Algoritmos e Estruturas de Dados II – SCC-203

Grafos: Caminhos mais Curtos

Gustavo Batista

Caminhos Mais Curtos: Algoritmo de Dijkstra

- A busca em largura encontra os caminhos mais curtos somente quando todas as arestas possuem o mesmo peso.
- O **algoritmo de Dijkstra** é capaz de encontrar caminhos mais curtos em um grafo ponderado com pesos diferentes entre as arestas.

Caminhos Mais Curtos: Algoritmo de Dijkstra

- Para um grafo ponderado, o peso de um caminho $c = v_0, v_1, \ldots, v_k$ é a soma de todos os pesos das arestas do caminho.
- O caminho mais curto do vértice v_0 para o vértice v_k é definido como o caminho de menor peso de v_0 para v_k.
- Diz-se que um caminho mais curto tem peso infinito se v_k não é alcançável a partir de v_0.

Caminhos Mais Curtos: Algoritmo de Dijkstra

- O algoritmo de Dijkstra encontra os caminhos mais curtos para todos os vértices de um grafo *G* a partir de uma origem única.
- Portanto, o algoritmo de Dijkstra soluciona o problema de caminhos mais curtos de origem única.

Caminhos Mais Curtos: Algoritmo de Dijkstra

- A representação de caminhos mais curtos pode ser realizada por um vetor Antecessor.
- Ao final do processamento, Antecessor contém uma árvore de caminhos mínimos definidos em termos dos pesos de cada aresta no grafo, ao invés do número de arestas.
- Caminhos mais curtos não são necessariamente únicos.

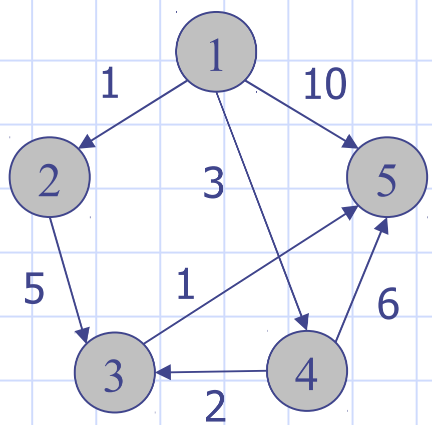
Caminhos Mais Curtos: Algoritmo de Dijkstra

- O algoritmo de Dijkstra mantém um conjunto *S* de vértices cujos pesos finais dos caminhos mais curtos desde a origem já foram determinados. Inicialmente *S* contém somente o vértice origem.
- O algoritmo de Dijkstra é um algoritmo "guloso" (*greedy*). A cada iteração, um vértice $w \in V - S$ cuja distância ao vértice origem é tão pequena quanto possível é adicionado a *S*.

Caminhos Mais Curtos: Algoritmo de Dijkstra

- Assumindo que todos os vértices possuem custos não negativos, sempre é possível encontrar um caminho mais curto do vértice origem a *w* que passa somente por vértices em *S*.
- A cada iteração, um vetor *D* armazena o custo do caminho mais curto conhecido até o momento entre o vértice origem e os demais vértices do grafo. Para os vértices em *S*, *D* possui o caminho mais curto final.
- O algoritmo termina quando todos os vértices estão em *S*.

