EAD

EDUCAÇÃO A DISTÂNCIA

UNIDADE II

Teoria dos Grafos

Profa. Dra. Miryam de Moraes

Teoria dos Grafos – Unidade II

- Objetivo:
- Apresentar algoritmos para Grafos.
  - Algoritmo de Prim.
  - Algoritmo de Kruskal.
  - Caminho mínimo de Bellman-Ford.
  - Algoritmo de Dijkstra.
  - Caminho mínimo entre todos os pares.
- Tecer considerações sobre: Problema das quatro cores;

  O problema do fluxo máximo;

  Teorema de Ford Fulkerson.

Caminho Mínimo

- Seja um grafo simples e conexo e **com peso**, em que os pesos são positivos.
- O peso representa, muitas vezes, a distância entre dois nós.
- Se o grafo é conexo, então existe ao menos um caminho entre dois nós quaisquer.
- Podem existir muitos caminhos.
- A pergunta é: se existirem muitos caminhos, como encontrar um caminho com peso mínimo?

O Problema da Árvore Geradora Mínima – Algoritmo de Prim

- Uma árvore geradora para um grafo conexo é uma árvore sem raiz cujo conjunto de nós coincide com o conjunto de nós do grafo e cujos arcos são (alguns dos) arcos do grafo.
- Um dos algoritmos para a construção da árvore geradora foi apresentado por Prim em 1957.

Para ilustrar a computação do algoritmo fez uso de um grafo cuja matriz de adjacência modificada é como se segue:

$$\begin{bmatrix} - & 6.7 & 5.2 & 2.6 & 5.6 & 3.6 \\ 6.7 & - & 5.7 & 7.3 & 5.1 & 3.2 \\ 5.2 & 5.7 & - & 3.4 & 8.5 & 4.0 \\ 2.8 & 7.3 & 3.4 & - & 8.0 & 4.4 \\ 5.6 & 5.1 & 8.5 & 8.0 & - & 4.6 \\ 3.6 & 3.2 & 4.0 & 4.4 & 4.6 & - \end{bmatrix}$$

- Para ilustrar o algoritmo, empregar-se-á a mesma matriz de adjacência.
- Iniciar-se-á arbitrariamente no nó 1.

O Problema da Árvore Geradora Mínima – Algoritmo de Prim

- IN = {1} (inicialização arbitrária no nó 1)
- Entre os nós adjacentes ao nó 1, aquele cuja aresta para alcançá-lo apresenta menor distância é o nó 4, que é selecionado. IN = {1,4}
- Entre os nós adjacentes ao nó 4, aquele cuja aresta para alcançá-lo apresenta menor distância é o nó 3, que é selecionado. IN = {1,4,3}
- Entre os nós adjacentes do nó 3, aquele cuja aresta para alcançá-lo apresenta menor custo é o nó 4. Uma vez que este foi inserido no conjunto IN, escolhe-se o nó 6. IN = {1,4, 3, 6}
- Entre os nós adjacentes ao nó 6, aquele cuja aresta para alcançá-lo apresenta menor custo é o nó 2, que é selecionado. IN = {1, 4, 3, 6, 2}

- Resta o nó 5.
- IN = { 1, 4, 3, 6, 2, 5}

Algoritmo de Prim – Pseudocódigo

```
Prim(G, s):
para cada vértice v em G:
        chave[v] = infinito
        pai[v] = nulo
chave[s] = 0
Q = conjunto de todos os vértices em G
enquanto Q não estiver vazio:
        u = vértice em Q com a menor chave
        remover u de Q
para cada vértice v adjacente a u:
        se v ainda estiver em Q e peso(u,v) < chave[v]:
                pai[v] = u
                chave[v] = peso(u,v)
retornar a AGM
```

Observação:

// Ao final do algoritmo:

retornar a AGM representada pelo conjunto de pares
(pai[v],v) para todos os vértices v em G, exceto para o vértice raiz s

Algoritmo de Kruskal

- O algoritmo de Kruskal destina-se a encontrar uma **árvore geradora mínima** em um grafo conexo.

- O algoritmo de Kruskal inclui **arcos em ordem crescente de distância**, onde quer que estejam no grafo.

- A única restrição é que um arco não é incluído se sua inclusão criar um ciclo.

- O algoritmo termina quando todos os nós estiverem incorporados em uma estrutura conexa.

Algoritmo AGMKruskal (GERSTING, 2014)

AGMKruska l (matriz n x n; coleção de arcos T)

// Algoritmo de Kruskal para encontrar uma árvore geradora mínima;

// inicialmente, T é vazio: ao final, T = árvore geradora mínima.

