GRAFOS – 25/2

Ciência da Computação
Universidade do Vale do Itajaí – UNIVALI

Profª Fernanda dos Santos Cunha
fernanda.cunha@univali.br

1

Grafos: Unidade 7 – Busca em Grafos

Leitura para aprofundamento

- Algoritmos, Dasgupta, S.; Papadimitriou, C.; Vazirani, U., disponível na Biblioteca A (Biblioteca da intranet)
  - Cap 6 Programação Dinâmica – Seção 6.6 Caminhos mínimos (Floyd)
  - Cap 4 Caminhos em Grafos – Seção 4.4 O algoritmo de Dijkstra

- Vídeo-aula 13 do Curso Projeto e Análise de Algoritmos, disponível em https://integra.univesp.br/courses/2629/pages/semana-7?module_item_id=199712

2

Grafos: Unidade 7 – Busca em Grafos

Determinação do caminho mais curto

- Caminho: sequência de nós e de arcos adjacentes.
- Problema muito frequente em processos de otimização associados a redes de transporte.
- Existem vários métodos para resolução destes problemas.

3

Método de Floyd-Warshall

- Resolve o problema de calcular o caminho mais curto entre todos os pares de vértices em um grafo orientado *(ou não)* e valorado.
- Publicado por Robert Floyd em 1962. Este algoritmo é o mesmo que foi publicado por Bernard Roy em 1959 e também por Stephen Warshall em 1962 para determinar o fechamento transitivo de um grafo*.

  (*) O conceito de fecho transitivo pode ser pensado como a construção de uma estrutura de dados que possibilita responder problemas de atingibilidade.

4

Método de Floyd-Warshall

1º passo:

Numerar todos os nós do grafo $G(N, A)$ com números inteiros em seqüência: $1, 2, 3, \ldots, n$.

2º passo:

Determinar as seguintes matrizes auxiliares:

✓ $D^{(0)}$ - matriz que representa a extensão da trilha

determinamos $D^{(0)}$ inicialmente, da seguinte maneira:

$$d_0(i,j) = \begin{bmatrix} elemento \\ i,j \ \ da \ matriz \\ D^{(0)} \end{bmatrix} = \begin{cases} l(i,j) & se \ o \ arco \ (i,j) \ existe \\ 0 & se \ i = j \\ \infty & se \ o \ arco \ (i,j) \ não \ existe \end{cases}$$

5

Método de Floyd-Warshall

2º passo:

Determinar as seguintes matrizes auxiliares:

✓ $P^{(0)}$ - matriz que fornece a seqüência de nós predecessores

determinamos $P^{(0)}$ inicialmente, da seguinte maneira:

$$P_0(i,j) = \begin{bmatrix} elemento \\ i,j \ \ da \ matriz \\ P^{(0)} \end{bmatrix} = \begin{cases} i & para \ \ i \neq j \\ 0 & para \ \ i = j \end{cases}$$

6

Método de Floyd-Warshall

3º passo:
Fazer numa primeira rodada $k = 1$.

4º passo:
Atualizar todos os elementos da matriz $D^{(K)}$ através da relação:

$$d_k(i,j) = Min\{d_{k-1}(i,j), d_{k-1}(i,k) + d_{k-1}(k,j)\}$$, variando i e j

5º passo:
Atualizar todos os elementos da matriz de nós predecessores $P^{(K)}$ através da relação:

$$P_k(i,j) = \begin{cases} P_{k-1}(k,j) & se\ d_k(i,j) \neq d_{k-1}(i,j) \\ P_{k-1}(i,j) & em\ caso\ contrário \end{cases}$$

7

Método de Floyd-Warshall

6º passo:
Se $k = n$ o processo termina. Se $k < n$, acrescentar uma unidade a k, reciclando o processo a partir do 4º *passo*.

