1^a prova de Algoritmos e Teoria dos Grafos

2° semestre de 2004

---

(1) LEIA COM ATENÇÃO TODA A PROVA

(2) EXPLIQUE todas as suas respostas.

(3) Entender o enunciado das questões faz parte da prova.

(4) Você pode usar todos os teoremas e algoritmos vistos nas aulas que cobrem os temas da prova. Nesses casos, escreva o que você está usando.

(5) A prova é sem consulta.

---

1) (**10**) Formule em linguagem de teoria de grafos a seguinte sentença: Em qualquer grupo de duas ou mais pessoas, sempre existem duas pessoas com exatamente o mesmo número de amigos nesse grupo.
   Prove a veracidade da sentença.

2) (**10**) Pode um grafo com 4 vértices ter 3 vértices de grau 3 e um vértice de grau 1? Justifique.

3) (**10**) Queremos representar um grafo com n vértices e m arestas. Para $n = 50$ determine os valores de m para os quais a representação por listas de adjacências gasta menos espaço que por matriz de adjacências nos dois casos: (1) assuma que $p = 1$ bit, (2) repita para $p = 4$ bytes; onde p é o número de bits usados para armazenar o rótulo do vértice e o ponteiro no caso de listas ligada.

4) (**10**) Qual representação de grafos é mais eficiente para responder rapidamente se um dado vértice tem grau 0? Justifique.

5) (**10**) Escreva uma versão não recursiva para o algoritmo de busca em profundidade e analise a complexidade.

6) (**20**) Prove que todo grafo G conexo e sem circuitos tem $|V(G)| - 1$ arestas.

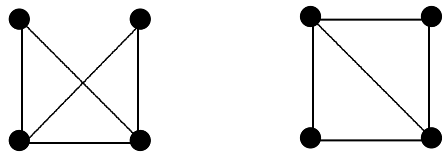
7) (**20**) Assuma que um grafo conexo $G = (V, E)$ é dado ou por sua matriz de adjacências A ou por uma lista de adjacências $adj[\,]$ (você terá que escolher). Escreva um algoritmo baseado na busca em largura que ao final da execução ou A ou $adj[\,]$ (dependendo da sua escolha) armazena um subgrafo conexo, sem circuitos e com V como conjunto de vértices. Escolha a representação (A ou $adj[\,]$) que torna o algoritmo mais eficiente para o caso em que $|E| = \alpha|V|^2$, $\alpha > 0$, e justifique a sua escolha. Explique o funcionamento do algoritmo.

8) (**10**) Escolha uma para resolver:
   (a) Se $\delta(G) \geq |V(G)| - 2$ então $\kappa(G) = \delta(G)$.
   (b) Se $\delta(G) \geq |V(G)|/2$ então $\lambda(G) = \delta(G)$.

Solução (1). Os vértices são as pessoas e duas pessoas amigas formam uma aresta. A sentença fica: Em todo grafo existem dois vértices com o mesmo grau.

Prova: Seja n o número de vértices. Se todos os graus dos vértices são distintos então $\delta = 0$ e $\Delta = n-1$ o que é uma contradição, pois o vértice de grau mínimo seria isolado e o de grau máximo ligado com todos os outros vértices, inclusive o isolado. □

Solução (2). Não, o grafo teria 5 arestas e as únicas possibilidades são os grafos isomorfos a:



□

Solução (3). Vimos que lista gasta $(4m+n)p$ bits e matriz n^2 bits.

$p = 1$ bit. $4m + n < n^2$, ou seja $m < (n^2 - n)/4 = 612,5$.

Lista de adjacências é mais econômica para grafos com 50 vértices e no máximo 612 arestas.

$p = 32$ bits. $(4m+n)32 < n^2$, ou seja $m < (n^2 - 32n)/128 = 7,03125$.

Lista de adjacências é mais econômica para grafos com 50 vértices e no máximo 7 arestas. □

Solução (4). No caso de matriz temos que percorrer a linha (ou coluna) do vértice dado. No caso de lista basta saber se a lista daquele vértice é vazia. Portanto, lista de adjacências é mais eficiente. □

Solução (5). O começo é como visto em sala, Algoritmo Percurso( ). Vamos descrever o Visite(k) Assumimos uma pilha P com as operações

`remove( ):` retorna e remove o elemento do topo da pilha P;
`empilha(v):` insere o vértice v no topo da pilha P;
`vazia( ):` devolve 1 se a pilha está vazia e 0 caso contrário.

Algoritmo 1 Visite(k)

```
1: empilha(k)
2: enquanto (!vazia( ))
3:     k = remove( )
4:     ordem[k] = ++cont
5:     para (t ∈ adj[k])
6:         se (ordem[t] == 0)
7:             empilha(t)
8:             ordem[t] = −1
```

Análise da complexidade: Para determinarmos os número de instruções efetuadas pelo algoritmo, observamos que

(a) o número de instruções no algoritmo Percurso é $O(|V|)$;
(b) todo vértice v de G entra na pilha se, e somente se, `ordem[v] == 0`: linhas 4 e 5 do algoritmo Percurso juntamente com a linha 1 do algoritmo 1, ou linhas 6 e 7 do algoritmo 1;
(c) quando um vértice v está na pilha `ordem[v] == -1`, exceto quando entra na pilha pela execução da linha 1 do algoritmo 1 mas nesse caso o vértice será removido logo (linha 3 do algoritmo) e o que importa é que nesse instante (entre linhas 1 — 3) a pilha não sofre alterações;
(d) quando um vértice v é removido da pilha `ordem[v]`>0 e todas as arestas em $E(v)$ são percorridas (linhas 4 e 5).