ordene os arcos em G por distância em ordem crescente

repita

se o próximo arco na ordem não completa um ciclo

então inclua esse arco em T

fim do se

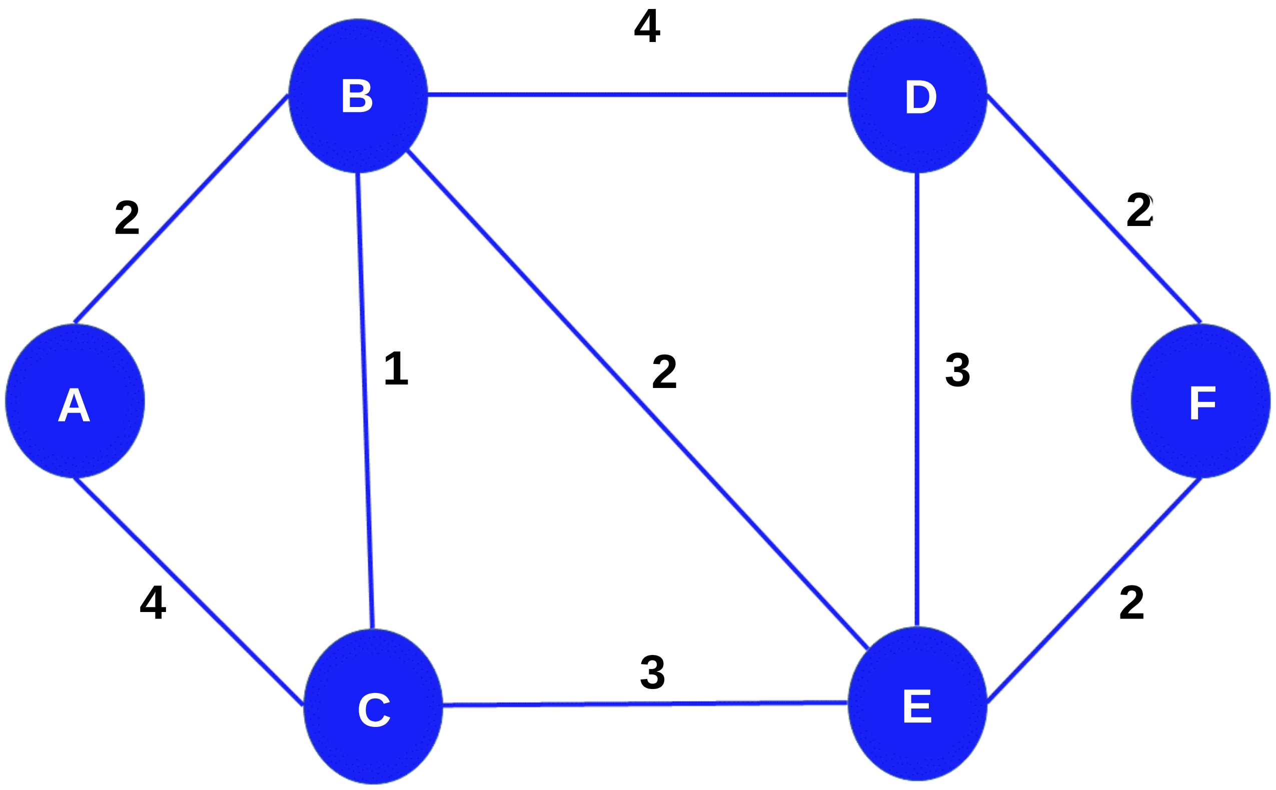
até T ser conexo e conter todos os nós em G.

fim de AGMKruskal

Algoritmo de Kruskal

- Considere o seguinte grafo e a matriz de **adjacência modificada**.

- $$A = \begin{bmatrix} \infty & 2 & 4 & \infty & \infty & \infty \\ 2 & \infty & 1 & 4 & 2 & \infty \\ 4 & 1 & \infty & \infty & 3 & \infty \\ \infty & 4 & \infty & \infty & 3 & 2 \\ \infty & 2 & 3 & 3 & \infty & 2 \\ \infty & \infty & \infty & 2 & 2 & \infty \end{bmatrix}$$



Fonte: autoria própria.

