FIAP ON

CÓDIGOS DE ALTA PEFORMANCE

GRAFOS

PATRICIA MAGNA

8

LISTA DE FIGURAS

SUMÁRIO

8 GRAFOS

8.1 Teoria dos Grafos: origem e aplicações

A história da teoria dos grafos inicia em meados de 1735, quando o matemático Euler resolveu o problema conhecido como problema das pontes de Königsberg. Na cidade de Königsberg (atual Kaliningrado), antiga capital da Prússia Oriental, o rio Pregel circunda uma ilha e separa a cidade em quatro zonas que, no século XVII, estavam ligadas por sete pontes, como mostra a Figura “Problema das pontes de Königsberg”**.** Havia uma questão baseada em uma especulação: seria possível passar pelas 7 pontes sem ter que cruzar mais de uma vez nenhuma delas?

O objetivo de Euler era demonstrar que tal feito era impossível, e o fez dando início ao raciocínio topológico, que é o marco da teoria dos grafos. A solução apresentada por Euler é considerada o primeiro teorema na teoria de grafos.

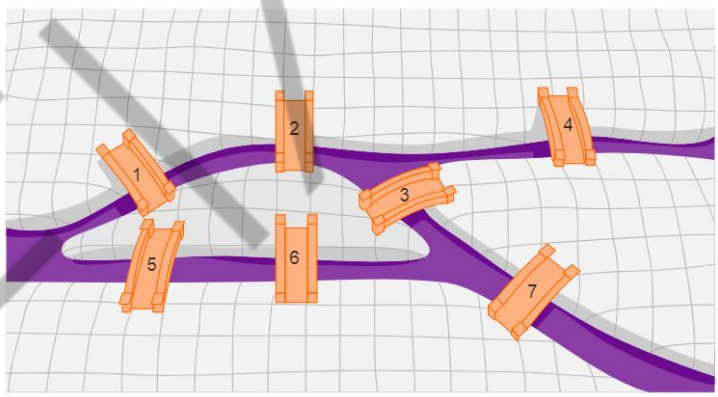


Figura 8.1–Problema das pontes de Königsberg
Fonte: https://www.britannica.com/topic/graph-theory (2015).

A teoria dos grafos tem como foco as aplicações em que é necessário o estudo de objetos combinatórios. Por exemplo, a combinação que especifica a exata sequência em que as pontes devem ser percorridas. Assim, grafos representam um bom modelo para muitos problemas em diversas áreas, tais como na representação de qualquer rede de rotas de transporte (um mapa de estradas, por exemplo), rede de comunicação (como em uma rede de computadores) ou rotas de distribuição de

produtos ou serviços, como dutos de gás ou água etc. A estrutura química de uma molécula também pode ser representada por um grafo.

Muitos desses problemas tornaram-se célebres porque representam um interessante desafio intelectual e, além disso, têm importantes aplicações práticas. É claro que esse tipo de problemas tem como objetivo resolver questões de alta complexidade computacional. Por isso, a utilização da teoria de grafos vem trazer a solução de forma que diminua a complexidade algorítmica, ou seja, tempo de processamento menor e, com isso, obtenção de tempo de resposta mais curto.

8.2 Algumas Importantes Definições Formais de Grafos

Um Grafo G(V, E) é uma estrutura matemática constituída pelos conjuntos:

- V, finito e não vazio de n vértices, e
- E, de m arestas, que são pares não ordenados de elementos de V.

Graficamente, é representado por uma figura com Nós ou **Vértices**, unidos por um traço denominado **Aresta,** configurando a relação imaginária, conforme pode ser visto na Figura “Exemplo de Grafos”. Normalmente, os vértices são identificados por letras ou números, no grafo (a), os vértices são **a**, b, c, d e e; já no grafo (b), 1, 2, 3, 4 e 5.

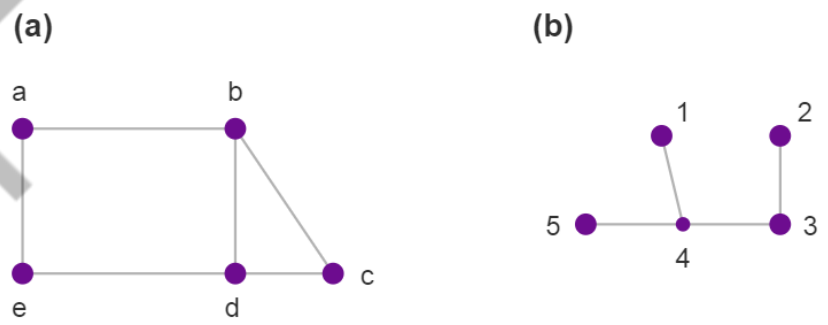


Figura 8.2–Exemplo de Grafos.
Fonte: Elaborado pelo autor (2019).