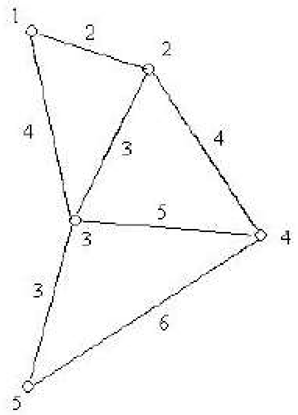
No momento em que o processo terminar, podemos determinar a extensão e a seqüência de arcos que formam a trilha mais curta entre i e j. Para encontrar a extensão mais curta entre i e j basta procurar na matriz $D^{(n)}$ o elemento $d_n(i,j)$. A matriz $P^{(n)}$, por sua vez permite determinar a seqüência de arcos que forma a trilha entre o par de nós (i,j).

8

Método de Floyd-Warshall

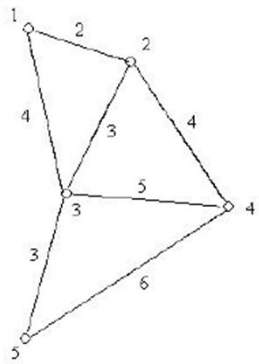
Exemplo:

Determinar as trilhas mais curtas entre os 5 nós da rede abaixo.



9

Método de Floyd-Warshall

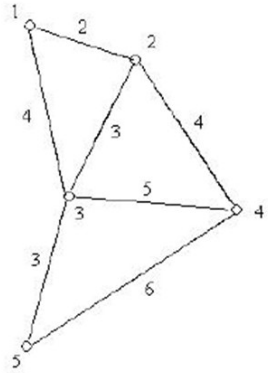


Solução:

$$D^{(0)} = \begin{bmatrix} 0 & 2 & 4 & \infty & \infty \\ 2 & 0 & 3 & 4 & \infty \\ 4 & 3 & 0 & 5 & 3 \\ \infty & 4 & 5 & 0 & 6 \\ \infty & \infty & 3 & 6 & 0 \end{bmatrix} \qquad P^{(0)} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 2 & 0 & 2 & 2 & 2 \\ 3 & 3 & 0 & 3 & 3 \\ 4 & 4 & 4 & 0 & 4 \\ 5 & 5 & 5 & 5 & 0 \end{bmatrix}$$

10

Método de Floyd-Warshall



Faço $k = 1$

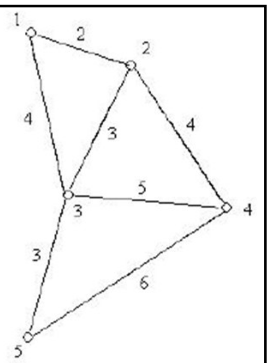
$d_1(i,j) = Min\{d_0(i,j), d_0(i,1) + d_0(1,j)\}$

$$D^{(1)} = \begin{bmatrix} 0 & 2 & 4 & \infty & \infty \\ 2 & 0 & 3 & 4 & \infty \\ 4 & 3 & 0 & 5 & 3 \\ \infty & 4 & 5 & 0 & 6 \\ \infty & \infty & 3 & 6 & 0 \end{bmatrix} \qquad P^{(1)} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 2 & 0 & 2 & 2 & 2 \\ 3 & 3 & 0 & 3 & 3 \\ 4 & 4 & 4 & 0 & 4 \\ 5 & 5 & 5 & 5 & 0 \end{bmatrix}$$

Como não houve alteração na matriz $D^{(1)}$, a matriz de nós predecessores não será alterada.

11

Método de Floyd-Warshall



Faço $k = 2$

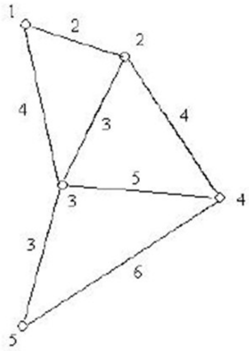
$d_2(i,j) = Min\{d_1(i,j), d_1(i,2) + d_1(2,j)\}$

$$D^{(2)} = \begin{bmatrix} 0 & 2 & 4 & 6 & \infty \\ 2 & 0 & 3 & 4 & \infty \\ 4 & 3 & 0 & 5 & 3 \\ 6 & 4 & 5 & 0 & 6 \\ \infty & \infty & 3 & 6 & 0 \end{bmatrix} \qquad P^{(2)} = \begin{bmatrix} 0 & 1 & 1 & 2 & 1 \\ 2 & 0 & 2 & 2 & 2 \\ 3 & 3 & 0 & 3 & 3 \\ 2 & 4 & 4 & 0 & 4 \\ 5 & 5 & 5 & 5 & 0 \end{bmatrix}$$

Note que houve alteração na nova matriz de distâncias, assim devemos atualizar a matriz de predecessores.

