ESTRUTURA DE DADOS NÃO LINEARES

Prof. Renato Matroniani

EDUCAÇÃO METODISTA

ESTRUTURAS NÃO LINEARES

- GRAFOS
- HASHS
- ÁRVORES

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

EDUCAÇÃO METODISTA

GRAFOS

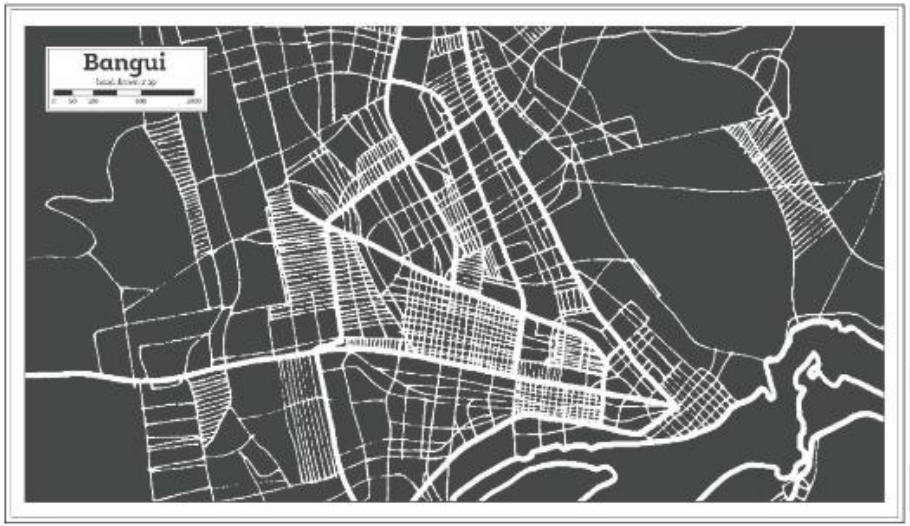
- As árvores fazem parte de uma construção de estrutura não linear, mas que segue uma determinada lógica na sua estrutura.
- No caso dos grafos, não há uma lógica ou padrão em sua construção, mas sim uma construção aleatória.

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

EDUCAÇÃO METODISTA

GRAFOS

- Exemplo prático: mapear ruas, estradas e rodovias de uma região.
- Com um mapa aéreo da cidade, através do processamento digital, extrai-se ruas e avenidas, como o exemplo:



Crédito: Shustriks/Shutterstock.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

GRAFOS

- Após o mapa ser processado, faz-se um mapeamento das estradas com o objetivo de desenvolver um software que calcule as melhores rotas a partir de um ponto de origem até um destino.
- Para representar o mapa rodoviário e os cálculos de rota, podemos utilizar uma estrutura do tipo **grafo.**
- **Nesse grafo, cada ponto de intersecção entre ruas e avenidas é um vértice desse grafo.**
- **Cada conexão entre duas intersecções, temos uma aresta desse grafo.**

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

GRAFOS

- Exemplo do mapa da cidade de Curitiba

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

GRAFOS

- No mapa de Curitiba utilizado como exemplo, os círculos pretos são os vértices de intersecção e as linhas pretas, as arestas.
- O processo de mapeamento pode ser repetido para uma região maior.
- Com o mapeamento pronto, aplica-se um algoritmo para encontrar o menor caminho do grafo, com o **Dijkstra**.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

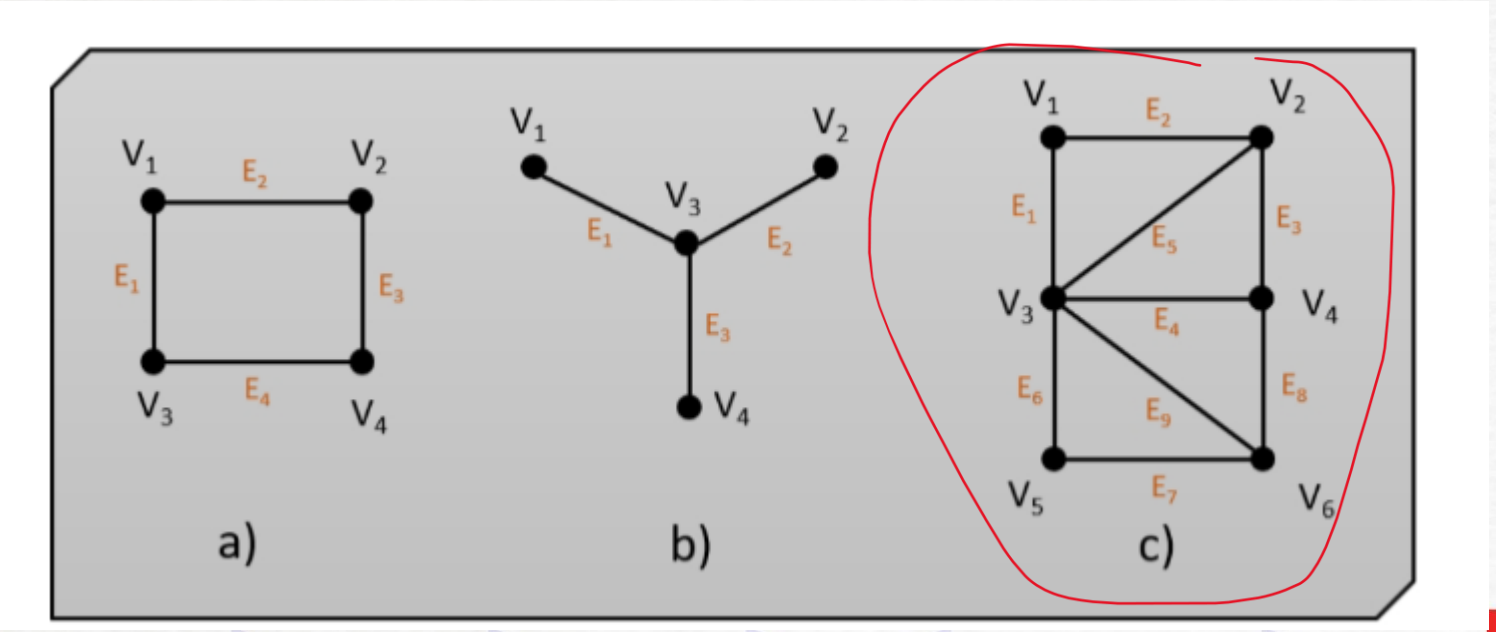
DEFINIÇÃO DE GRAFOS

- Um grafo G é um conjunto de vértices conectados por meio de arestas sem uma distribuição fixa ou padronizada.
- Os vértices V de um grafo são seus pontos. Cada ponto pode ser um ponto de encontro entre caminhos (rotas) de um grafo. Um vértice também pode conter informações relevantes para o grafo, como informações de cadastro (dependendo da aplicação).
- As arestas E são linhas de conexão entre os vértices. Cada aresta conecta dois vértices. Nem todo vértice precisa ter uma aresta conectada.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