Caminhos Mais Curtos: Algoritmo de Dijkstra



$S = \varnothing$

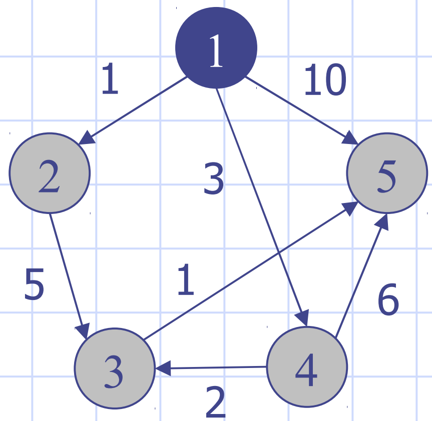
D =

	∞	∞	∞	∞
1	2	3	4	5

A =

-	-	-	-	-
1	2	3	4	5

Caminhos Mais Curtos: Algoritmo de Dijkstra



S = {1}

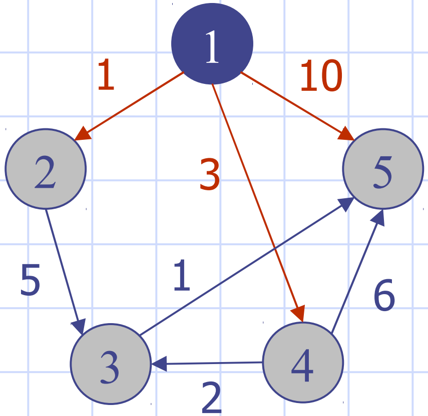
D =

1	2	3	4	5
	∞	∞	∞	∞

A =

1	2	3	4	5
-	-	-	-	-

Caminhos Mais Curtos: Algoritmo de Dijkstra



$S = \{1\}$

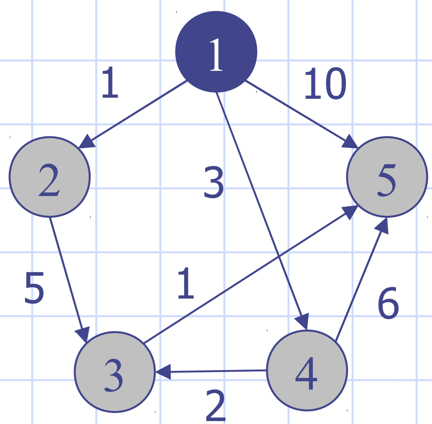
$D =$

	1	∞	3	10
1	2	3	4	5

$A =$

-	1	-	1	1
1	2	3	4	5

Caminhos Mais Curtos: Algoritmo de Dijkstra



S = {1}

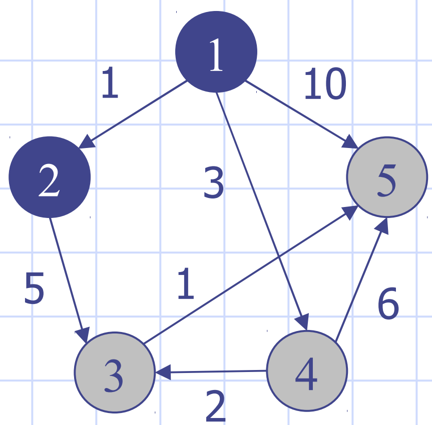
D =

	1	∞	3	10
1	2	3	4	5

A =

-	[illegible]	-	1	1
1	2	3	4	5

Caminhos Mais Curtos: Algoritmo de Dijkstra



$S = \{1, 2\}$

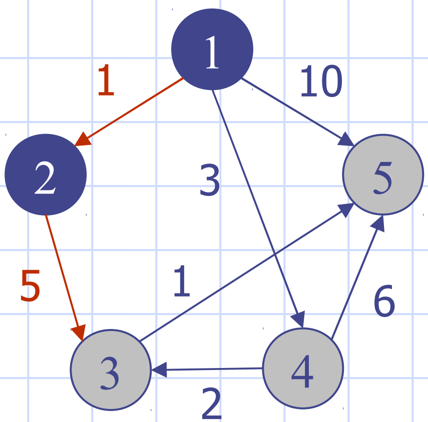
D =

	1	∞	3	10
1	2	3	4	5

A =

-	1	-	1	1
1	2	3	4	5

Caminhos Mais Curtos: Algoritmo de Dijkstra



S = {1, 2}