12

Método de Floyd-Warshall



Faço $k = 3$

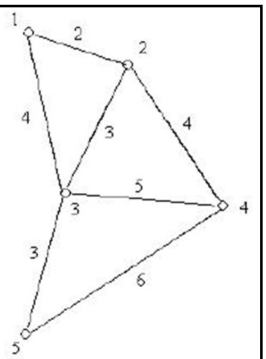
$$d_3(i,j) = Min\{d_2(i,j), d_2(i,3) + d_2(3,j)\}$$

$$D^{(3)} = \begin{bmatrix} 0 & 2 & 4 & 6 & 7 \\ 2 & 0 & 3 & 4 & 6 \\ 4 & 3 & 0 & 5 & 3 \\ 6 & 4 & 5 & 0 & 6 \\ 7 & 6 & 3 & 6 & 0 \end{bmatrix} \qquad P^{(3)} = \begin{bmatrix} 0 & 1 & 1 & 2 & 3 \\ 2 & 0 & 2 & 2 & 3 \\ 3 & 3 & 0 & 3 & 3 \\ 2 & 4 & 4 & 0 & 4 \\ 3 & 3 & 5 & 5 & 0 \end{bmatrix}$$

Note que houve alteração na nova matriz de distâncias, assim devemos atualizar a matriz de predecessores.

13

Método de Floyd-Warshall



Faço $k = 4$

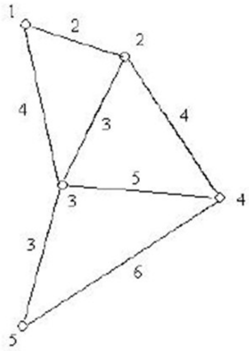
$$d_4(i,j) = Min\{d_3(i,j), d_3(i,4) + d_3(4,j)\}$$

$$D^{(4)} = \begin{bmatrix} 0 & 2 & 4 & 6 & 7 \\ 2 & 0 & 3 & 4 & 6 \\ 4 & 3 & 0 & 5 & 3 \\ 6 & 4 & 5 & 0 & 6 \\ 7 & 6 & 3 & 6 & 0 \end{bmatrix} \qquad P^{(4)} = \begin{bmatrix} 0 & 1 & 1 & 2 & 3 \\ 2 & 0 & 2 & 2 & 3 \\ 3 & 3 & 0 & 3 & 3 \\ 2 & 4 & 4 & 0 & 4 \\ 3 & 3 & 5 & 5 & 0 \end{bmatrix}$$

Como não houve alteração na matriz $D^{(4)}$, a matriz de nós predecessores não será alterada.

14

Método de Floyd-Warshall



Faço $k = 5$

$d_5(i,j) = Min\{d_4(i,j), d_4(i,5) + d_4(5,j)\}$

$$D^{(5)} = \begin{bmatrix} 0 & 2 & 4 & 6 & 7 \\ 2 & 0 & 3 & 4 & 6 \\ 4 & 3 & 0 & 5 & 3 \\ 6 & 4 & 5 & 0 & 6 \\ 7 & 6 & 3 & 6 & 0 \end{bmatrix} \qquad P^{(5)} = \begin{bmatrix} 0 & 1 & 1 & 2 & 3 \\ 2 & 0 & 2 & 2 & 3 \\ 3 & 3 & 0 & 3 & 3 \\ 2 & 4 & 4 & 0 & 4 \\ 3 & 3 & 5 & 5 & 0 \end{bmatrix}$$

Como não houve alteração na matriz $D^{(5)}$, a matriz de nós predecessores não será alterada.