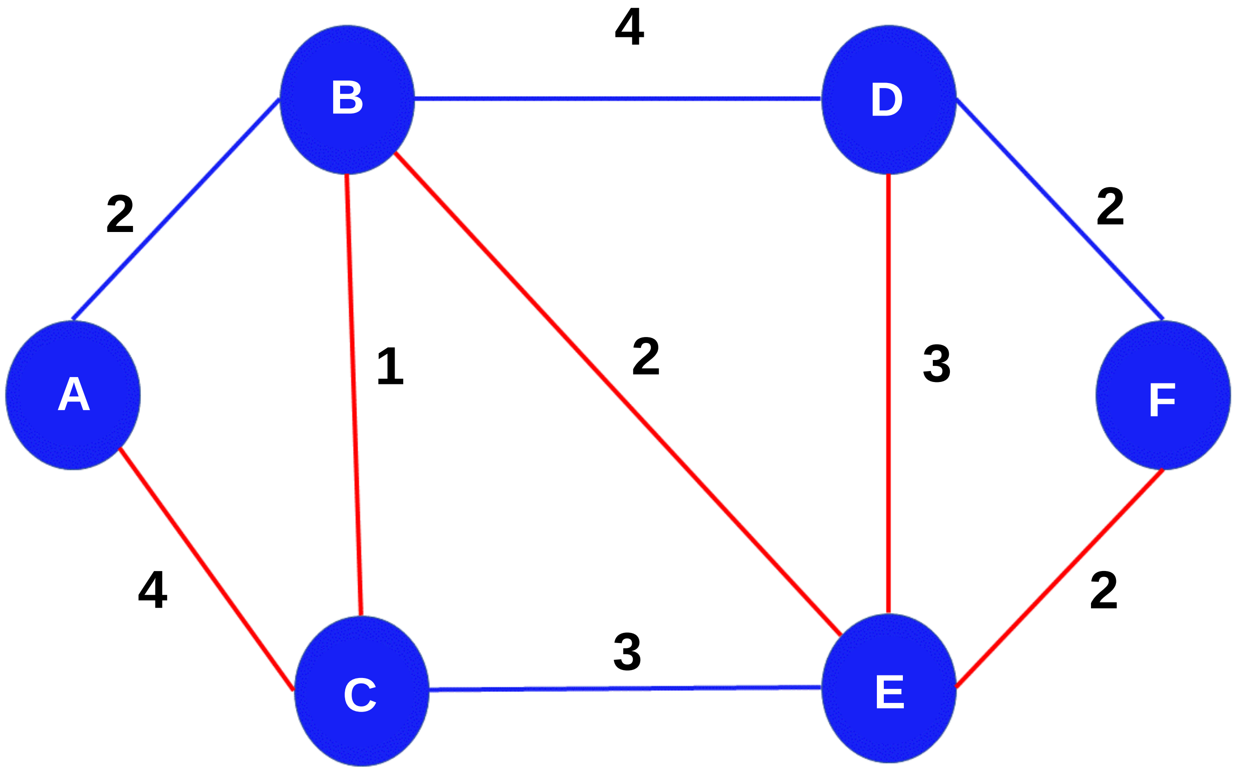
Algoritmo de Kruskal

Ao se ordenar as arestas, segundo os pesos, tem-se:

- D = 1, aresta: B-C; d = 2, arestas: B-A; B-E; E-F; F-D; d=3, arestas E-D e E-C; d=4: C-A

Árvore Geradora Mínima:

- A-C; C-B; B-E; E-D; E-F



Fonte: autoria própria.

Interatividade

Considere as seguintes afirmações:

I. Em um grafo de n vértices e m arestas, a matriz de adjacência modificada apresenta n linhas e m colunas.

II. Considere um circuito digital com n portas lógicas de 2 entradas. Imagine que se deseja testar se há interligações com problemas tais como curto-circuito ou interconexões com trilhas em aberto. Se se considerar cada porta como nós de grafos e as interligações entre as portas como arestas, é possível afirmar, em tempo polinomial, se existe um caminho que passa por todos os nós uma única vez.

III. O algoritmo de Prim permite encontrar a árvore geradora mínima em um grafo simples e conexo.

Interatividade

São corretas as afirmações:

a) Apenas I e III.
b) Apenas II e III.
c) I, II e III.
d) Apenas I e II.
e) Apenas I.

Resposta

Considere as seguintes afirmações:

I. Em um grafo de n vértices e m arestas, a matriz de adjacência modificada apresenta n linhas e m colunas.

II. Considere um circuito digital com n portas lógicas de 2 entradas. Imagine que se deseja testar se há interligações com problemas tais como curto-circuito ou interconexões com trilhas em aberto. Se se considerar cada porta como nós de grafos e as interligações entre as portas como arestas, é possível afirmar, em tempo polinomial, se existe um caminho que passa por todos os nós uma única vez.

III. O algoritmo de Prim permite encontrar a árvore geradora mínima em um grafo simples e conexo.

Resposta

São corretas as afirmações:

a) Apenas I e III.
b) Apenas II e III.
c) I, II e III.
d) Apenas I e II.
e) Apenas I.

Algoritmo de Bellman – Ford (GERSTING, 2014)

- Executa uma série de cálculos, procurando encontrar **caminhos mínimos**, sucessivamente, de comprimento 1, depois de comprimento 2, e assim por diante, até o comprimento máximo n-1.
- Algoritmo CaminhoMinimoBF.
- CaminhoMinimoBF (matriz n x n A; no x; vetor de inteiros d, vetor de nós s[y].
- //Algoritmo de Bellman-Ford. A é uma matriz de adjacência
- //modificada de um grafo simples e conexo com pesos positivos
- // x é um nó no grafo, quando o algoritmo terminar, os nós
- // do caminho mínimo de x para um nó y são y, s[y], s[s[y]],...,x;
  - //a distância correspondente é d[y]
  - Variáveis locais:
  - Nós z, p   //nós temporários
  - Vetor de inteiros t   //vetor de distâncias temporário

Algoritmo de Bellman – Ford (GERSTING, 2014)

//inicializa os vetores d e s; estabelece os caminhos mínimos de comprimento 1 a partir de x

d[x]=0

Para todos os nós z diferentes de x **faça**

d[z]=A[x, z]

s[z] = x

Fim do para

//encontrar os caminhos mínimos de comprimentos 2, 3 etc.

para i=2 até n-1 **faça**

t=d

// modifica t para guardar os menores caminhos de

//comprimento i

Algoritmo de Bellman – Ford (GERSTING, 2014)