Assim, podemos concluir que todo vértice é inserido na pilha exatamente uma vez e cada aresta é percorrida duas vezes.

As arestas são percorridas na linha 5, portanto se usamos lista de adjacências o tempo do algoritmo é $O(|V| + |E|)$ e se usamos matriz o tempo do algoritmo é $O(|V|^2)$. □

Solução (6). Vimos em aula (proposição 6) que se $\delta(G) \geq 2$, então G contém circuito. Portanto, como G não contém circuito, $\delta(G) = 1$.

Vamos provar por indução em $|V(G)|$.

Num grafo com 1 vértice, o número de arestas é 0 (base da indução).

Vamos assumir que em todo grafo H conexo e sem circuitos com menos que $|V(G)|$ vértices, o número de arestas é $|V(H)| - 1$ (hipótese indutiva).

Tome $v \in V(G)$ um vértice de grau 1. O grafo $H = \big(V(G) - \{v\}, E(G) - E(v)\big)$ é conexo e sem circuitos e, pela hipótese indutiva, $|E(H)| = |V(H)| - 1$. Como $|E(v)| = 1$ temos $|E(G)| = |E(H)| + 1$ que implica $|E(G)| = (|V(H)| - 1) + 1 = |V(H)|$.

Mas $|V(H)| = |V(G)| - 1$, logo $|E(G)| = |V(G)| - 1$. □

Solução (7). A idéia do algoritmo é: numa busca em largura, cada vez que visitamos um vértice t que já foi visitado fechamos um circuito. Assim, se t foi re-visitado a partir de k a aresta tk pertence a um circuito, então removemos-a. Como a busca visita todos os vértices e como removemos as arestas de circuitos, no final temos um grafo conexo sobre V e sem circuitos.

Como $|E| = \alpha|V|^2$ a busca em largura usual tem a mesma complexidade para lista de adjacências e matriz de adjacências, $O(|V| + |E|) = O(|V|^2)$. A remoção de arestas é feita de forma mais eficiente em matriz, pois para a remoção de kt na lista, teremos que percorrer, no pior caso, toda a lista de t para remover k. Então a escolha é usar matriz de adjacências: □

Algoritmo 2 VISITE(k)

```
ENFILA(k)
ENQUANTO (!VAZIA( ))
  k = REMOVE( )
  ordem[k] = ++cont
  PARA (t = 1; t ≤ |V|; t++)
    SE (A[k][t] == 1) */ se kt e' uma aresta
      SE (ordem[t] == 0) */ se t nao foi visitado
        ENFILA(t)
        ordem[t] = −1
        pai[t] = k
      SENÃO */ se kt forma circuito, remover.
        SE pai[k] ≠ t
          A[k][t] = A[t][k] = 0
```

Solução(8). (a) Se $\delta(G) = 0$, então G é desconexo e $\kappa(G) = 0$. Temos G conexo e, ou $\delta(G) = |V(G)| - 1$ ou $\delta(G) = |V(G)| - 2$.

Se $\delta(G) = |V(G)| - 1 > 0$, então G é o grafo completo sobre $V(G)$ e, portanto, $\kappa(G) = |V(G)| - 1 = \delta(G)$.

Vamos assumir que $\delta(G) = |V(G)| - 2 > 0$.

Vamos provar por contradição que o grafo é $\delta(G)$-conexo. Suponha que existe um corte $U \subset V(G)$ com $|U| = \delta(G) - k = |V(G)| - (k+2)$ para algum k $(0 < k < \delta(G))$. Então $G - U$ é desconexo com $|V(G)| - |U| = k + 2$ vértices.

O grau mínimo em $G - U$ é $\delta(G) - |U| = k$, portanto, (visto em sala - proposição 6) $G - U$ contém um caminho de comprimento k. Um vértice de $G - U$ fica de fora desse caminho. Como o grau mínimo é > 0, esse vértices têm vizinhos no caminho logo o grafo $G - U$ é conexo, o que é um absurdo.

Com isso provamos que G é $\delta(G)$-conexo. Vimos em aula que (teorema 7) $\kappa(G) \leq \delta(G)$, donde concluímos que $\kappa(G) = \delta(G)$.

(b) Se $\delta(G) = 0$ então $\lambda(G) = 0$ pois G é desconexo. Assuma que $\delta(G) \geq |V(G)|/2 \geq 1$.

Se um corte de aresta separa um vértice do resto do grafo, esse corte tem cardinalidade $\geq \delta(G) \geq |V(G)|/2$.

Considere cortes de aresta onde as compontentes têm pelo menos dois vértices. Se um corte $F \subset E(G)$ determina as componentes C_1 e C_2, uma delas tem no máximo $|V(G)|/2$ vértices, digamos que C_1. Como $\delta(G) \geq |V(G)|/2$ em todo vértice de C_1 incide pelo menos uma aresta cujo outro extremo está em C_2.

O número de arestas em F é pelo menos

$$|V(C_1)|\big(\delta(G) - |V(C_1)| + 1\big) = -|V(C_1)|^2 + (\delta(G)+1)|V(C_1)|,$$

que é uma função do 2° grau na variável $|V(C_1)|$ com concavidade para baixo. Portanto, o mínimo ocorre nos extremos do intervalo; como $2 \leq |V(C_1)| \leq |V(G)|/2$:

$|V(C_1)| = 2$: $|F| \geq -2^2 + (\delta(G)+1)2 \geq \delta(G)$, pois $\delta(G) \geq 1$;

$|V(C_1)| = |V(G)|/2$: $\quad |F| \geq -|V(G)|^2 + (\delta(G)+1)|V(G)| \geq \delta(G)$, pois $\delta(G) \geq |V(G)|/2$. $\square$