EXEMPLOS DE GRAFOS



EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

DEFINIÇÃO DE GRAFOS

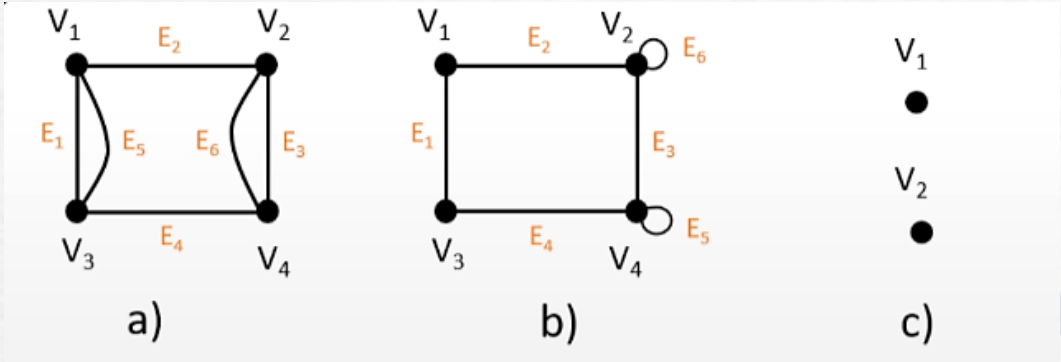
- Para a construção do grafo, não existe uma regra fixa. Qualquer arranjo de vértices e arestas pode ser considerado um grafo.
- Chamamos de grafo completo quando existe uma, e somente uma aresta para cada par distinto de vértices.
- Chamamos de grafo trivial aquele que apresenta um único vértice.
- Assim como temos grau dos nós na estrutura árvore, temos o grau de cada nó de um vértice na estrutura grafo.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

DEFINIÇÃO DE GRAFOS

- As arestas múltiplas ocorrem em construções particulares de grafos, e são arestas que estão conectadas aos mesmos vértices.
- Um laço acontece quando uma aresta contém somente um vértice ao qual se conectar, iniciando e terminando nele.
- Um grafo desconexo temos pelo menos um vértice sem nenhuma aresta.



EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

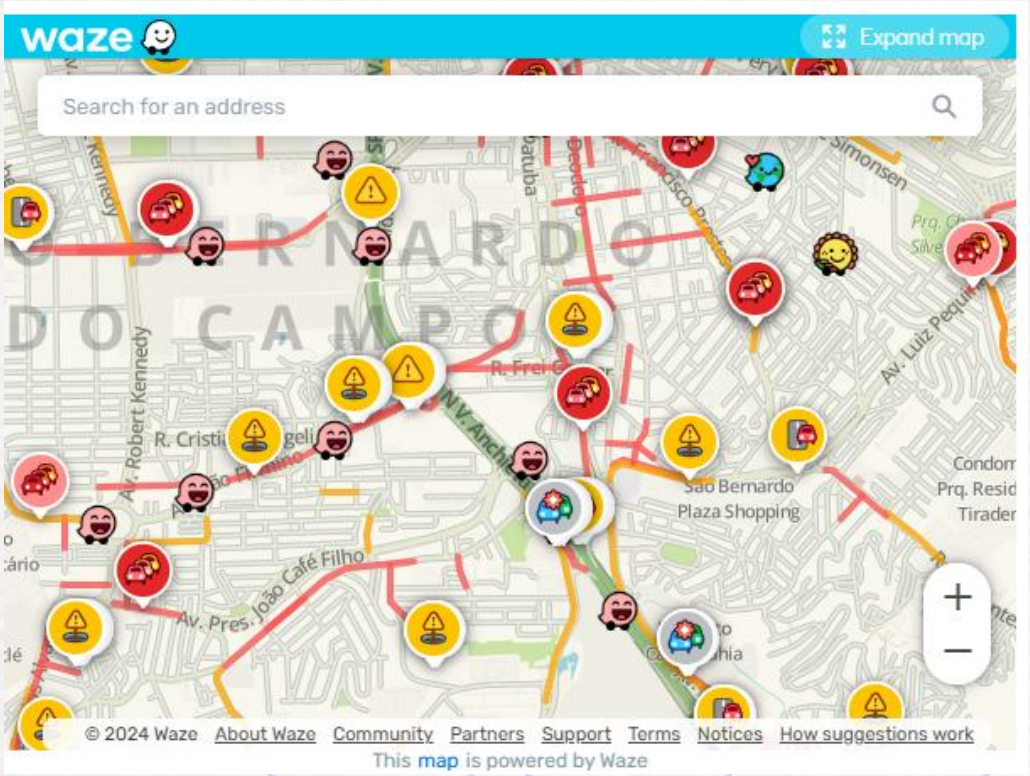
DEFINIÇÃO DE GRAFOS

- Podemos atribuir peso às arestas. O peso da aresta representa um custo atrelado àquele caminho.
- No exemplo da estrada, em uma rota entre A e B de uma cidade mapeada, os pesos nas arestas podem ser calculados a partir do tráfego de veículos da rua, por exemplo. Quanto mais veículos, maior o peso e menor o desempenho na escolha daquela rota.
- Quais outras variáveis poderíamos colocar em um algoritmo de rotas e que trazem esse “peso” para cada uma dessas arestas, como acontece no aplicativo Waze?

EDUCAÇÃO METODISTA

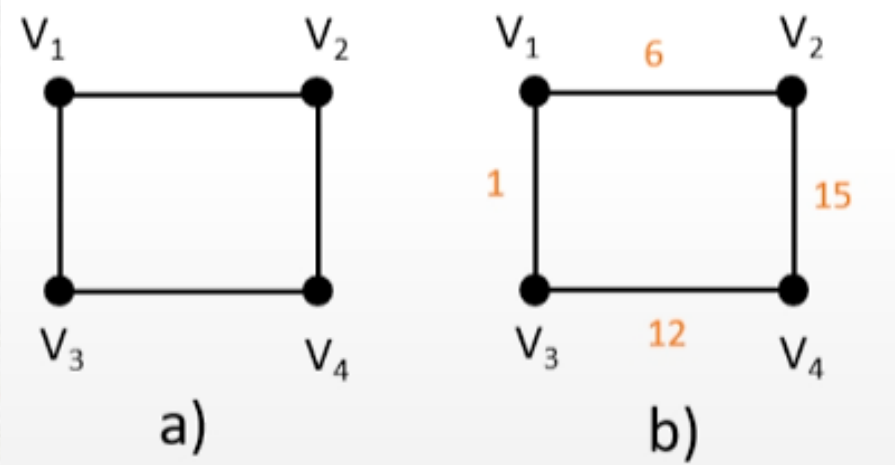
Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

DEFINIÇÃO DE GRAFOS



Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

DEFINIÇÃO DE GRAFOS



Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

REPRESENTAÇÃO DE GRAFOS

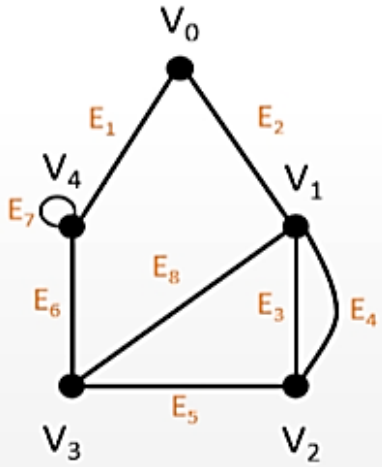
- Os grafos podem ser representados matematicamente ou graficamente. As representações matemáticas acabam por ser ideais para a implementação em um algoritmo, já que podem ser manipuladas da forma que o desenvolvedor necessitar.
- As outras formas de apresentação são: matriz de incidências, matriz de adjacências e lista de adjacências