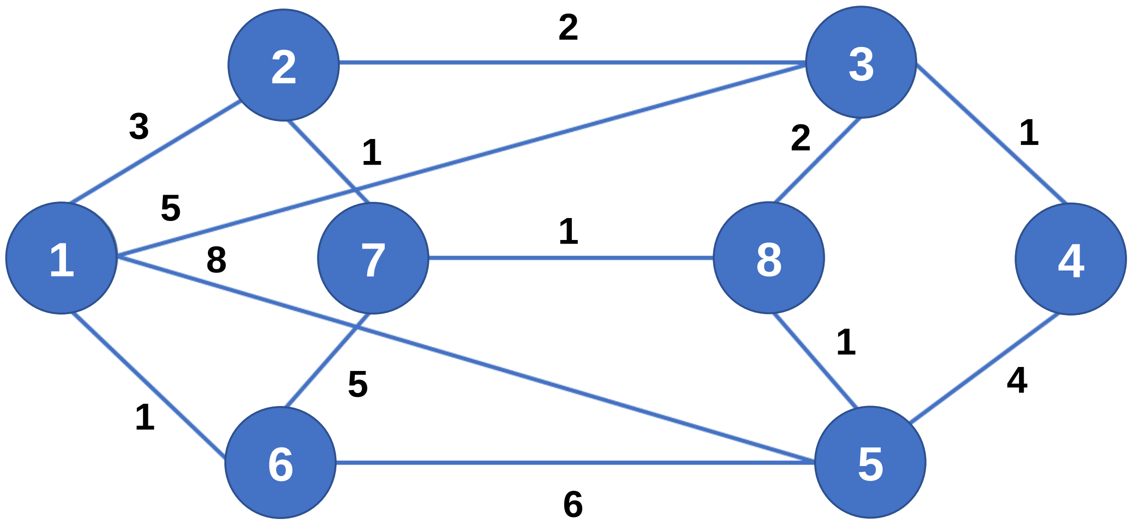
```
        para todos os nós z diferentes de x faça
            //encontra o caminho mínimo com mais um arco
            p = nó em G para o qual (d[p] + A[p,z]) é mínimo
            t[z] = d[p] + A[p,z]
            se p≠ z então
                s[z] = p
            fim do se
        fim do para
        d = t;
    fim do para

fim do CaminhoMinimoBF
```

Exemplo: Algoritmo de Bellman – Ford

- O seguinte exemplo foi fundamentado em Gersting, J. L. exercício 11, capítulo 7.3, 7. ed.
- Deseja-se usar o algoritmo de Bellman-Ford para encontrar o caminho da origem nó 2 para qualquer outro nó.

Após 2 iterações, tem-se:

	1	2	3	4	5	6	7	8
d	3	0	2	3	3	4	1	2
s	2	-	2	3	8	1	2	7



Fonte: Gersting (2014).

Caminho Mínimo entre todos os pares (GERSTING, 2014)

CaminhoMinimoEntreTodosOsPares(matriz n x n A) **//----------------O(n^3)---------------------**

//Algoritmo de Floyd – calcula o caminho mínimo entre todos pares; inicialmente A é a matriz

//de adjacência; ao final, A vai conter todas as distâncias dos caminhos mínimos

Para k=1 até n **faça**

Para i =1 até n **faça**

Para j = 1 até n **faça**

se A[i,k] + A[k,j] < A[i,j] **então**

A[i,j]= A[i,k] + A[k,j]

fim do se

fim do para

fim do para

fim do para

fim de CaminhoMinimoEntreTodosOsPares

Interatividade

Fundamentada na questão 35, Poscomp, 2012. Concernente aos algoritmos em grafos, relacione a coluna 1 com a coluna 2.

Coluna 1:

I. Árvore Geradora Mínima (Prim)
II. Caminho Mais Curto (Dijkstra)
III. Árvore Geradora Mínima (Kruskal)

Interatividade

Coluna 2

- (A) Toma como entrada um grafo não orientado com pesos nas arestas, ordena as arestas por peso e escolhe as arestas de forma a não fechar ciclos para resolver o problema.
- (B) Toma como entrada um grafo não orientado com pesos nas arestas, utiliza basicamente busca em largura escolhendo arestas de menor peso para resolver o problema.
- (C) Toma como entrada um grafo não orientado com pesos nas arestas, utiliza basicamente busca em largura escolhendo distâncias acumuladas de menor peso para resolver o problema. Assinale a alternativa correta:

a) I-B; II-C; III-A.
b) I-A;II-B;III-C.
c) I-A; II-C; II-B.
d) I-B; II-A;III-C.
e) I-C; II-B; III-A.

Resposta

Coluna 2

- (A) Toma como entrada um grafo não orientado com pesos nas arestas, ordena as arestas por peso e escolhe as arestas de forma a não fechar ciclos para resolver o problema.
- (B) Toma como entrada um grafo não orientado com pesos nas arestas, utiliza basicamente busca em largura escolhendo arestas de menor peso para resolver o problema.
- (C) Toma como entrada um grafo não orientado com pesos nas arestas, utiliza basicamente busca em largura escolhendo distâncias acumuladas de menor peso para resolver o problema. Assinale a alternativa correta:

a) I-B; II-C; III-A.

b) I-A;II-B;III-C.

c) I-A; II-C; II-B.