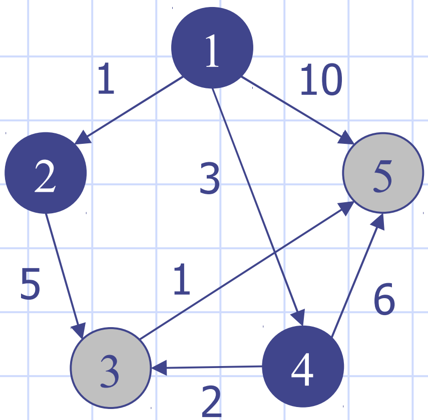
D =

1	2	3	4	5
	1	6	3	10

A =

1	2	3	4	5
-	1	2	1	1

Caminhos Mais Curtos: Algoritmo de Dijkstra



S = {1, 2, 4}

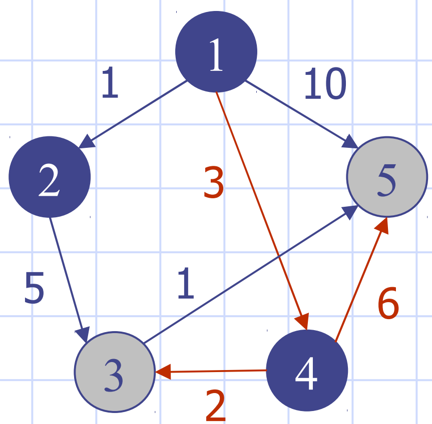
D =

1	2	3	4	5
	1	6	3	10

A =

1	2	3	4	5
-	1	2		1

Caminhos Mais Curtos: Algoritmo de Dijkstra



S = {1, 2, 4}

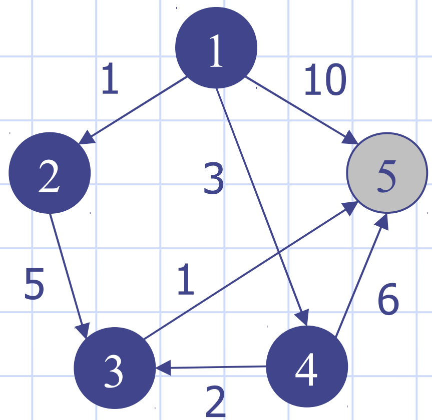
D =

1	2	3	4	5
	1	5	3	9

A =

1	2	3	4	5
-	1	4	1	4

Caminhos Mais Curtos: Algoritmo de Dijkstra



S = {1, 2, 4, 3}

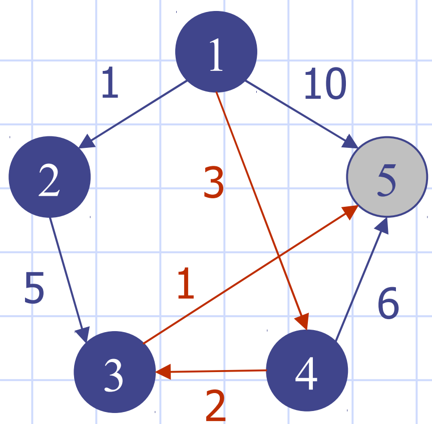
D =

1	2	3	4	5
	1	5	3	9

A =

1	2	3	4	5
-	1		1	4

Caminhos Mais Curtos: Algoritmo de Dijkstra



S = {1, 2, 4, 3}

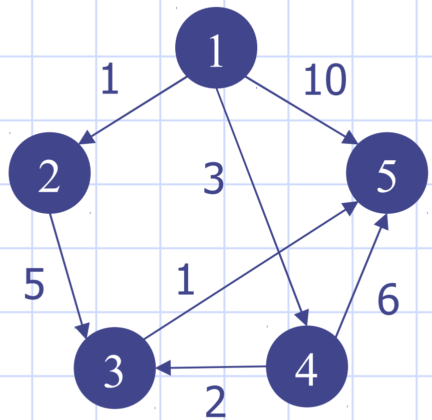
D =

1	2	3	4	5
	1	5	3	6

A =

1	2	3	4	5
-	1	2	1	3

Caminhos Mais Curtos: Algoritmo de Dijkstra



S = {1, 2, 4, 3, 5}

D =

	1	5	3	6
1	2	3	4	5

A =

	1	2	1	3
1	2	3	4	5