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

MATRIZ DE INCIDÊNCIAS

- A representação de um grafo por uma matriz de incidências consiste em criar uma matriz de dimensão V x E [número de vértices *(vertex)* por número de arestas *(edge*)].
- Exemplo de um grafo com 5 V e 8 E:



VÉRTICES \ ARESTAS	1	2	3	4	5	6	7	8
0	1	1	0	0	0	0	0	0
1	0	1	1	1	0	0	0	1
2	0	0	1	1	1	0	0	0
3	0	0	0	0	1	1	0	1
4	1	0	0	0	0	1	2	0

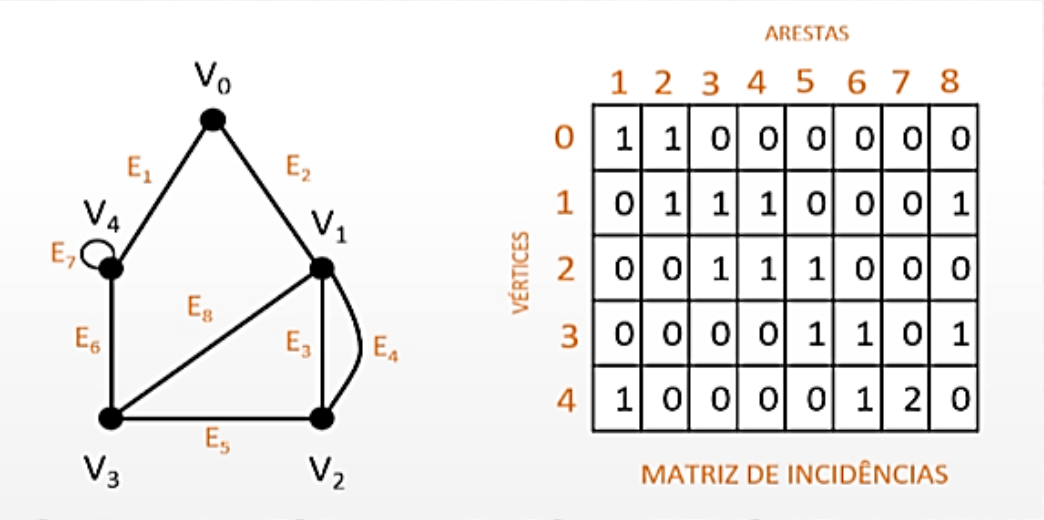
MATRIZ DE INCIDÊNCIAS

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

MATRIZ DE INCIDÊNCIAS

- Um grafo com 5 vértices e 8 arestas resulta em uma matriz 5x8.
- Essa matriz é então preenchida para indicar o grafo construído.



VÉRTICES \ ARESTAS	1	2	3	4	5	6	7	8
0	1	1	0	0	0	0	0	0
1	0	1	1	1	0	0	0	1
2	0	0	1	1	1	0	0	0
3	0	0	0	0	1	1	0	1
4	1	0	0	0	0	1	2	0

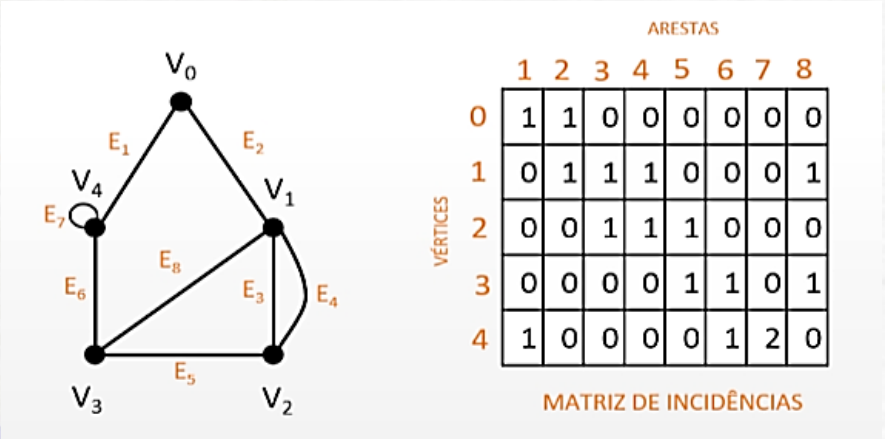
MATRIZ DE INCIDÊNCIAS

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

MATRIZ DE INCIDÊNCIAS

- Padrão de preenchimento:
- 0 → caso o vértice não faça parte da ligação.
- 1 → caso o vértice faça parte da ligação
- 2 → caso a aresta seja do tipo laço.

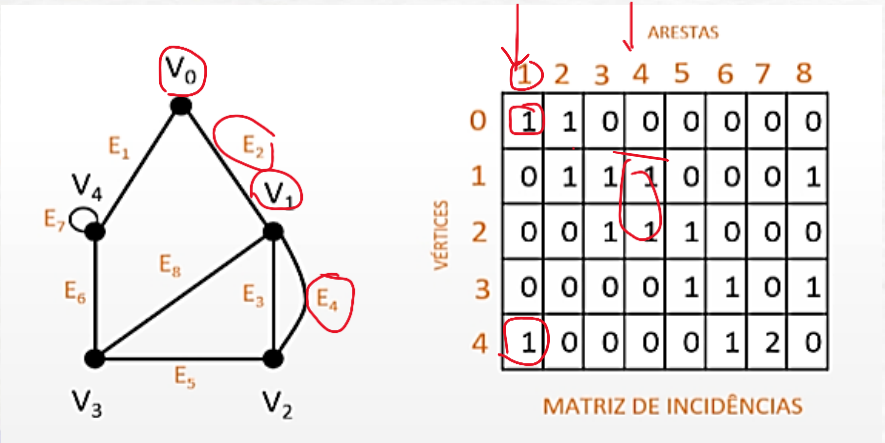


EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

MATRIZ DE INCIDÊNCIAS

- No exemplo, a aresta E1 possui uma ligação entre o vértice 0 e o vértice 4 → as linhas desses vértices recebem os valores 1, enquanto todas as outras linhas recebem o valor 0.
- O vértice 7 é um laço e recebe o valor 2.