Uma aresta sempre liga 2 vértices do grafo e é descrita por meio da identificação dos vértices que une. No exemplo de grafo da Figura “Exemplo de Grafos” (a), as arestas são: (a,b), (b,c), (b,d), (c,d), (d,e) e (a,e). Já no grafo (b), as arestas são: (4,1), (4,3), (4,5) e (2,3).

Dois vértices x e y são ditos adjacentes ou vizinhos se existe uma aresta unindo-os. Já duas arestas são adjacentes se elas têm ao menos um vértice em comum.

Os vértices x e y são ditos incidentes na aresta, se eles são extremos da aresta.

Laço é uma aresta que une um par de vértices idênticos. É possível também que entre 2 vértices existam mais uma aresta, as múltiplas arestas entre 2 vértices são chamadas de arestas paralelas. Quando um grafo possui arestas paralelas, ele é chamado de **multigrafo**.

A Figura “Exemplo de Grafo com Laço e Arestas Paralelas” apresenta um exemplo de um grafo com um laço (vértice a), sendo esse grafo classificado como multigrafo por possuir arestas paralelas (existentes entre vértices b,d).

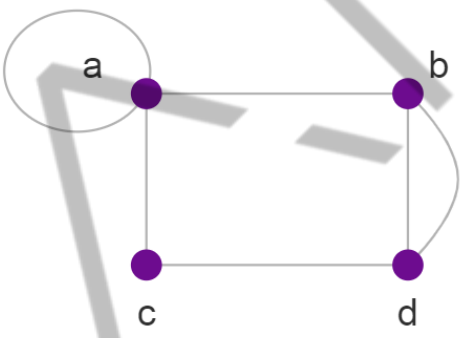


Figura 8.3–Exemplo de Grafo com Laço e Arestas Paralelas.
Fonte: Elaborado pelo autor (2019).

Quando um grafo não possui laços nem arestas paralelas, é chamado de **grafo simples**.

Outras definições importantes são de trajeto e caminho:

- Dado um grafo G(V, A), um **trajeto** em G consiste de uma sequência finita alternada de vértices e arestas, começando e terminando por vértices, tal que cada aresta aparece apenas uma vez e é incidente ao vértice que a precede e ao que a sucede.
- Um **caminho** em um grafo G(A,V) consiste de uma sequência finita alternada de vértices e arestas, começando e terminando por vértices, tal que cada aresta é incidente ao vértice que a precede e ao que a sucede e

não há repetição de vértices. Em outras palavras, um caminho é um trajeto onde não há repetição de vértices.

Observando novamente a Figura “Exemplo de Grafos” (a), temos como exemplo de trajeto a-b-d-c-d ou d-c-b-d-e. Já como exemplos de caminhos a-b-c-d-e ou c-b-d.

Quando usamos a definição de caminho para sairmos de um determinado vértice (origem) para atingirmos outro vértice (destino), podemos definir o comprimento do caminho, que é o número de vértices percorridos. Por exemplo, tendo como origem o vértice a e destino o vértice c, temos três possíveis caminhos: **a-b-c** (comprimento 3), **a-b-d-c** (comprimento 4) e **a-e-d-c** (comprimento 4)

Até aqui, apenas cada aresta tem sido descrita como uma ligação entre 2 vértices, sem uma direção específica. Esse tipo de grafo é dito **grafo não orientado**.

Um **grafo direcionado** D(V,E) (chamados de **dígrafos**) é formado por arestas que ligam vértices em apenas uma direção. A Figura “Exemplo Grafo Direcionado” apresenta um exemplo de um dígrafo.

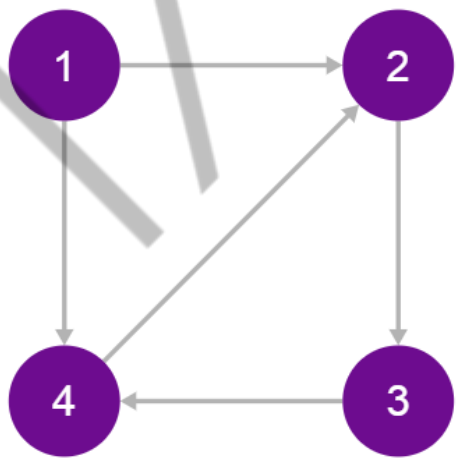


Figura 8.4–Exemplo Grafo Direcionado.
Fonte: Elaborado pelo autor (2019).

