínimo três vértice ou um conjunto independente com no mínimo três vértices.

2.10 . Encontre um conjunto independente máximo na grade p-por-q.

3. Coloração e Emparelhamento

3.1 . Mostre que o grafo de Petersen é 4-cromático de aresta, isto é, mostre que as arestas do grafo de Petersen não podem ser colorias com apenas 3 cores.

3.2 . Mostre que toda árvore pode ser 2-colorida.

3.3 . Mostre que os itens são equivalentes:

a) G é bipartido.
b) G pode ser 2-colorido.
c) G não tem ciclo ímpar.

3.4 . Seja Δ o grau máximo de qualquer vértice em um grafo G . Prove que podemos colorir G com $\Delta + 1$ cores.

3.5 . Mostre que toda grade é bicolorível.

3.6 . 5 Suponha que um grafo G tem um emparelhamento perfeito. Mostre que n é par.

3.7 . É verdade que todo grafo regular tem um emparelhamento perfeito?

3.8 . Mostre que é impossível cobrir um tabuleiro de xadrez 8×8 usando peças retangulares 1×2 se duas casas quinas opostas forem removidas.

3.9 . Calcule um emparelhamento máximo em um grafo 3-regular dotado de circuito hamiltoniano.

3.10 . Em uma companhia, n trabalhadores $X_1, X_2, \ldots, X_n$ estão disponíveis para n tarefa $Y_1, Y_2, \ldots, Y_n$. Cada trabalhador X_i possui uma aptidão p_{ij} para desenvolver a tarefa Y_j. Todo os homens devem ser alocados a exatamente uma tarefa — um homem por trabalho e uma tarefa por homem. Modele o problema usando grafo e proponha uma solução com a máxima soma das aptidões.

4. Grafos Planares; Grafos Direcionados e Fluxo

4.1 . Três casas devem ser interligadas às companhias de luz, água e gaz. No entanto, as tubulações não podem se cruzar. Solucione o problema ou mostre que ele não possui solução.

4.2 . Mostre que o grafo de Petersen não é planar.

4.3 . Mostre que um grafo é planar se e somente se cada uma de suas componentes conexas for planar.

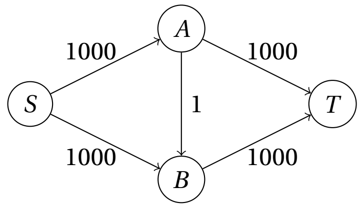
4.4 . Seja e uma ponte de um grafo conexo G. Mostre que G é planar se e somente se $G - e$ é planar.

4.5 . Seja G o grafo dos estados do Brasil. Mostre que $\chi(G) = 4$, em que $\chi(G)$ é o número cromático de G.

4.6 . Mostre que $\sum_{v \in V} d^-(v) = |E| = \sum_{v \in V} d^+(v)$.

4.7 . Dê um exemplo mostrando que o fluxo máximo é igual ao corte mínimo.

4.8 . Considere o grafo abaixo. Mostre que, se o algoritmo de Ford-Fulkerson é executado sobre este grafo, uma escolha sem cuidado de atualizações pode fazê-lo realizar 2000 iterações.



4.9 . Seja G um grafo planar com n vértices, m arestas, c componentes conexas e f faces. Mostre que $n - m + f - c = 1$.

4.10 . Seja G um grafo com 11 vértice, mostre que G ou $\overline{G}$ não é planar.

Parte II. 2ª VA

Implementação

Regras:

- O trabalho consistem em implementar 5 algoritmos em grafos em C ou C++.
- Os algoritmos serão i) `kosaraju`, ii) `prim`, iii) `kruskal`, iv) `dijkstra` e v) um algoritmo de sua escolha.
- O último algoritmo escolhido não poderá ser: busca em largura, busca em profundidade e ordenação topológica.
- Cada algoritmo deve ter sua própria pasta, com o código e um makefile.