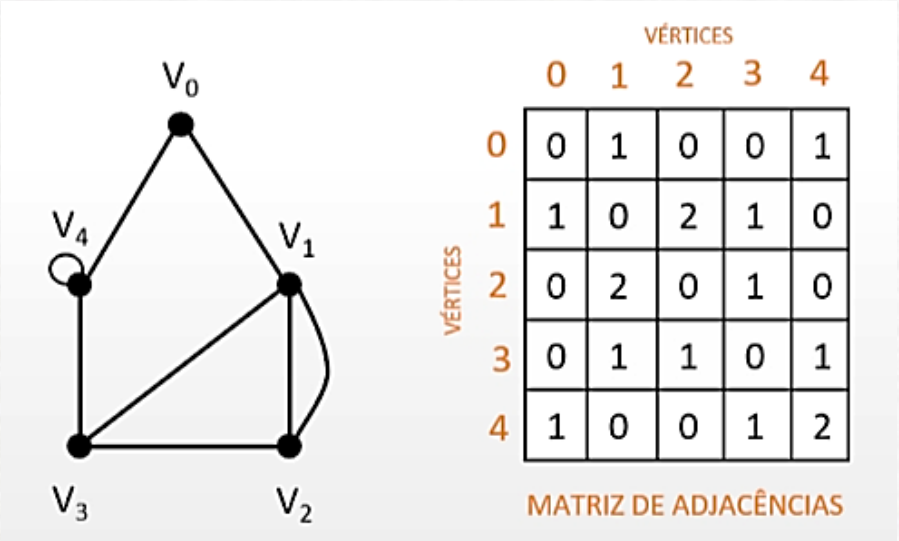


EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

MATRIZ DE ADJACÊNCIAS

- A representação de um grafo por uma matriz de adjacências consiste em criar uma matriz quadrada de dimensão V (número de vértices do grafo).

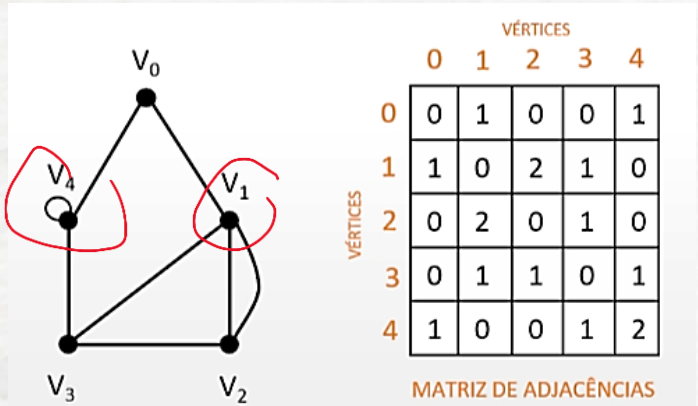


EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

MATRIZ DE ADJACÊNCIAS

- A matriz é então preenchida com valores 0, 1 e 2.
- Para o preenchimento, observamos cada uma das linhas dos vértices e preenchemos na matriz, seguindo a regra:
- 0 → caso o outro vértice não tenha conexão com o vértice analisado.
- 1 → caso o outro vértice tenha conexão com o vértice analisado.
- 2 → caso o outro vértice tenha conexão com o vértice analisado e seja um laço.

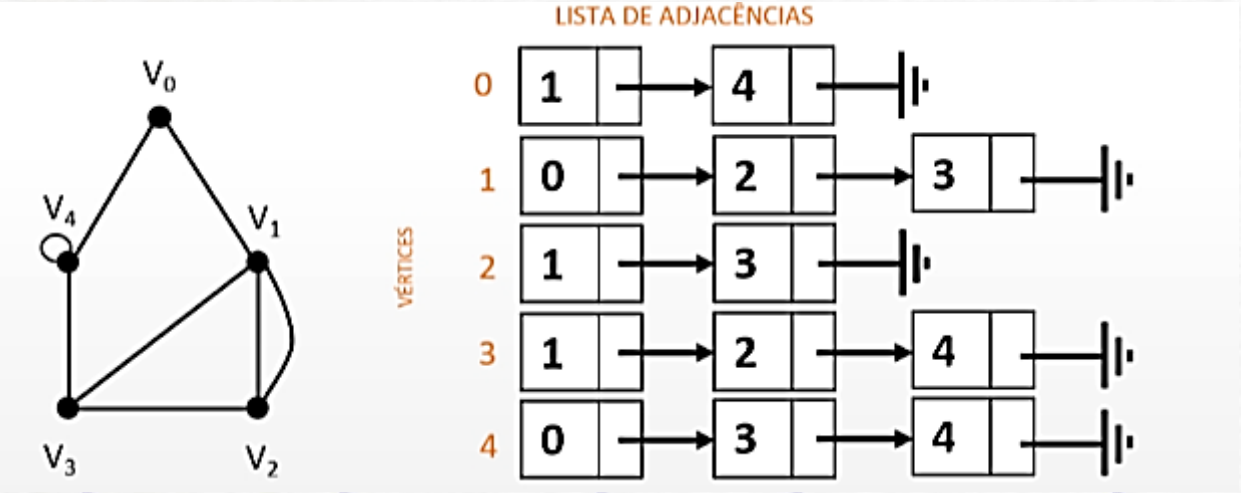


EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

LISTA DE ADJACÊNCIAS

- A representação de um grafo por uma lista de adjacências é bastante adotada em programação, pois trabalha com **listas encadeadas** e manipulação de **ponteiros de dados**.



EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

LISTA DE ADJACÊNCIAS

- A proposta dessa representação é criar um vetor ou uma lista encadeada do tamanho da quantidade de vértices existentes no grafo.
- Em cada posição desse vetor, teremos uma lista encadeada contendo os endereços dos vizinhos de cada vértice.
- Conceitualmente, vizinhos de um vértice são todos os vértices que se conectam diretamente a ele por meio de uma aresta.
- Assim, teremos uma lista encadeada de vizinhos para cada posição do vetor de vértices criados.
- Na lista, cada vizinho apontará para outro vizinho daquele mesmo nó.

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

LISTA DE ADJACÊNCIAS

- No exemplo, temos 5 vértices e, portanto, podemos ter um vetor de dimensão 5. Cada posição do vetor terá o endereço do *head* (também vizinho) para uma lista encadeada de vizinhos daquele vértice.
- O vértice V0 está na primeira posição do vetor. Ele contém, como vizinhos, o vértice 1 e o vértice 4. Assim, na posição zero do vetor, temos o endereço para um dos vizinhos de V0.
- Esse vizinho será o *head* de uma lista encadeada.
- O vizinho escolhido para ser o *head* foi o vértice 1. Assim o V1 na lista apontará para V4:

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

LISTA DE ADJACÊNCIAS

- Para analisar o algoritmo de criação da estrutura de dados do tipo lista de adjacências, podemos:
- 1) declaramos um vetor de dimensão igual ao número de vértices do grafo. Esse vetor será do tipo ponteiros de registros. Assim, cada posição do vetor poderá conter uma estrutura do tipo lista encadeada, armazenando o primeiro elemento da lista, que é o *head*.

```
1  //Vetor do tamanho do número de vértices do grafo
2  //O vetor conterá o endereço o primeiro vizinho (variável ponteiro)
3  Vertices[NUMERO_DE_VERTICES]: ListaDeVizinhos[->)
4
5  //Lista encadeada de vizinhos de cada vértice
6  registro ListaDeVizinhos
7     prox: ListaDeVizinhos[->)
8  fimregistro
```

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

LISTA DE ADJACÊNCIAS

- 2) substituir o vetor por uma estrutura do tipo lista. Assim teremos duas listas encadeadas, uma chamada de vertical, contendo os vértices e os endereços para os primeiros vizinhos, e uma lista horizontal, contendo as listas de vizinhos de cada vértice.

```
1   //Lista encadeada com os vértices e o primeiro vizinho
2   registro Vertices
3       numero: inteiro
4       prox: Vertices[->)
5   fimregistro
6
7   //Lista encadeada de vizinhos de cada vértice
8   registro ListaDeVizinhos
9       prox: Vertices[->)
10  fimregistro
```

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Como trabalharemos com um grafo ponderado, precisamos adaptar nossa lista de adjacências para que o registro também armazene os pesos das arestas. Podemos deixar nossa lista como vemos na Figura 28, alterando o registro de vértices para também conter um campo para pesos.