O conceito de trajeto e caminho definido anteriormente é válido também para dígrafos, com a diferença que deve ser respeitada a direção das arestas que unem os vértices. Assim, usando o exemplo da Figura “Exemplo Grafo Direcionado”, temos que há apenas 1 caminho para ter como vértice origem 4 e destino o vértice 2; já para vértice origem 1 ao vértice destino 4, temos 2 caminhos possíveis: 1-4 ou 1-2-3-4.

Muitas aplicações de grafos necessitam que cada aresta receba um valor associado, que é chamado de peso. O peso pode ser relacionado à distância entre 2 vértices, custo, tempo de percurso etc. Esse tipo de grafo é denominado de **grafo ponderado** ou valorado.

8.2.1 Grafo para Resolver Problema das Pontes de Königsberg

Depois das definições necessárias já terem sido apresentadas, vamos voltar ao problema das pontes que originou a teoria dos grafos.

O problema consiste em achar um caminho ao longo do qual um pedestre, partindo de uma das margens, ou de qualquer uma das ilhas, percorra todas as pontes, sem passar mais de uma vez por qualquer uma delas.

Para resolver o problema das sete pontes da cidade de Königsberg, Euler fez a observação fundamental que, para efeito da questão proposta, as margens e as ilhas são como se fossem pontos A, B, C e D. As pontes são como arcos que têm esses pontos como extremidades, como apresentado na Figura “Grafo que modela o problema das pontes de Königsberg”.

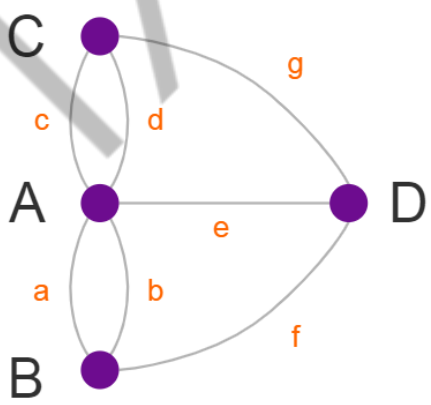


Figura 8.5–Grafo que modela o problema das pontes de Königsberg
Fonte: https://miltonborba.org/Algf/Pontes.htm (2019).

Usando a forma de um grafo para representar um conjunto finito de pontos, chamado de vértices do grafo, e um conjunto finito de arcos, chamado de arestas do grafo, Euler chamou a atenção para uma noção muito simples, porém, crucial, que é

a ordem de um vértice do grafo. A ordem de um vértice é o número de arcos ou arestas que saem dele.

Euler observou que toda vez que um caminho unicursal (percurso Euleriano) chega a um vértice, deve sair dele por um arco diferente daquele por onde chegou (a menos que esse vértice seja o fim do caminho); portanto, para conseguir passar por todas as sete pontes sem passar mais de uma vez pelo mesmo caminho, os vértices desse grafo devem ser todos com arestas par, com exceção dos vértices do início e do fim do caminho. Se o início e o fim do caminho coincidirem (isto é, se o caminho for fechado), então, todos os vértices do grafo, sem exceção, têm ordem par.

Conclui-se, então, que se um grafo é unicursal (Euleriano), ou todos os seus vértices têm ordem par (caminho unicursal fechado) ou exatamente dois vértices têm ordem ímpar (caminho unicursal), deve começar em um vértice de ordem ímpar e terminar em outro.

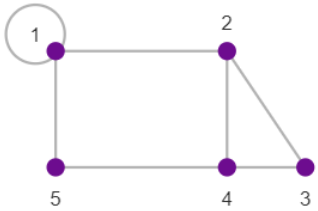
8.3 Representação de Grafos em Computação

Para que grafos possam ser utilizados para a solução de problemas computacionais, é preciso definir como podem ser representados dentro dos nossos programas.

