Aluno(a):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Matrícula:\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| 1a) | 2,0 |  |
| 2a) | 2,0 |  |
| 3a) | 2,0 |  |
| 4a) | 2,0 |  |
| 5a) | 2,0 |  |
|  | 10.0 |  |

**LEIA COM CUIDADO**

1. A prova é individual e sem consulta.
   1. **Qualquer tentativa de “cola” resultará na anulação da prova do aluno ou dos alunos envolvidos**.
   2. Os aparelhos celulares deverão permanecer desligados e guardados fora do alcance durante toda a prova. **Aparelhos celulares ligados ou de alguma forma visíveis serão tratados como tentativa de “cola”**.
2. A interpretação faz parte da questão.
   1. **Não há perguntas durante a prova.**
   2. Em caso de dúvida escreva a dúvida e a sua interpretação na resposta.
3. O tempo de prova é 1:45 h.
4. **Após o início da prova, não será possível sair e voltar à sala.**
5. As respostas devem seguir as questões. Caso precise de rascunho use o verso da folha.
6. A prova pode ser feita a lápis.

**Questão 1 (2,0 pontos)** Considere o grafo com pesos nos arcos mostrado na Figura 1. Apresente as sucessivas matrizes computadas pelo algoritmo de Floyd-Warshall e os respectivos caminhos, como nas Figuras 2a e 2b, para k=0.

Caminhos para *k*=0

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *k*=0 | | *j* | | | |
| 1 | 2 | 3 | 4 |
| *i* | 1 | 0 | ∞ | -2 | ∞ |
| 2 | 4 | 0 | 3 | ∞ |
| 3 | ∞ | ∞ | 0 | 2 |
| 4 | ∞ | -1 | ∞ | 0 |

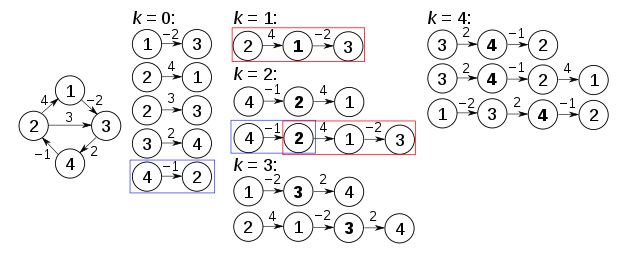
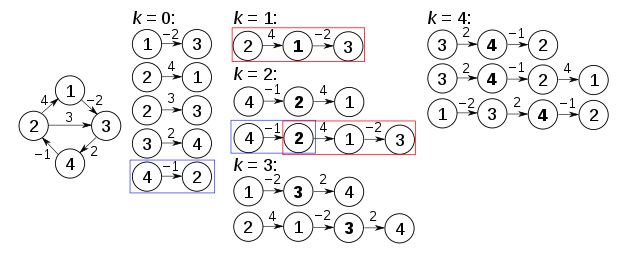
******

Figura 2a

Figura 1

Figura 2b

***Resposta***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *k*=1 | | *j* | | | |
| 1 | 2 | 3 | 4 |
| *i* | 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *k*=2 | | *j* | | | |
| 1 | 2 | 3 | 4 |
| *i* | 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |

Caminhos para *k*=3

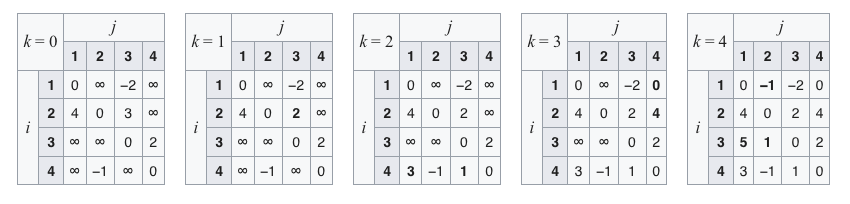
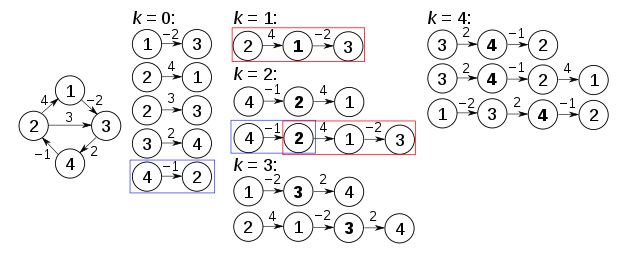
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *k*=4 | | *j* | | | |
| 1 | 2 | 3 | 4 |
| *i* | 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *k*=3 | | *j* | | | |
| 1 | 2 | 3 | 4 |
| *i* | 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |

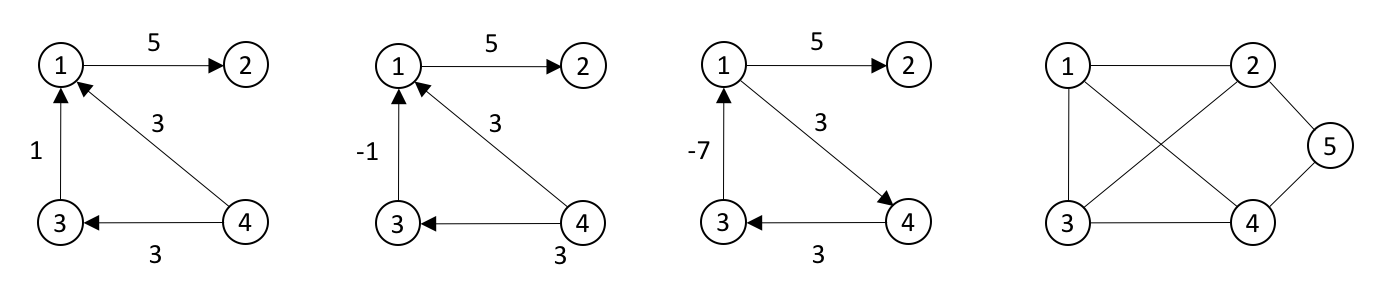
Caminhos para *k*=4

Caminhos para *k*=2

Caminhos para *k*=1

****

**Questão 2 (2,0 pontos)** Paracada um dos grafos abaixo, indique qual a melhor forma de computar o custo do caminho mínimo entre cada par de vértices, para todos os pares de nós. Note que o grafo do item (d) é não dirigido e não tem pesos; neste caso, considere que cada aresta tem peso 1. Explique brevemente a sua resposta.



(a) (0,5 ponto) (b) (0,5 ponto) (c) (0,5 ponto) (d) (0,5 ponto)

***Resposta***

(a) Usar Dijkstra a partir de cada nó. Esta é a melhor opção pois o grafo é esparso: dos 12 arcos possíveis, o grafo só tem 4.

(b) Usar Floyd-Warshall. Não é possível usar Dijkstra pois há um peso negativo.

(c) Não é possível usar Dijkstra pois há um peso negativo, nem Floyd-Warshall pois há um ciclo negativo.

(d) Usar Floyd-Warshall. Esta é a melhor opção pois o grafo é denso: das 10 arestas possíveis, o grafo possui 7.

**Questão 3 (2,0 pontos)** O *giant coeficient* de um grafo não dirigido é o número de nós da maior componente conexa do grafo, dividido pelo total de nós do grafo.

(a) **(1,0 ponto)** Escreva uma rotina eficiente, em pseudo-código, para computar o *giant coeficient* de um grafo não dirigido, representado pela suas listas de adjacência. A rotina pode usar um dos algoritmos de grafos apresentados em sala. Não é necessário escrever o código em C.

(b) **(1,0 ponto)** Mostre como a sua rotina computa o *giant coeficient* do grafo mostrado na Figura 3.

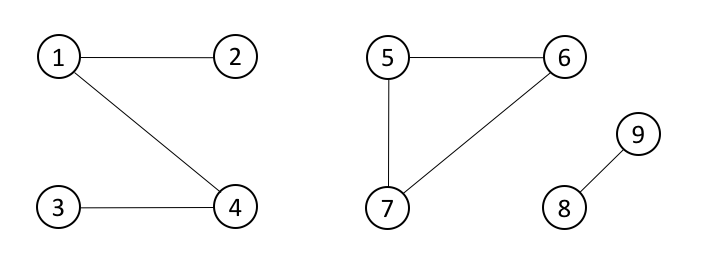


Figura 3

***Resposta***

(a) (1,0 ponto)

1. Inicialize um vetor, *vis*, com 0 para indicar que nenhum nó foi visitado.
2. Para *i*=1 até o número de nós do grafo, enquanto houver um nó não visitado, faça
   1. Procure em *vis* um nó *n* não visitado.
   2. Faça uma busca em profundidade a partir de *n*, marcando todos os nós visitados, inclusive *n*, com *i*.
3. Conte os nós de cada componente conexa e determine a componente com maior número de nós.
4. Compute e retorne o *giant coeficient*.