O algoritmo investigado neste tema será o **algoritmo de Dijkstra**, que recebeu esse nome em homenagem ao cientista da computação holandês Edsger Dijkstra, que publicou o algoritmo pela primeira vez em 1959. Esse algoritmo é o mais tradicional para realizar a tarefa de encontrar um caminho. Para realizar tal procedimento, podemos utilizar um **grafo ponderado**, ou seja, aquele com pesos distintos nas arestas.

Aplicaremos o algoritmo de Dijkstra em um grafo ponderado e obteremos as menores rotas, partindo de uma origem, para todos os outros vértices do grafo. Todo esse algoritmo será apresentado utilizando uma **métrica aditiva**. Isso significa que essa métrica encontrará a menor rota considerando o menor peso somado entre os caminhos.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Aplicaremos o algoritmo de Dijkstra em um grafo ponderado e obteremos as menores rotas, partindo de uma origem, para todos os outros vértices do grafo. Todo esse algoritmo será apresentado utilizando uma **métrica aditiva**. Isso significa que essa métrica encontrará a menor rota considerando o menor peso somado entre os caminhos.

Um exemplo prático de métrica aditiva é o tráfego de veículos em rodovias. Quanto maior o tráfego, pior o desempenho da aresta do grafo.

Existem outros tipos de métricas. Na multiplicativa, por exemplo, o melhor caminho é encontrado calculando-se o produto dos pesos das arestas. O maior valor dentre os produtos é a melhor rota. Um exemplo dessa métrica é o limite de velocidade das estradas: quanto maior o limite, mais rápido o veículo anda, e, portanto, melhor é aquela rota/aresta.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Figura 28 – Pseudocódigo de lista de adjacências para grafo ponderado

```
1   //Lista encadeada com os vértices e o primeiro vizinho
2   registro Vertices
3       numero: inteiro
4       peso: inteiro
5       prox: Vertices[->)
6   fimregistro
7
8   //Lista encadeada de vizinhos de cada vértice
9   registro ListaDeVizinhos
10      prox: Vertices[->)
11  fimregistro
```

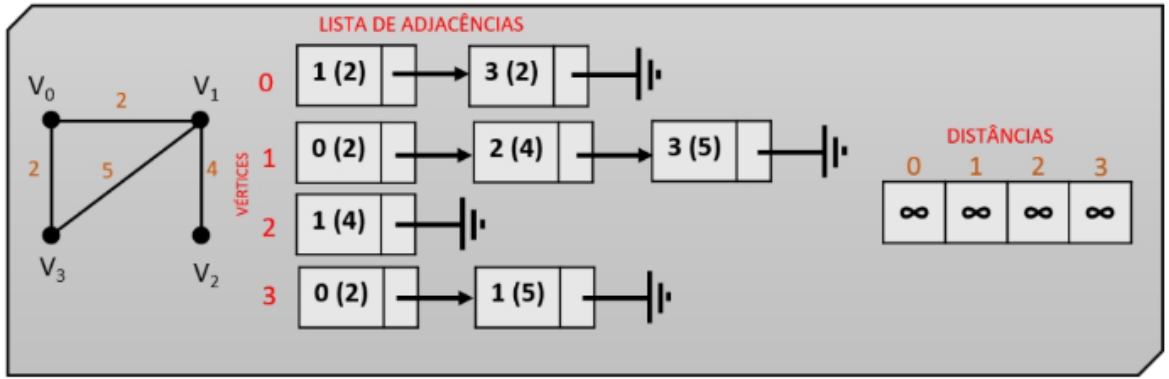
EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Na Figura 29, vemos o estado inicial da lista de adjacências ponderadas. Os valores que estão entre parênteses são os pesos das arestas. Por exemplo, na lista encadeada do vértice V_1, temos como primeiro elemento V_0, e o peso da aresta entre eles está indicado pelo valor 2 entre parênteses.

Figura 29 – Algoritmo de Dijkstra: partindo de V_1: estado inicial.



Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

O algoritmo do caminho mínimo precisa calcular as menores rotas de um vértice para todos os outros. Portanto, um vetor com distâncias fica armazenado. Nesse vetor, todas as rotas iniciam com um valor infinito, ou seja, como se o caminho de um vértice até o outro não fosse conhecido. À medida que os trajetos vão sendo calculados, os pesos das rotas são alterados.

Explicaremos o funcionamento do algoritmo iniciando pelo vértice V_1. Assim, encontraremos a rota de V_1 para todos os outros vértices existentes no grafo ponderado.

Na Figura 30, iniciamos a primeira etapa da operação do método. Como partiremos de V_1, já iniciamos acessando a lista de vizinhos desse vértice e calculando as rotas para eles. Encontramos a rota de V_1 para ele mesmo. Nesse

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

O algoritmo do caminho mínimo precisa calcular as menores rotas de um vértice para todos os outros. Portanto, um vetor com distâncias fica armazenado. Nesse vetor, todas as rotas iniciam com um valor infinito, ou seja, como se o caminho de um vértice até o outro não fosse conhecido. À medida que os trajetos vão sendo calculados, os pesos das rotas são alterados.

Explicaremos o funcionamento do algoritmo iniciando pelo vértice V_1. Assim, encontraremos a rota de V_1 para todos os outros vértices existentes no grafo ponderado.

Na Figura 30, iniciamos a primeira etapa da operação do método. Como partiremos de V_1, já iniciamos acessando a lista de vizinhos desse vértice e calculando as rotas para eles. Encontramos a rota de V_1 para ele mesmo. Nesse caso, o vértice de origem para ele mesmo terá sempre peso zero, conforme apresentado no vetor de distâncias.

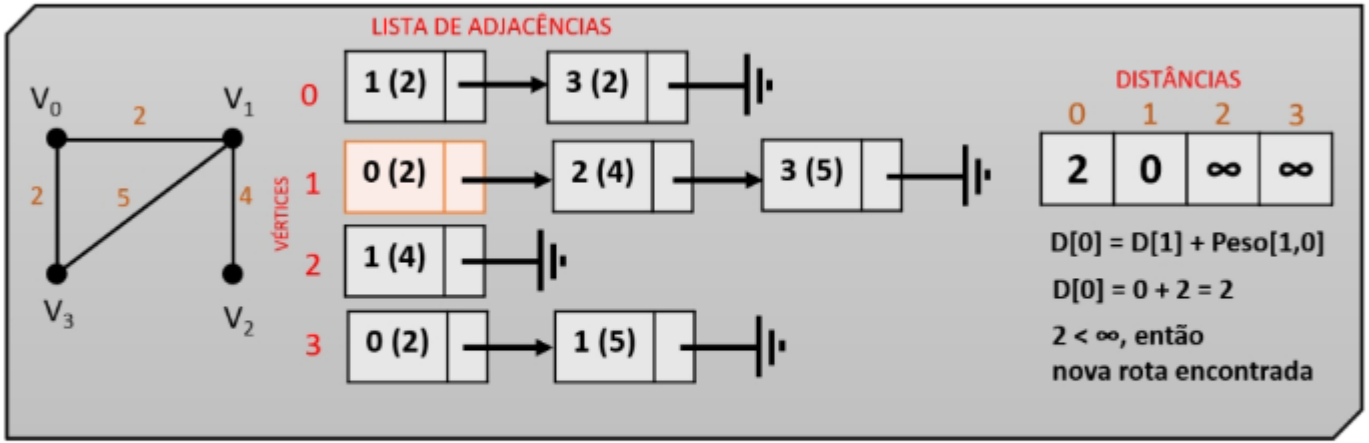
Em seguida, acessamos o *head* da lista encadeada de V_1, que é V_0, com um peso de aresta de 2. Assim, o cálculo da distância entre eles será o peso em $V_1 = 0$ acrescido do peso dessa aresta, resultando no valor 2. Esse valor, por ser menor do que infinito, é colocado no vetor, na posição de V_0. O cálculo é apresentado no canto inferior direito da Figura 30.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Figura 30 – Algoritmo de Dijkstra: partindo de V_1: etapa 1