Como $n = 5$ chegamos ao final do processo.

15

Método de Floyd-Warshall

$$D^{(5)} = \begin{bmatrix} 0 & 2 & 4 & 6 & 7 \\ 2 & 0 & 3 & 4 & 6 \\ 4 & 3 & 0 & 5 & 3 \\ 6 & 4 & 5 & 0 & 6 \\ 7 & 6 & 3 & 6 & 0 \end{bmatrix} \qquad P^{(5)} = \begin{bmatrix} 0 & 1 & 1 & 2 & 3 \\ 2 & 0 & 2 & 2 & 3 \\ 3 & 3 & 0 & 3 & 3 \\ 2 & 4 & 4 & 0 & 4 \\ 3 & 3 & 5 & 5 & 0 \end{bmatrix}$$

- O caminho mais curto será dado pela matriz dos predecessores $P^{(5)}$.
- Entre os nós 1 e 5, verificamos o elemento $P(1,5)=3$, na sequência verifique o elemento $P(1,3)=1$ => como foi alcançado o nó de origem, a pesquisa termina. Assim tem-se o caminho procurado: 5-3-1, ou inversamente 1-3-5.
- A distância mínima será dada pelo elemento contido na matriz $D^{(5)}$, na posição $D(1,5) = 7$.

16

Algoritmo de Dijkstra

- O algoritmo de Dijkstra (1959) é um dos algoritmos que calcula o caminho de custo mínimo entre vértices de um grafo.
- Escolhido um vértice como raiz da busca, este algoritmo calcula o custo mínimo deste vértice para todos os demais vértices do grafo.
- Ele é bastante simples e com um bom nível de performance.
- **Restrição**
  - Os arcos não podem ter valor negativo!

17

Algoritmo de Dijkstra

Seja G(V,A) um grafo orientado e **s** um vértice de G:

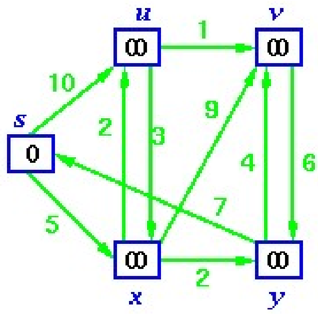
1. Atribua valor zero à estimativa do custo mínimo do vértice **s** (a raiz da busca) e infinito às demais estimativas;
2. Atribua um valor qualquer aos precedentes;
3. Enquanto houver vértice aberto:
   a. seja **k** um vértice ainda aberto cuja estimativa seja a menor dentre todos os vértices abertos;
   b. feche o vértice **k**
   c. Para todo vértice **j** ainda aberto e sucessor de **k** faça:
      1. some a estimativa do vértice **k** com o custo do arco que une **k** a **j**;
      2. caso esta soma seja melhor que a estimativa anterior para o vértice **j**, substitua-a e anote **k** como precedente de **j**.

18

Algoritmo de Dijkstra (Exemplo 1)

Inicialmente todos os vértices tem um custo infinito, exceto **s** (a raiz da busca) que tem valor 0:

vértices	**s**	**u**	**v**	**x**	**y**
estimativas	0	∞	∞	∞	∞
precedentes	-	-	-	-	-

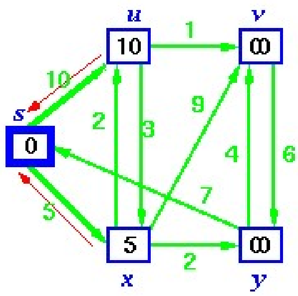


19

Algoritmo de Dijkstra (Exemplo 1)

- selecione **s** (vértice aberto de estimativa mínima)
- feche **s**
- recalcule as estimativas de u e x (adjacentes de s)

vértices	**s**	**u**	**v**	**x**	**y**
estimativas	**0**	10	∞	5	∞
precedentes	**-**	s	-	s	-