Uma matriz de adjacências M de um grafo com n vértices possui n linhas e n colunas com a seguinte regra de preenchimento:

$$M = \begin{cases} 1, se\ existir\ \ uma\ aresta\ que\ ligue\ i,j \\ ou \\ 0, se\ não\ existir\ \ uma\ aresta\ que\ una\ i,j \end{cases}$$

Suponha como exemplo de grafo não direcionado o grafo apresentado na Figura “Exemplo de grafo e sua matriz de adjacências”. Nesse grafo, a matriz de incidência M tem dimensão 5x5 por haver 5 vértices no grafo (numerados de 1 a 5).

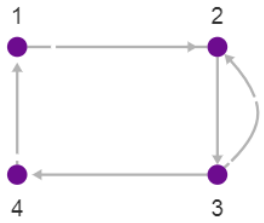


	1	2	3	4	5
1	1	1	0	0	1
2	1	0	1	1	0
3	0	1	1	0	0
4	0	1	1	0	1
5	1	0	0	1	0

Figura 8.6– Exemplo de grafo e sua Matriz de Adjacências
Fonte: Elaborado pelo autor (2019).

Por ser um grafo não direcionado, a matriz de adjacências é simétrica, ou seja, a aresta que une o vértice 2 ao 3 (marcada como 1 no elemento (2,3) da matriz) também une o vértice 3 ao 2 (elemento (3,2). No grafo, existe um laço no vértice 1, essa aresta é representada como 1 na matriz no elemento (1,1).

Em um dígrafo (grafo direcionado), nesse tipo de grafo, a matriz de adjacências não é simétrica uma vez que cada aresta deve apenas ser representada como 1 na matriz na direção especificada no grafo. Um exemplo é apresentado na Figura “Exemplo de dígrafo e sua matriz de adjacências”.



	1	2	3	4
1	0	1	0	0
2	0	0	1	0
3	0	1	0	1
4	1	0	0	0

Figura 8.7– Exemplo de Dígrafo e sua Matriz de Adjacências
Fonte: Elaborado pelo autor (2019).

8.3.2 Lista de Adjacências

Uma forma alternativa de armazenar as arestas adjacentes de cada vértice de um grafo é representando como uma lista ligada. A vantagem de usar essa forma de representação é que quando um grafo possuir um número elevado de vértices a quantidade de memória para guardar cada aresta será bem menor.

A figura a seguir apresenta como usar a lista de adjacências para um grafo não orientado. Note que é criado um vetor com ponteiro (referência) ao primeiro nó da lista que contém a informação de um nó adjacente. No exemplo, o vértice 1 tem como adjacentes os vértices 2, 5 e 1 (laço para ele mesmo). Como o grafo não é orientado, o vértice 2 também tem em sua lista de vértices adjacentes o nó com identificação do vértice 1.

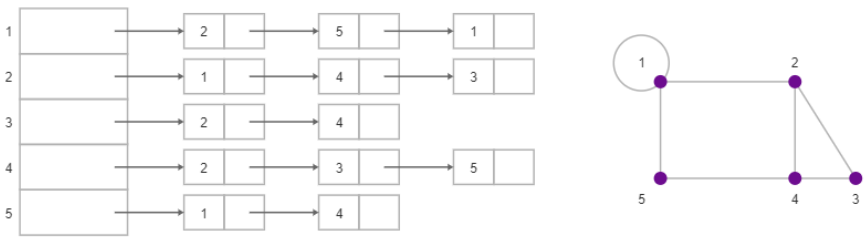


Figura 8.8– Exemplo de Grafo Não Orientado com Lista de Adjacências
Fonte: Elaborado pelo autor (2019).

Quando temos um dígrafo (grafo orientado) como apresentado na Figura “Exemplo de Dígrafo e sua lista de adjacências”, a lista deve apenas conter os nós com a direção descrita no dígrafo. Assim, a partir do vértice 1, temos apenas uma aresta direcionada que o liga com o vértice 2. Já no vértice 3, temos arestas direcionadas com os vértices 2 e 4.

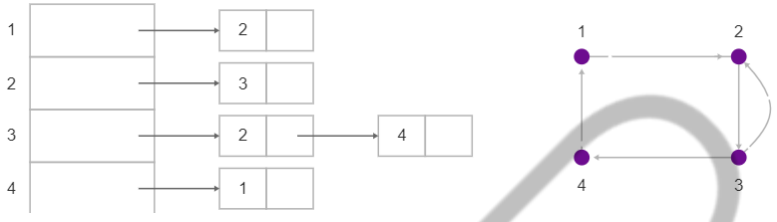


Figura 8.9– Exemplo de Dígrafo e sua Lista de Adjacências
Fonte: Elaborado pelo autor (2019).