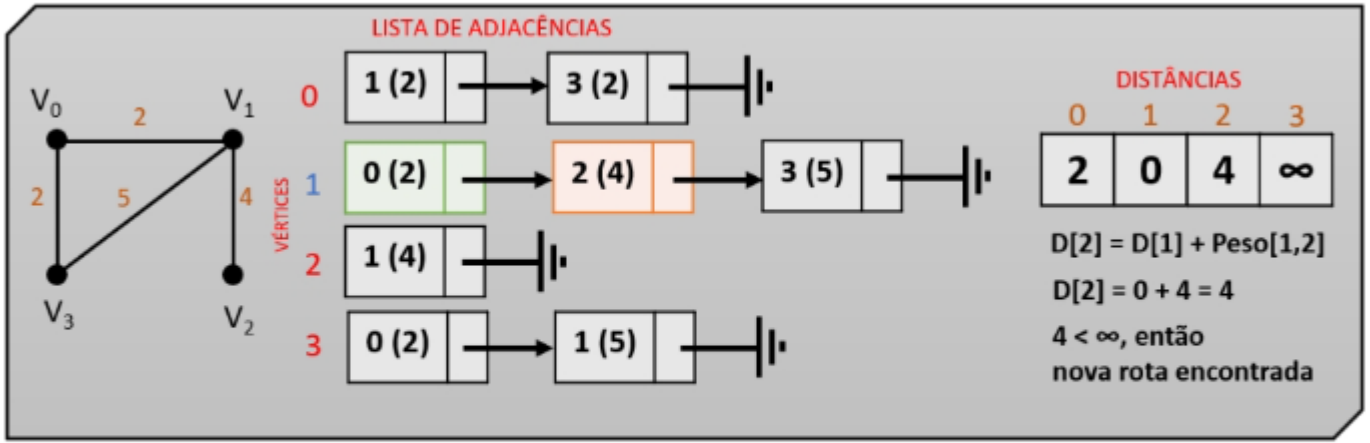
Seguimos na Figura 31 acessando o segundo elemento da lista de vizinhos de V_0. Em V_2, fazemos o peso de $V_0 = 0$, acrescido da aresta para V_2, resultando no valor 4, que por ser menor do que infinito é colocado como rota válida entre V_1 e V_2.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Figura 31 – Algoritmo de Dijkstra: partindo de V_1: etapa 2



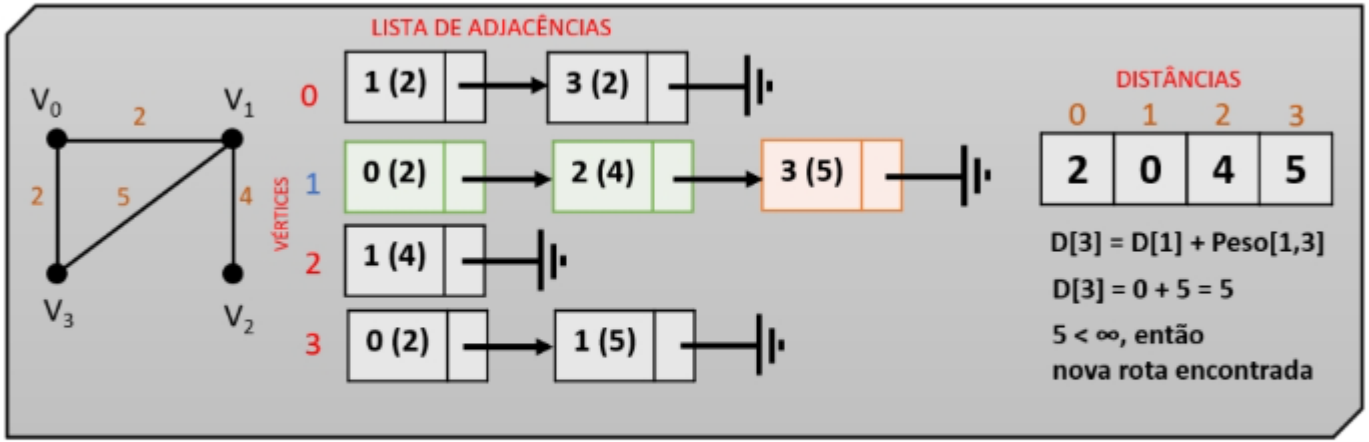
A etapa 3 é mostrada na Figura 31. De forma semelhante às duas etapas anteriores, calculamos a distância entre V_1 e V_3 e obtemos o resultado 5; assim, colocamos o vetor de distâncias na posição de V_3.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Figura 32 – Algoritmo de Dijkstra: partindo de V_1: etapa 3



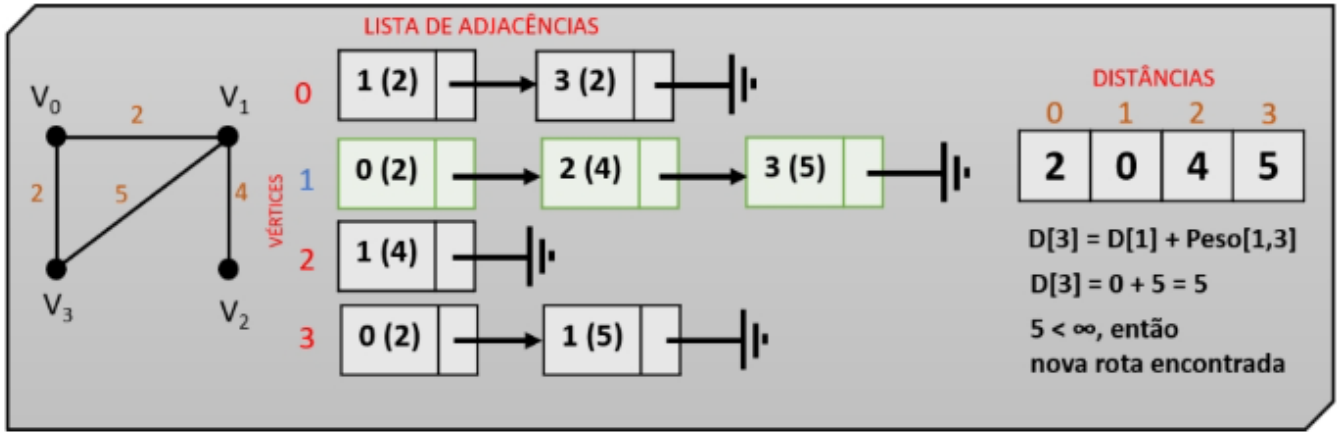
Todos os vizinhos do vértice V_1 têm suas rotas calculadas. Um panorama geral é apresentado na Figura 33, indicando que todos os vizinhos foram calculados (cor verde).