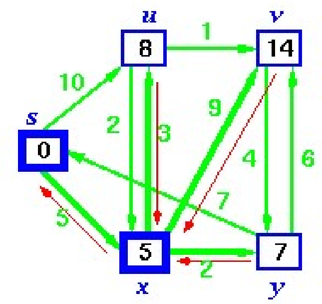


20

Algoritmo de Dijkstra (Exemplo 1)

- selecione **x** (vértice aberto de estimativa mínima)
- feche **x**
- recalcule as estimativas de u, v e y (adjacentes de x)

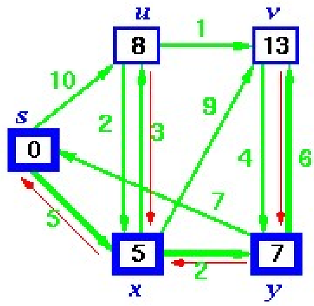
vértices	s	u	v	x	y
estimativas	0	8	14	5	7
precedentes	-	x	x	s	x



Algoritmo de Dijkstra (Exemplo 1)

- selecione **y** (vértice aberto de estimativa mínima)
- feche **y**
- recalcule a estimativa de **v** (adjacente de **y**)

vértices	s	u	v	x	y
estimativas	0	8	13	5	7
precedentes	s	x	y	s	x



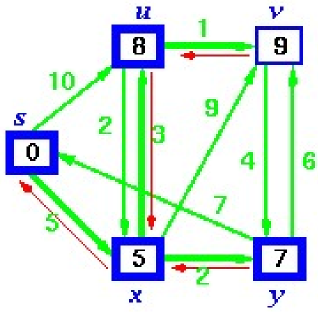
Algoritmo de Dijkstra (Exemplo 1)

•selecione **u** (vértice aberto de estimativa mínima)

•feche **u**

•recalcule a estimativa de **v** (adjacente de **u**)

vértices	s	u	v	x	y
estimativas	0	8	9	5	7
precedentes	s	x	u	s	x

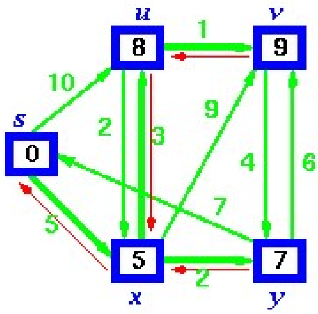


Algoritmo de Dijkstra (Exemplo 1)

•selecione **v** (vértice aberto de estimativa mínima)

•feche **v**

vértices	s	u	v	x	y
estimativas	0	8	9	5	7
precedentes	s	x	u	s	x



Algoritmo de Dijkstra (Exemplo 1)

- Quando todos os vértices tiverem sido fechados, os valores obtidos serão os custos mínimos dos caminhos que partem do vértice raiz da busca até os demais vértices do grafo.
- O caminho propriamente dito é obtido a partir dos precedentes.
  - P.ex.: analisando o caminho de custo mínimo de **s** até **v**, cujo valor é 9, tem-se que o precedente de v na última tabela é u.

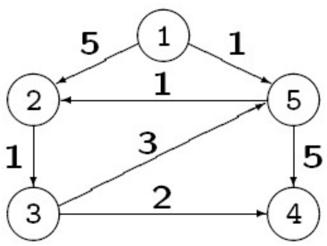
vértices	s	u	v	x	y
estimativas	0	8	9	5	7
precedentes	s	x	u	s	x

  - Logo, o precedente de u é x, de x é s (origem). Assim, o caminho de custo mínimo é: s → x → u → v

25

Algoritmo de Dijkstra (Exemplo 2)

- Dado um grafo orientado (e valorado), e dois vértices, *o* e *d*, como encontrar um caminho mais curto de *o* para *d*?

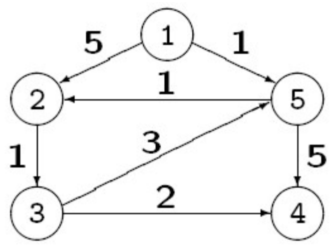


- Qual o caminho mais curto de 1 para 2 ???