Árvore Geradora Mínima
Para o Problema da Árvore Geradora Mínima a entrada deve possuir o seguinte formato:

```
Lista de adjacencia de um grafo G com 6 vértice e 8 arestas (3a coluna é o peso da aresta)
6 8
1 2 5
1 3 4
1 4 2
1 6 6
2 4 1
2 5 7
3 5 6
4 6 1
```

O nome binário deverá ser o nome do algoritmo (e.g., `prim`). Para o problema da AGM os algoritmos devem possuir os seguintes parâmetros:

```
-h                      : mostra o help
-o <arquivo>            : redireciona a saida para o ‘‘arquivo’’
-f <arquivo>            : indica o ‘‘arquivo’’ que contém o grafo de entrada
-s                      : mostra a solução
-i                      : vértice inicial (para o algoritmo de Prim)
```

Exemplos de execução:

```
Calcula o custo da AGM com o grafo de entrada "arquivo-entrada.dat" e vértice inicial 1.
$ ./prim -f arquivo-entrada.dat -i 1
14
```

```
Imprime a árvore do exemplo anterior
$ ./prim -f arquivo-entrada.dat -i 1 -s
(1,3) (1,4) (2,4) (3,5) (4,6)

Note que a solução não precisa estar em ordem.
(4,1) (6,1) (4,2) (3,4) (5,6)
Também seria uma solução válida.
```

Componentes fortemente conexas
Para o problema de componentes fortemente conexos o algoritmo de kosaraju deve possuir os seguintes parâmetros:

```
 -h                       : mostra o help
 -o <arquivo>             : redireciona a saida para o ‘‘arquivo’’
 -f <arquivo>             : indica o ‘‘arquivo’’ que contém o grafo de entrada
```

Exemplos de execução:

```
Imprime as componentes fortemente conexas do grafo

arquivo-entrada.dat
12 17
1 2
1 4
2 3
3 1
3 7
4 6
5 4
6 7
7 5
8 6
8 11
9 8
10 9
11 10
11 12
12 10
12 7


$ ./kosaraju -f arquivo-entrada.dat
8 9 10 11 12
1 3 2
7 6 4 5
```

Caminhos Mínimos
Para o problema de caminho mínimo o algoritmo de dijkstra deve possuir os seguintes parâmetros:

```
 -h                       : mostra o help
 -o <arquivo>             : redireciona a saida para o ‘‘arquivo’’
 -f <arquivo>             : indica o ‘‘arquivo’’ que contém o grafo de entrada
 -i                       : vértice inicial
```

Exemplos de execução:

```
Imprime a distância do vértice inicial 1 até os demais
$ ./djikstra -f arquivo-entrada.dat -i 1
1:0 2:3 3:4 4:2 5:10 6:3
```

Caso um vértice seja inalcançável a partir do vértice inicial, o algoritmo deve mostrar apresentar o valor -1. No site da disciplina, será disponibilizado um conjunto de arquivos de teste e suas repectivas saidas (`Bat1`).

Critérios de Avaliação.

- Cada algoritmo será avaliado por uma nota de 0 a 10.
- A nota final será a médias das notas.
- Os 4 primeiros algoritmos serão avaliados por duas baterias de testes `Bat1` e `Bat2`, em que `Bat1` será fornecido.

- O 5º algoritmo deve vir acompanhado de um conjunto com pelo menos 20 entradas de tamanhos similares às fornecidas em `Bat1`.
- O algoritmo receberá 10 ou 0 se passar por `Bat1` ou não.
- O algoritmo que passar por `Bat1` terá pontos descontados com os seguintes critérios:
  - Implementação ineficiente (-1 ponto).
  - Não passar por Bat2 (-3 pontos).
  - Não utilização das estruturas de dados corretas (-1 ponto).
  - Não possuir makefile (-1 ponto).
  - Não for estruturado em uma pasta própria (-1 ponto).
- O 4º algoritmo será avaliado de forma análoga só que sem a bateria de teste.