Muitas vezes, as aplicações que utilizam grafos precisam definir arestas com pesos que, por exemplo, estabelecem a distância entre 2 vértices. Para esses casos, cada nó da lista de vértices adjacentes precisa armazenar também essa informação adicional. A Figura “Exemplo de grafo ponderado com lista de adjacências” apresenta como um grafo com arestas com pesos é representado em uma lista de adjacências.

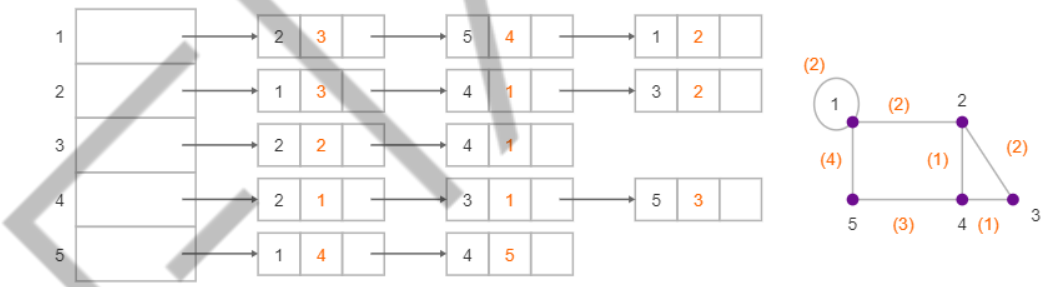


Figura 8.10– Exemplo de Grafo Ponderado com Lista de Adjacências
Fonte: Elaborado pelo autor (2019).

8.4 Busca ou Percurso em Grafos

Há duas formas de realizar busca ou percurso em grafos: percurso em profundidade e em largura. Vamos conhecer qual a diferença entre elas. Em Ascêncio e Araújo (2010), é apresentado como os dois algoritmos podem ser implementados em JAVA.

8.4.1 Percurso em Profundidade

Na teoria dos grafos, busca em profundidade (em inglês por *Depth-First Search* – DFS- busca em profundidade primeiro) é um algoritmo usado para realizar uma busca ou percurso. A ideia desse algoritmo é começar a partir de um vértice de origem e explorar em profundidade cada um dos seus ramos, antes de retroceder (*backtracking*). Assim, se quisermos partir de um ponto A para um ponto B em um grafo, vamos tentar alcançar o ponto B por todos os ramos.

8.4.2 Percurso em Largura

A busca ou percurso em largura (em inglês *Breadth-First Search* - BFS) é um algoritmo de busca em grafos utilizado para realizar uma busca ou percurso em um grafo. De forma intuitiva, o processo de busca começa pelo vértice de origem e explora todos os vértices vizinhos. Então, para cada um desses vértices mais próximos, exploramos os seus vértices vizinhos inexplorados e assim por diante, até que ele encontre o alvo da busca.

Esse algoritmo examina sistematicamente todos os vértices de um grafo e para evitar que nenhum vértice ou aresta será visitado mais de uma vez, o algoritmo faz uso de uma estrutura de dados do tipo fila para garantir a ordem de chegada dos vértices. Dessa maneira, as visitas aos vértices são realizadas pela ordem de chegada na estrutura fila e um vértice que já foi marcado não pode entrar novamente nessa estrutura.

O mecanismo de percurso do grafo usado por esse algoritmo permite que se percorra a menor distância do vértice de origem até o que se deseja atingir. Essa característica do algoritmo permite construir uma árvore de distâncias mínimas (menor número de arestas) entre o vértice-raiz e os demais.

REFERÊNCIAS

ASCÊNCIO, A. F. G.; ARAÚJO, G. S. **Estruturas de dados:** algoritmos, análise de complexidade e implementações em JAVA e C/C++. São Paulo: Pearson Prentice Hall, 2010.

BOAVENTURA NETTO, P. O.; JURKIEWICZ, S. **Grafos:** introdução e prática. 2. ed. Blucher, 2017.

DEITEL, P. J.; DEITEL, H. M. **Java:** como programar. 8. ed. São Paulo: Pearson Prentice Hall, 2010.

TENENBAUM, A. M. et al. **Estruturas de dados usando C.** Makron Books, 1995.

ZIVIANI, N. **Projeto de algoritmos com implementações em Pascal e C.** São Paulo: Pioneira, 2000.

GLOSSÁRIO

Termo	Explicação.
Termo	Explicação.