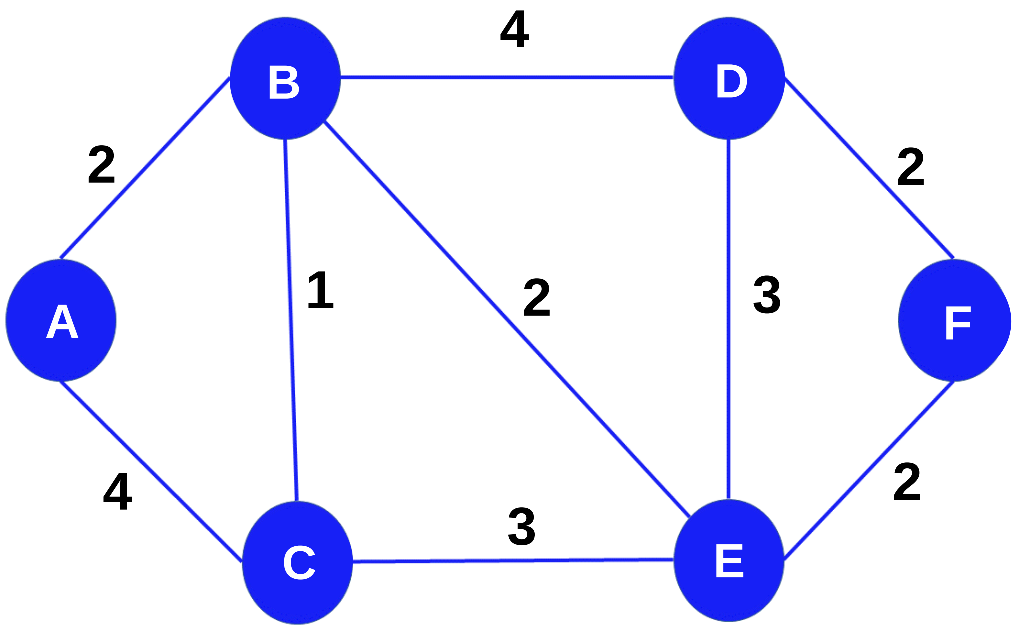
d) I-B; II-A;III-C.

e) I-C; II-B; III-A.

Algoritmo de Dijkstra

- Considere o seguinte grafo e a matriz de **adjacência modificada**. Deseja-se o caminho mínimo entre os nós A e F.

- $A = \begin{bmatrix} \infty & 2 & 4 & \infty & \infty & \infty \\ 2 & \infty & 1 & 4 & 2 & \infty \\ 4 & 1 & \infty & \infty & 3 & \infty \\ \infty & 4 & \infty & \infty & 3 & 2 \\ \infty & 2 & 3 & 3 & \infty & 2 \\ \infty & \infty & \infty & 2 & 2 & \infty \end{bmatrix}$



Fonte: autoria própria.

Algoritmo de Dijkstra

- IN = {A}

	A	B	C	D	E	F
d	0	2	4	∞	∞	∞
s	A	A	A	A	A	A

	A	B	C	D	E	F
d	0	2	4	∞	∞	∞
s	A	A	A	A	A	A

Algoritmo de Dijkstra

- O menor peso é o da aresta de A para B, com valor 2. B deverá ser inserido no conjunto IN, e marcar o peso como d = 2.

- Para se selecionar o próximo nó, deve-se identificar, para cada nó x do grafo, o peso da aresta entre B e x, ou seja d(B,x).

- Seguidamente, deve-se **somar d(B,x) a d e comparar o resultado com d(A,x).**
- Entre os dois valores seleciona-se o valor mínimo e insere-se no conjunto IN, o nó selecionado.

Repete-se sucessivamente, até que todo o nó destino seja inserido em IN. Tem-se:

Algoritmo de Dijkstra: novamente…

- IN = {A}

	A	B	C	D	E	F
d	0	2	4	∞	∞	∞
s	A	A	A	A	A	A

	A	B	C	D	E	F
d	0	2	4	∞	∞	∞
s	A	A	A	A	A	A

Algoritmo de Dijkstra

- IN = {A, B}
- d= 2
- d[C]= min(4, 2 +A[B, C]) =min(4,3)=3
- d[D]= min(∞, 2 +A[B, D]) =min(∞,2+4)=6
- d[E]= min(∞, 2 +A[B, E]) =min(∞,2+2)=4
- d[F]= min(∞, 2 +A[B, F]) =min(∞,2+ ∞)= ∞

	A	B	C	D	E	F
d	0	2	B	6	4	∞
s	A	A	3	B	B	A

	A	B	C	D	E	F
d	0	2	3	6	4	∞
s	A	A	B	B	B	A

Algoritmo de Dijkstra

- IN = {A, B, C}
- d= 3
- d[D]= min(6, 3 +A[C, D]) =min(6,3+4)=6
- d[E]= min(4, 3 +A[C, E]) =min(4,3+3)=4
- d[F]= min(∞, 3 +A[C, F]) =min(∞,3+ ∞)= ∞

	A	B	C	D	E	F
d	0	2	3	6	4	∞
s	A	A	B	B	B	A

	A	B	C	D	E	F
d	0	2	3	6	4	∞
s	A	A	B	B	B	A

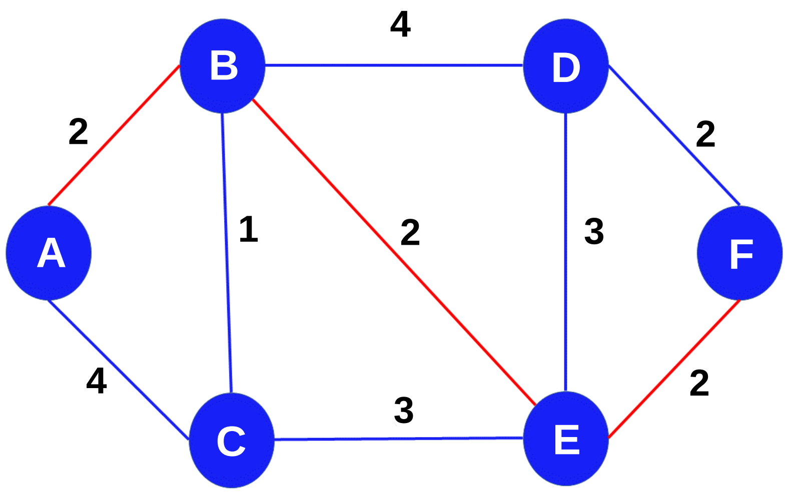
Algoritmo de Dijkstra

- IN = {A, B, C,E}
- D = 4
- d[D] = min(6, 4 +A[E, D]) = min(6,4+3)=6
- d[F] = min(∞, 4 +A[E, F]) = min(∞,4+ 2)= 6

