MC558 - Projeto e Análise de Algoritmos II

Lista de exercícios 04 - Conceitos e algoritmos em grafos

Esta lista é avaliativa e as respostas devem ser entregues em um arquivo .pdf ao professor via o Google Sala de Aula. Embora seja possível discutir os exercícios com seus colegas, as entregas devem ser feitas de forma individual.

1. Escolha um dos seguintes itens para entregar:

(a) Se um grafo possui um passeio fechado que passa exatamente única vez por cada aresta, dizemos que o grafo é **euleriano**. Suponha que em um grafo conexo G todo vértice tem grau par. Mostre que G é euleriano.

(b) Se um grafo possui um ciclo gerador, dizemos que é **hamiltonianoSuponha** que em um grafo $G = (V, E)$, com $|V| \geq 3$, todo par de vértices não adjacentes u, v satisfaz que $d(u)+d(v) \geq |V|$. Mostre que G é hamiltoniano.

(c) Considere um grafo G. Um **emparelhamento** de G é um conjunto $M \subseteq E$ de arestas, tal que cada vértice em V incide no máximo em uma aresta de M. Uma **cobertura** de G é um conjunto $S \subseteq V$ de vértices tal que $G - S$ não possui arestas. Suponha que M^* é um emparelhamento máximo de G e S^* uma cobertura mínima de G. Mostre que se G for bipartido $M^* = S^*$.

(d) Dado um grafo $G = (V, E)$, um emparelhamento $M \subseteq E$ **satura** um conjunto $S \subseteq V$, se cada vértice de S incide em uma aresta de M. M é um **emparelhamento perfeito** de G se satura V. Mostre que se G é bipartido e todo vértice tem o mesmo grau $k \geq 1$, então G possui um emparelhamento perfeito.

2. Escolha dois dos seguintes itens para entregar:

(a) Analise o tempo e mostre a correção do seguinte algoritmo para encontrar componentes fortemente conexas. O algoritmo 1 é a função principal e o 2 é o chamado recursivo para encontrar as componentes:

```
   input    : Grafo G
 1 begin
 2 |   for u ∈ V do
 3 |   |   cor[u] ← branco
 4 |   end
 5 |   tempo ← 0
 6 |   l ← 0
 7 |   S ← ∅
 8 |   for u ∈ V do
 9 |   |   if cor[u] = branco then
10 |   |   |   Find_CC(G, u)
11 |   |   end
12 |   end
13 end
```

Algorithm 1: *Main*

```
   input  : Grafo G, vértice u
 1 begin
 2     cor[u] ← cinza
 3     tempo ← tempo + 1
 4     d[u] ← tempo
 5     menor[u] ← tempo
 6     S.push(u)
 7     for v ∈ Adj[u] do
 8         if cor[v] = branco then
 9             Find_CC(G, v)
10             menor[u] ← min(menor[u], menor[v])
11         else
12             if cor[v] = cinza then
13                 menor[u] ← min(menor[u], menor[v])
14             end
15         end
16     end
17     cor[u] ← preto
18     if menor[u] = d[u] then
19         l ← l + 1
20         v ← S.pop()
21         while u ≠ v do
22             comp[v] ← l
23             v ← S.pop()
24         end
25         comp[v] ← l
26     end
27 end
```

Algorithm 2: *Find_CC*

(b) Projete um algoritmo eficiente que determine se um grafo é euleriano ou não (ver definição na questão anterior). Em caso de ser euleriano, imprima um passeio fechado que passe exatamente uma vez por cada aresta do grafo. Analise a complexidade do seu algoritmo e prove a correção.

(c) O diâmetro de um grafo é a maior distância entre qualquer par de vértices ($max_{u,v \in V} dist(u, v)$). Proponha um algoritmo eficiente para encontrar o diâmetro em de um grafo geral e outro para o caso do grafo ser uma árvore. Analise a complexidade e a correção dos seus algoritmos.

(d) Considere o seguinte problema: é dada uma árvore T com n vértices e são feitas m consultas para saber a distância entre pares de vértices. Proponha um pré-processamento em $O(n)$, que permita calcular a distância entre cada par de vértices u, v em $O(dist(u, v))$. Justifique a complexidade e mostre a correção da sua solução.

3. Crie um usuário no **beecrowd** (`https://judge.beecrowd.com/`). Use seu nome e sobrenome no campo *"NOME DE USUÁRIO"* do seu perfil, caso contrário não será possível identificar a suas entregas na plataforma.

Após criar o usuário, entre na disciplina **MC558 - 2024/2** (`https://judge.beecrowd.com/pt/disciplines/join/13244`), a chave de acesso é: **TffkJq8**

Complete pelo menos 10 pontos em nível de dificuldade na lista:

- **Algoritmos em grafos** - `https://judge.beecrowd.com/pt/homeworks/view/44605`.

Para cada solução submetida, adicione no arquivo .pdf que deve ser entregue: a definição do problema ser resolvido (relação entre a entrada e saída), explicando sucintamente o algoritmo empregado, sua complexidade e o motivo de funcionar para esse problema.