(b) (1,0 ponto)

1. A busca em profundidade a partir do nó 1 marca os nós 1, 2, 3, 4 como pertencentes à componente conexa 1.
2. A busca em profundidade a partir do nó 5 marca os nós 5, 6, 7 como pertencentes à componente conexa 2.
3. A busca em profundidade a partir do nó 8 marca os nós 8, 9 como pertencentes à componente conexa 3.
4. A componente conexa 1 possui 4 nós; a componente 2 possui 3 nós, e a componente 3 possui 2 nós.
5. O *giant coeficiente* será então: *g* = 4/9

**Questão 4 (2,0 pontos)** O *grau de separação* entre um par de nós (*n*,*m*) de um grafo não dirigido é o comprimento do caminho mínimo entre (*n*,*m*). O *grau de separação médio* de um grafo não dirigido é a média dos graus de separação entre os pares de nós do grafo. A *densidade de arestas* de um grafo não dirigido é igual ao número de arestas do grafo, dividido pelo número total de arestas possíveis entre os nós do grafo.

(a) **(1,0 ponto)** Escreva uma rotina eficiente, em pseudo-código, para computar o grau de separação médiode um grafo não dirigido, representado pela sua matriz de adjacências. A rotina poderá usar um ou mais algoritmos de grafos apresentados em sala e deverá levar em consideração a densidade de arestas do grafo. Não é necessário escrever o código em C.

(b) **(1,0 ponto)** Qual seria o custo (em tempo), no pior caso, da sua rotina? Explique a sua resposta.

***Resposta***

(a)

1. Compute a densidade de arestas do grafos.
2. Se a densidade for maior do que 0,5
   1. Então use o algoritmo de Floyd-Warshall para computar os caminhos mínimos entre todos os pares de nós.
   2. Senão use o algoritmo de Dijskatra para cada nó do grafo para computar os caminhos mínimos entre todos os pares de nós;
   3. Assuma que a saída em ambos os casos é uma matriz *M* com as distâncias mínimas.
3. Varra a matriz *M* e compute a média dos elementos de *M*.
4. Retorne o a média obtida – este valor é o grau de separação médio do grafo.

(b) O custo (em tempo), no pior caso, da rotina é O(*n*3), onde *n* é o número de nós do grafo. De fato, o custo da rotina é dominado pelo custo do uso de executar o algoritmo de Floyd-Warshall no passo (1a), ou pelo custo de executar *n* vezes o algoritmo de Dijkstra, no passo (2b).

**Questão 5 (2,0 pontos)** Considere o grafo dirigido da Figura 4.

0

1

2

3

4

2

10

7

5

6

4

3

Figura 4

(a) **(0,5 ponto)** Represente o grafo por listas de adjacências.

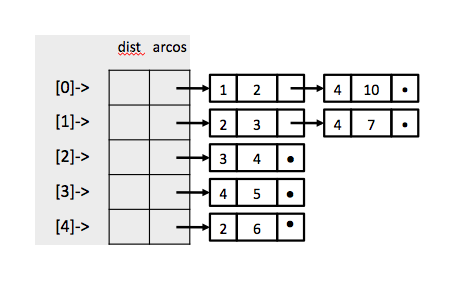
(b) **(0,5 ponto)** Liste os nós na ordem que serão visitados se for realizada uma busca em amplitude (breadth-first search) começando pelo nó 0. Assuma que os vizinhos de um nó são escolhidos em ordem crescente. Mostre todos os passos durante o processamento do algoritmo.

(c) **(0,5 ponto)** Liste os nós na ordem que serão visitados se for realizada uma busca em profundidade (depth-first search) começando pelo nó 0. Assuma que os vizinhos de um nó são escolhidos em ordem crescente. Mostre todos os passos durante o processamento do algoritmo.

(d) **(0,5 ponto)** Discuta se há alguma forma de otimizar a busca em amplitude.

***Resposta***

(a)



1

(b) Nó visitado / Fila

0 1 4

1 4 2 - 4 não precisa ser incluído novamente

4 2 - 2 não precisa ser incluído novamente

2 3

3 - - 4 já foi visitado

Ordem: 0 1 4 2 3

(c) Nó visitado / Pilha

0 1 4

1 2 4 - 4 não precisa ser incluído novamente

2 3 4

3 4 - 4 não precisa ser incluído novamente

4 - - 2 já foi visitado

Ordem: 0 1 2 3 4

(d) Na resposta do item (b), um nó é incluído na fila apenas uma vez. Isto exige percorrer a fila toda vez que um nó é incluído. Outra opção seria admitir duplicatas na fila e não processar um nó que já tivesse sido visitado, o que seria mais eficiente (e já é feito de qualquer forma).