Solução: Caminho Mínimo = A-B-E-F

	A	B	C	D	E	F
d	0	2	3	6	4	∞
s	A	A	B	B	B	A

	A	B	C	D	E	F
d	0	2	3	6	4	6
s	A	A	B	B	B	E



Fonte: autoria própria.

Algoritmo de Dijkstra (GERSTING, 2014)

- Caminho Mínimo(matriz nxn; nós x, y)
- //Algoritmo de Dijkstra. A é uma matriz de adjacência modificada
- //de um grafo simples e conexo com pesos positivos; x e y são nós
- //no grafo; o algoritmo escreve os nós do caminho mínimo de x
- // para y e a distância correspondente.

Variáveis locais:

- Conjunto de nós IN //nós cujo caminho mínimo de x é conhecido
  - Nós z, p //nós temporários
  - Vetor de inteiros d //para cada nó, distância de x usando //nós em IN
  - Vetor de nós s // para cada nó, nó anterior no caminho //mínimo
  - Inteiro DistânciaAnterior //distância para comparar

Algoritmo Caminho Mínimo

- //inicializa o conjunto IN e os vetores d e s

  IN = {x}

  d[x] = 0

- **<u>Para</u>** todos os nós z não pertencentes a IN **<u>faça</u>**

  d[z] = A[x, z]

  s[z]= x

  <u>Fim do para</u>

// coloca nós em IN

<u>enquanto</u> y não pertence a IN **<u>faça</u>**

//adiciona o nó de distância mínima não
// pertencente a IN

p = nó z não pertencente a IN com d[z] mínimo

IN = IN ∪ {p}

Algoritmo Caminho Mínimo (continuação)

```
            d[z] = min(d[z], d[p] + A[p, z]
            Se d[z] ≠ DistânciaAnterior então
                  S[z] =p
            Fim do se
      Fim do para
Fim do enquanto

//escreve  os nós do caminho
escreva (“Em ordem inversa, os nós do caminho são”)
escreva(y)
repita
      escreva(s[z])
      z=s[z]
até z = x
```

Algoritmo Caminho Mínimo (continuação)

// escreve a distância correspondente

Escreva (“A distância percorrida é: “, d[y])

Fim de CaminhoMinimo

Interatividade

Considere as seguintes afirmações a respeito do algoritmo de Dijkstra:

I. O algoritmo de Dijkstra é um algoritmo de caminho mínimo de fonte única que resolve o problema de encontrar o caminho mais curto em um grafo ponderado com arestas não negativas.
II. O objetivo do algoritmo é determinar o caminho mais curto a partir de um nó de origem para todos os outros nós do grafo.
III. O algoritmo de Dijkstra começa selecionando um nó inicial e definindo sua distância como zero. São corretas as afirmações:

a) Apenas I e II.
b) Apenas I e III.
c) Apenas II e III.
d) I, II e III.
e) Apenas I.

Resposta

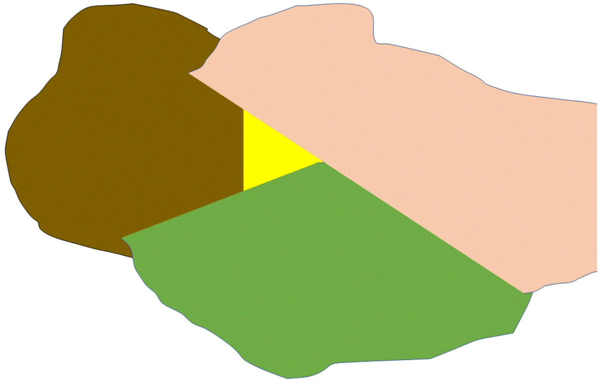
Considere as seguintes afirmações a respeito do algoritmo de Dijkstra:

I. O algoritmo de Dijkstra é um algoritmo de caminho mínimo de fonte única que resolve o problema de encontrar o caminho mais curto em um grafo ponderado com arestas não negativas.
II. O objetivo do algoritmo é determinar o caminho mais curto a partir de um nó de origem para todos os outros nós do grafo.
III. O algoritmo de Dijkstra começa selecionando um nó inicial e definindo sua distância como zero. São corretas as afirmações:

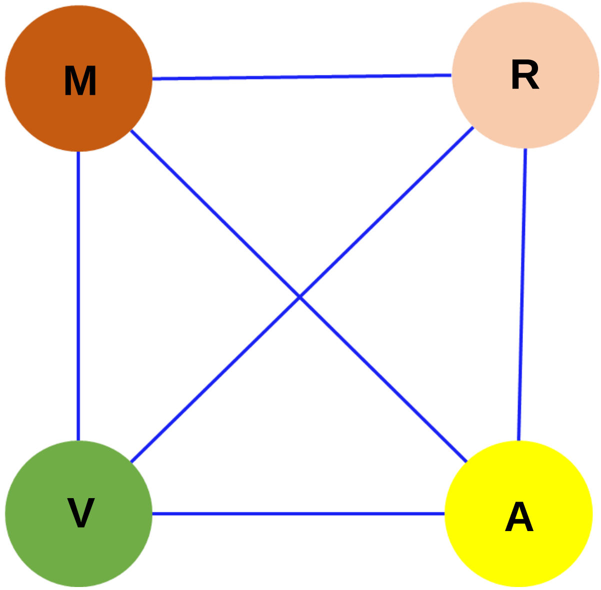
a) Apenas I e II.
b) Apenas I e III.
c) Apenas II e III.
d) I, II e III.
e) Apenas I.

Problema das Quatro Cores

- Associado a qualquer mapa existe um grafo, chamado de grafo dual do mapa, em que cada região do mapa corresponde a um vértice no grafo e um arco entre dois nós devem estar associados a cada duas regiões adjacentes.
- O mapa M seguinte tem como dual o grafo completo K4.



Mapa M



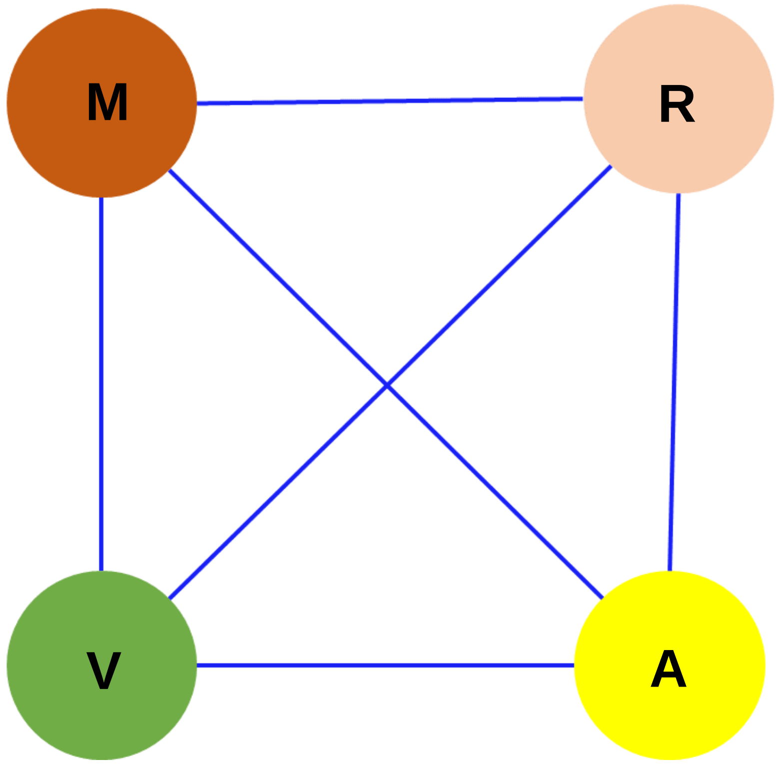
K4

Fonte: autoria própria.

Problema das Quatro Cores

Problema das Quatro Cores: “Todo mapa (plano ou esférico) pode ser pintado com quatro cores de modo que duas regiões, tendo uma fronteira comum, não fiquem com a mesma cor?”

- O número cromático de um grafo é o menor número de cores necessário para se colorir um grafo.
- O número cromático do grafo K4 é 4.



Fonte: autoria própria.

Fluxos em Grafos

- Fluxos em grafos são um conceito importante em teoria dos grafos que representa a quantidade de “fluxo” que pode ser transportada em um grafo com arestas ponderadas. Essa teoria tem diversas aplicações em áreas como engenharia de redes, transporte, programação linear, entre outras.

- Formalmente, um fluxo em um grafo direcionado ponderado é uma atribuição de uma quantidade não negativa de **fluxo a cada aresta**, que respeita a capacidade máxima de cada aresta e conservação do fluxo em cada nó.

  - A capacidade máxima de uma aresta representa a quantidade máxima de fluxo que pode ser transportada através dela.

Fluxos em Grafos

- A conservação do fluxo em cada nó exige que a quantidade de fluxo que entra em um nó deve ser igual à quantidade de fluxo que sai dele, exceto em nós especiais, conhecidos como fontes e sumidouros, em que o fluxo pode entrar ou sair livremente.

- Em resumo, o conceito de fluxos em grafos é fundamental na teoria dos grafos, permitindo a modelagem e solução de problemas relacionados ao transporte de fluxo em redes complexas.

Teorema de Ford Fulkerson

- Um **corte** na rede de fluxo é um conjunto de arestas que, se removido, desconecta a fonte do sumidouro. O valor de um corte é definido como a soma das capacidades das arestas que compõem o corte. O **corte mínimo** é o corte de valor mínimo na rede de fluxo.
- Um **caminho de aumento** é um caminho da fonte ao sorvedouro em que cada aresta tem uma capacidade residual maior que zero.
- O problema do fluxo máximo pode ser resolvido usando o algoritmo de Ford-Fulkerson, que utiliza o conceito de caminhos aumentantes para aumentar iterativamente o fluxo até que não seja mais possível encontrar caminhos adicionais. Ainda, utiliza o conceito de corte mínimo.