26

Algoritmo de Dijkstra (Exemplo 2)

- Dado um grafo orientado (e valorado).
  - Qual o caminho de menor custo partindo do vértice 1 para os demais vértices ???

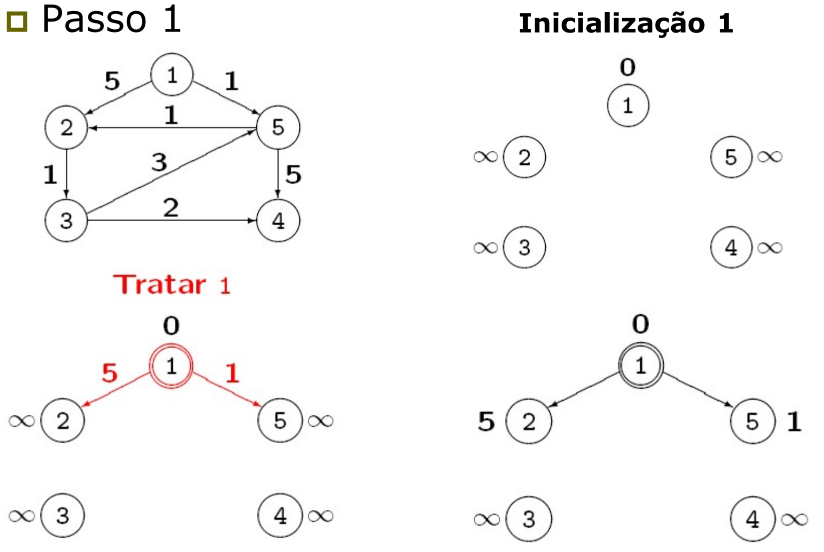


- Para todos os vértices x para os quais há caminho a partir de o, encontrar um caminho mais curto de o para x, selecionando, em cada passo, um novo vértice (e um novo caminho).

27

Algoritmo de Dijkstra (Exemplo 2)

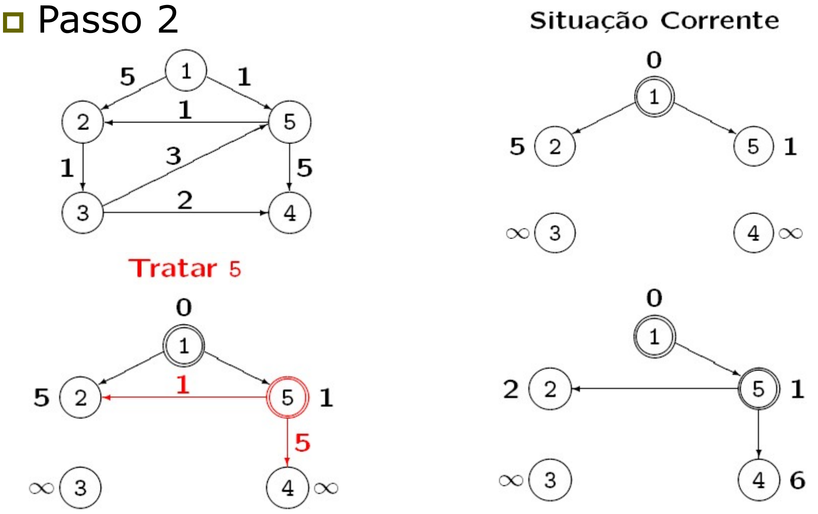
- Passo 1



28

Algoritmo de Dijkstra (Exemplo 2)