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Figura 33 – Algoritmo de Dijkstra: partindo de V_1: etapa 4



Precisamos passar para o próximo vértice e calcular as rotas partindo de V_1, passando por esse vértice intermediário. O vértice que calcularemos agora é o de menor caminho no vetor distâncias e que ainda não tenha sido marcado. Será, portanto, o vértice V_0, com distância 2 para V_1.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Na Figura 34, temos V_0 marcado e seu primeiro vizinho, acessado. Seu primeiro vizinho é o próprio vértice de origem V_0. Isso significa que precisamos calcular o peso da rota que inicia em V_0, passa para V_1 e retorna para V_0. O peso dessa rota será o peso já calculado até $V_0 = 2$, acrescido do peso da aresta de V_0 para V_1 $(2 + 2 = 4)$.

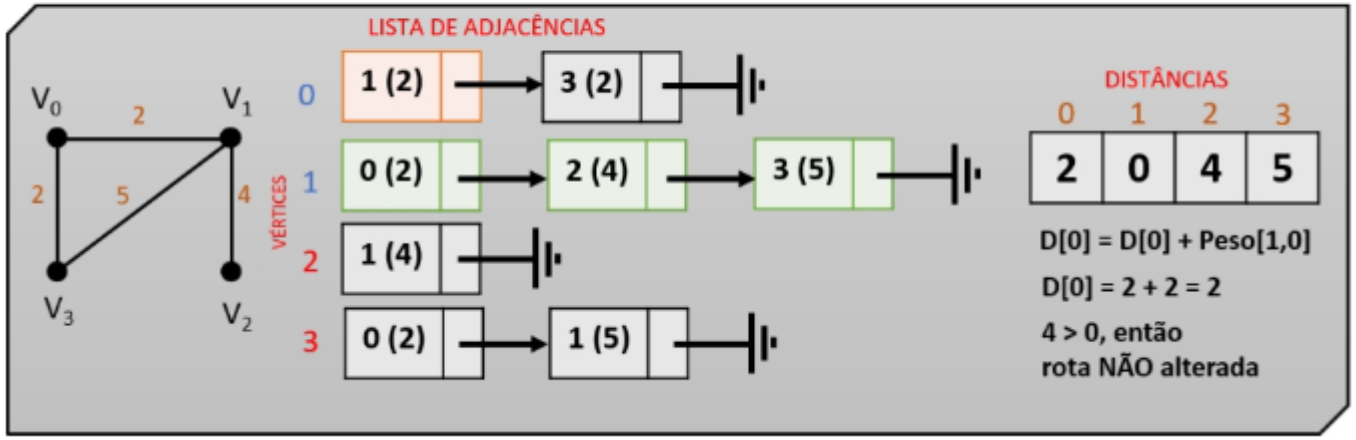
Fazer um trajeto partindo da origem, passando por outro vértice e depois retornando para a origem sempre resultará em um peso na rota superior, se comparado com o peso da origem individualmente $(V_1 = 0)$. Portanto, essa nova rota não substitui aquela já existente, como vemos na Figura 34.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Figura 34 – Algoritmo de Dijkstra: partindo de V_1: etapa 5



Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Na Figura 35, seguimos para o próximo vizinho de V_0, o V_3. Isso significa que calcularemos uma rota $V_1 \rightarrow V_0 \rightarrow V_3$. O peso dessa rota será o peso até V_0 já calculado e de valor 2, somado ao peso da aresta V_0 e V_3, que também é 2. Assim, $2 + 2 = 4$.

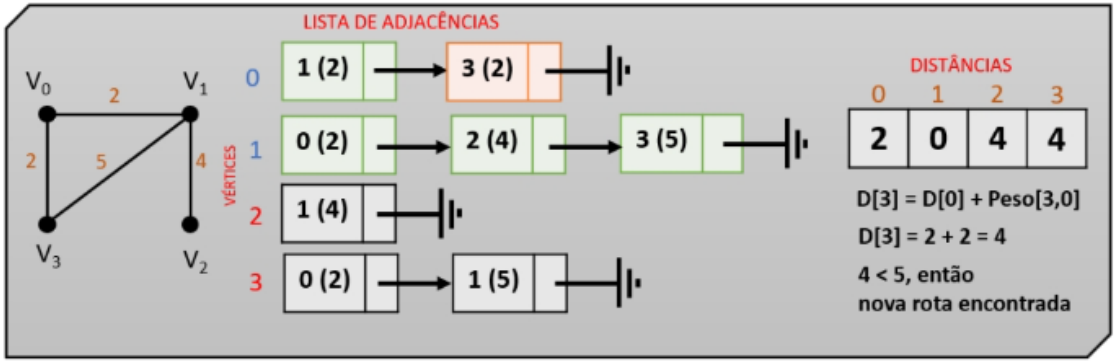
Observe agora algo interessante: o peso da rota de V_1 até V_3, passando por V_0, resultou em peso 4. Anteriormente, calculamos o peso da rota direta entre V_1 e V_3, cujo valor resultou em 5. Portanto, a rota que passa por V_0 tem um peso menor ($4 < 5$), resultando em uma nova rota até V_3. Note que uma rota com peso menor, mesmo passando por um vértice a mais, acaba resultando em um caminho menor que a outra.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Figura 35 – Algoritmo de Dijkstra: partindo de V_1: etapa 6



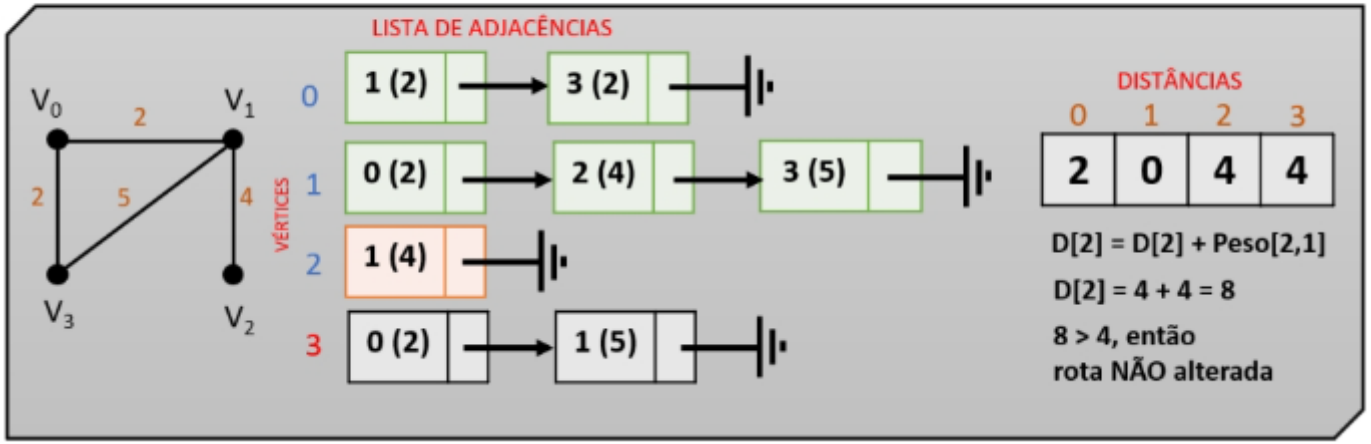
Na Figura 36, já encerramos nossos cálculos passando pelo vértice V_0. Seguimos para o próximo vértice não visitado e de menor distância no vetor, o vértice V_2. O único vizinho de V_2 é o vértice origem V_1. Fazer o trajeto $V_1 \rightarrow V_2 \rightarrow V_1$ tem um custo atrelado inferior ao custo zero de permanecer em V_1. Portanto, essa rota não é válida.

EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Figura 36 – Algoritmo de Dijkstra: partindo de V_1: etapa 7

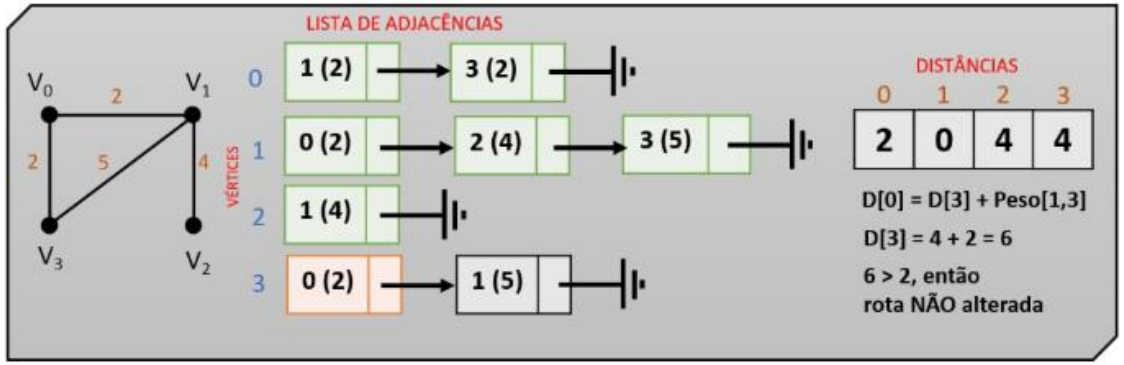


Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Na Figura 37, seguimos para o próximo, e último, vértice não visitado e de menor distância no vetor: o vértice V_3. O primeiro vizinho de V_3 é V_0. Assim, faremos o trajeto $V_1 \rightarrow V_3 \rightarrow V_0$. O custo até V_3 é 4 e o peso de V_3 até V_0 é 2, resultando em 6 (4 + 2), custo superior ao da rota atual, que é 4.

Figura 37 – Algoritmo de Dijkstra: partindo de V_1: etapa 8



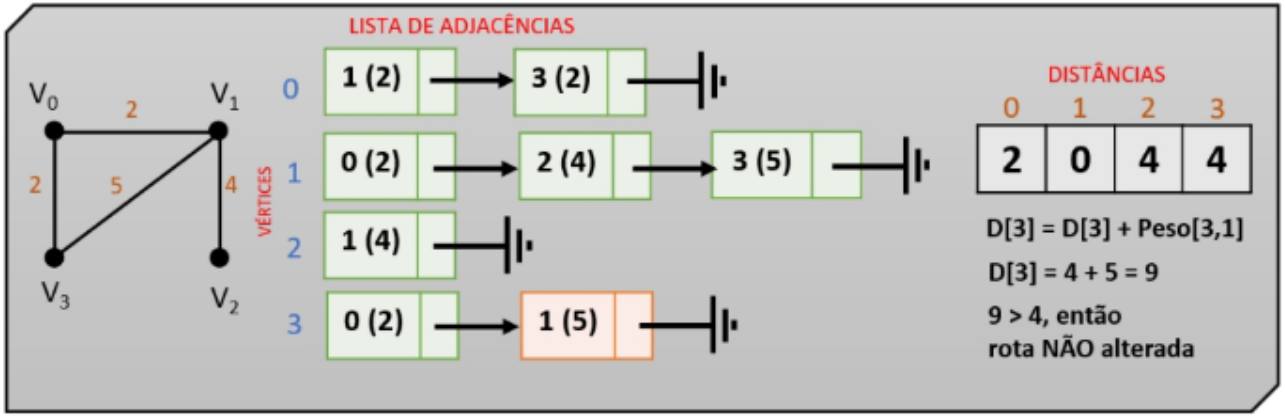
EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Na Figura 38, temos o segundo vizinho de V_3, o V_1. Assim, faremos o trajeto $V_1 \rightarrow V_3 \rightarrow V_1$. O custo até V_3 é 4 e o peso de V_3 até V_1 é 5, resultando em 9 (4 + 5), custo superior ao da rota atual que é 0.

Figura 38 – Algoritmo de Dijkstra: partindo de V_1: etapa 9



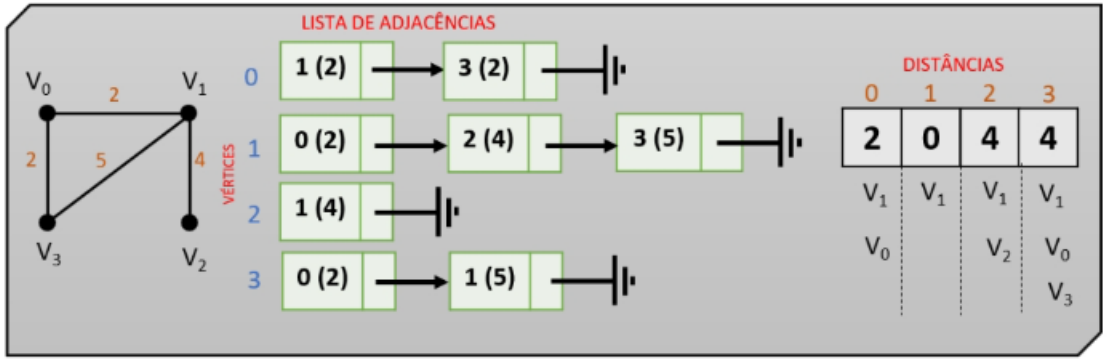
EDUCAÇÃO METODISTA

Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.

ALGORITMO DE CAMINHO MÍNIMO

Na Figura 39, já percorremos todos os vizinhos de todos os vértices e calculamos todas as rotas possíveis. Desse modo, as distâncias resultantes estão no vetor de distâncias, e abaixo de cada valor está a sequência de vértices para aquela rota. Por exemplo, para atingirmos o vértice V_3, a melhor rota encontrada foi $V_1 \rightarrow V_0 \rightarrow V_3$, e não $V_1 \rightarrow V_3$ diretamente.

Figura 39 – Algoritmo de Dijkstra: partindo de V_1: etapa 10



Texto base/Fonte: BORIN, Vinicius Pozzobon. Estrutura de dados. 1. ed. São Paulo: Contentus, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br.