Teorema de Ford Fulkerson

O pseudocódigo básico do algoritmo é o seguinte:

1. Inicialize o fluxo F em todas as arestas para 0
2. Enquanto existir um caminho de aumento P na rede residual:

A. Encontre a capacidade mínima residual ao longo de P (chamada de f)

B. Atualize o fluxo F e a capacidade residual das arestas ao longo de P

3. Retorne o fluxo F

- O algoritmo encontra o caminho de aumento com a menor capacidade residual ao longo dele e, em seguida, atualiza o fluxo e as capacidades residuais das arestas.

- O algoritmo de Ford-Fulkerson usa um método guloso para encontrar caminhos de aumento na rede residual.

Interatividade

Considere as seguintes afirmações:

I. Os algoritmos de caminhos mínimos têm diversas aplicações em teoria dos grafos e em outras áreas, como em redes de comunicação, roteamento, planejamento de trajetórias em robótica, jogos de estratégia, entre outros.

II. O algoritmo de Dijkstra funciona construindo um conjunto de vértices não visitados e um conjunto de vértices visitados, e calculando as distâncias mínimas a partir do vértice de origem para todos os vértices não visitados. O algoritmo utiliza uma fila de prioridade para selecionar o vértice com a menor distância entre os vértices não visitados a cada iteração.

III. O algoritmo de Bellman-Ford é outro algoritmo que pode ser usado para encontrar o caminho mínimo de fonte única em grafos ponderados, inclusive em grafos com pesos negativos.

Interatividade

São corretas as afirmações:

a) Apenas I e III.
b) Apenas II e III.
c) I, II e III.
d) Apenas I e II.
e) Apenas I.

Resposta

Considere as seguintes afirmações:

I. Os algoritmos de caminhos mínimos têm diversas aplicações em teoria dos grafos e em outras áreas, como em redes de comunicação, roteamento, planejamento de trajetórias em robótica, jogos de estratégia, entre outros.

II. O algoritmo de Dijkstra funciona construindo um conjunto de vértices não visitados e um conjunto de vértices visitados, e calculando as distâncias mínimas a partir do vértice de origem para todos os vértices não visitados. O algoritmo utiliza uma fila de prioridade para selecionar o vértice com a menor distância entre os vértices não visitados a cada iteração.

III. O algoritmo de Bellman-Ford é outro algoritmo que pode ser usado para encontrar o caminho mínimo de fonte única em grafos ponderados, inclusive em grafos com pesos negativos.

Resposta

São corretas as afirmações:

a) Apenas I e III.
b) Apenas II e III.
c) I, II e III.
d) Apenas I e II.
e) Apenas I.

Referências

- BOAVENTURA NETTO, P. O.; JURKIEWICZ S. *Grafos:* introdução e prática. São Paulo: Blucher, 2009.
- CALAÇA, O. Uma introdução às redes neurais para grafos (GNN). *Midium*, 2020. Disponível em: https://medium.com/@otaviocx/uma-introdu%C3%A7%C3%A3o-%C3%A0s-redes-neurais-para-grafos-gnn-60e53fcd77d6. Acesso em: 6 mar. 2023.
- GERSTING, J. L. *Fundamentos matemáticos para a ciência da computação*. Matemática Discreta e suas Aplicações. Rio de Janeiro: LTC, 2014.
- GOLDBARG, M.; GOLDBARG, E. *Grafos conceitos, algoritmos e aplicações*. Rio de Janeiro: LTC, 2018.
- GOTTLICH, S.; TOTZECH, C. Calibração de parâmetros com descida de gradiente estocástico para sistemas de partículas interativos conduzidos por redes neurais. *Springer Link*, 2021. Disponível em: https://doi.org/10.1007/s00498-021-00309-8. Acesso em: 10 maio 2023.

Referências

- HAYKIN, S. *Redes neurais*: princípios e prática. Porto Alegre: Bookman, 2007.
- LESKOVEC, J. *CS224W*: *Machine Learning with Graphs Stanford University*. Disponível em: http://cs224w.stanford.edu/. Acesso em: 19 fev. 2023.
- LIPSCHUTZ, S.; LIPSON, M. *Matemática discreta*: coleção Schaum. Porto Alegre: Bookman, 2013.
- MAX PIXEL. Disponível em: https: www.maxpixel.net. Acesso em: 12 jan. 2022.
- NICOLETTI, M.; HRUSHKA, E. *Fundamentos da teoria dos grafos para computação*. Rio de Janeiro: LTC, 2018.
- NORVIG, P. *Inteligência artificial*. São Paulo: Grupo GEN, 2013.
- PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de software*: uma abordagem profissional. Porto Alegre: AMGH, 2016.

Referências

- SIMÕES-PEREIRA, J. *Grafos e redes teoria e algoritmos básicos*. Rio de Janeiro: Interciência, 2014.
- SWARCFITER, J. L. ; PINTO, P. E. D*. Teoria computacional de grafos*: os algoritmos com programas Python. Rio de Janeiro: Elsevier, 2018.
- SZWARCFITER, J. L.; Markenzon, L. *Estruturas de dados e seus algoritmos*. São Paulo: Grupo GEN, 2010.

ATÉ A PRÓXIMA!