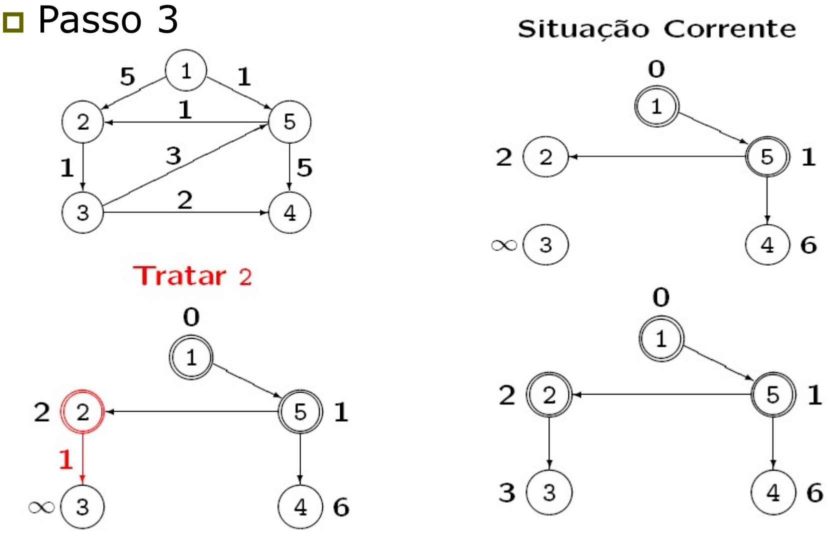
- Passo 2



29

Algoritmo de Dijkstra (Exemplo 2)

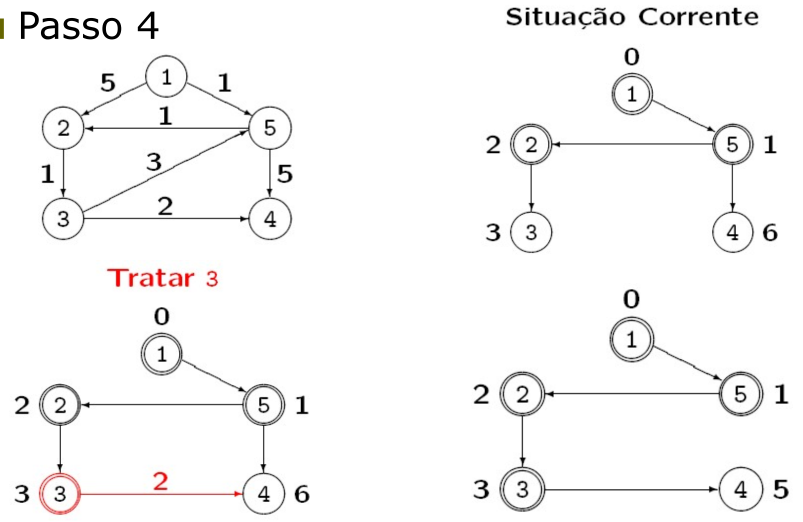
- Passo 3



30

Algoritmo de Dijkstra (Exemplo 2)

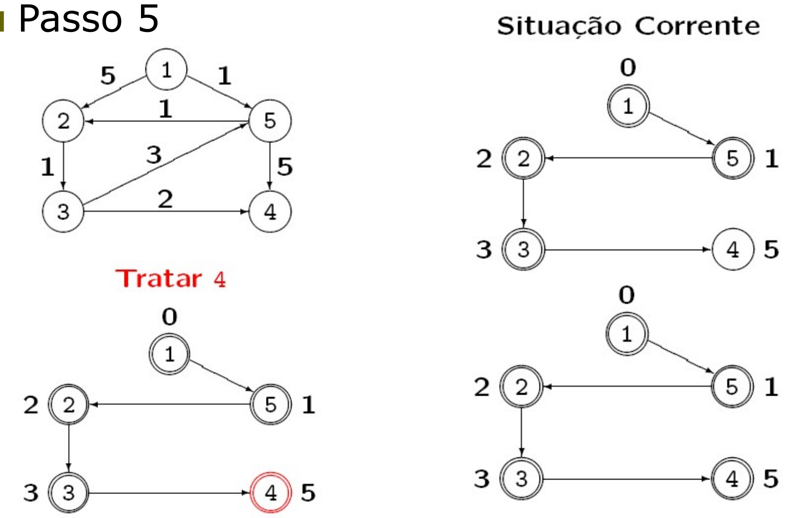
- Passo 4



31

Algoritmo de Dijkstra (Exemplo 2)

- Passo 5



32