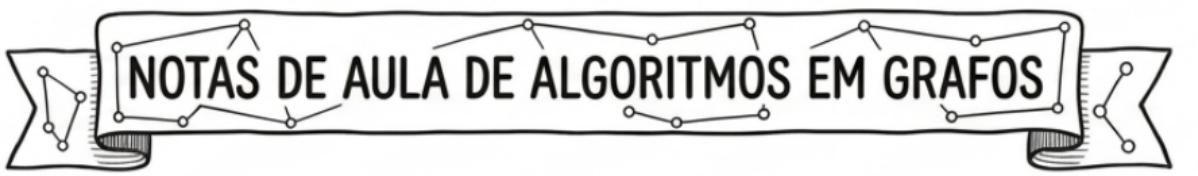


J Donadelli

2026-3

Conteúdo

1 Fundamentos

1.1 Definições iniciais

Seja V um conjunto. Definimos

$$\binom{V}{2} = \big\{\{u, v\} \subseteq V : u \neq v\big\}$$

por exemplo

$$\binom{\{1,2,3\}}{2} = \big\{\{1,2\},\{1,3\},\{2,3\}\big\}.$$

Definição 1.1: grafo

Um ***grafo*** é uma estrutura matemática definida por um par ordenado de conjuntos finitos (V, E) tal que $E \subseteq \binom{V}{2}$.
Os elementos de V são chamados ***vértices***.
Os elementos de E são chamados ***arestas***.

Se G denota o grafo que é definido pelo par (V, E) então escrevemos $G := (V, E)$.

Além disso, se G é um grafo e não especificamos o conjunto dos vértices e o conjunto das arestas que definem G, esses passam a ser referidos como $V(G)$ e $E(G)$, respectivamente.

Exemplo 1.1

Os conjuntos

$$V = \{1,2,3,4,5,6,7,8\}$$
$$E = \big\{\{1,2\},\{1,5\},\{2,5\},\{3,4\},\{6,7\}\big\}$$

definem um grafo.

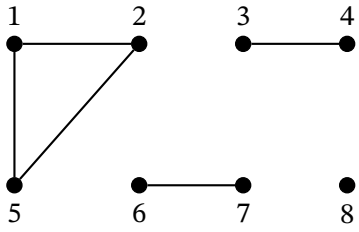


Figura 1.1: diagrama do grafo do Exemplo 1.1.

Todo grafo pode ser representado geometricamente por um ***diagrama***: no plano, desenhamos um ponto para cada vértice e um segmento de curva ligando cada par de vértices que forma uma aresta. Um diagrama de um grafo não é único, mas cada diagrama descreve um único grafo. Um diagrama do grafo do Exemplo 1.1 é apresentado na figura ao lado.

- Os vértices u e v de uma grafo G são ***adjacentes*** se $\{u, v\}$ é uma aresta de G. Também dizemos que u e v são os ***extremos*** da aresta.
- Duas arestas e e f de uma grafo G são ***arestas adjacentes*** se elas têm um vértice em comum, ou seja, $|\, e \cap f \,| = 1$.

- A aresta e ***incide*** no vértice v, em um grafo G, se $v \in e$. A quantidade de arestas de G que incidem em v é o ***grau de*** v em G, $\mathrm{d}_G(v)$:

$$\mathrm{d}_G(v) \coloneqq \Big|\big\{ w \in V(G) : \{v, w\} \in E(G) \big\}\Big|.$$

- A ***vizinhança*** do vértice v no grafo G, denotada $N_G(v)$, é formada pelos outros extremos das arestas que incidem em v, isto é,

$$N_G(v) \coloneqq \{ u \in V(G) : \{u, v\} \in E(G) \}.$$

  Alternativamente, temos $\mathrm{d}_G(v) = |N_G(v)|$.

Teorema 1.1

Para todo grafo G vale que

$$\sum_{u \in V(G)} \mathrm{d}_G(u) = 2|E(G)|. \tag{1.1}$$

Demonstração. Seja $G = (V, E)$ um grafo e defina o conjunto

$$X \coloneqq \{ (u, e) \in V \times E : u \in e \}.$$

Vamos contar seu número de elementos de duas maneiras distintas.

Primeiro, cada vértice $u \in V$ participa de $\mathrm{d}_G(u)$ elementos de X, portanto

$$|X| = \sum_{u \in V} d(u). \tag{1.2}$$

Depois, cada aresta e está presente em dois elementos de X, logo

$$|X| = 2|E|. \tag{1.3}$$

De (1.2) e (1.3) temos (1.1). □

1.2 Subgrafo

Definição 1.2: subgrafo

Dizemos que o grafo H é um ***subgrafo*** do grafo G se, e somente se,

$$V(H) \subseteq V(G) \text{ e } E(H) \subseteq E(G)$$

e, nesse caso, escrevemos $H \subseteq G$ para indicar que H é subgrafo de G.

Exemplo 1.2

Considerando o grafo G do Exemplo 1.1 temos que

$$\begin{aligned} G' &:= (\{1,2,5\},\{\{1,5\},\{5,2\},\{1,2\}\}) \quad \text{e} \\ G'' &:= (\{3,5,6\},\varnothing) \end{aligned}$$

são subgrafos de G, enquanto que

$$\begin{aligned} H &:= (\{1,2,3\},\{\{1,2\},\{3,4\}\}), \\ I &:= (\{1,2,3,4,9\},\{\{1,2\},\{3,4\}\}) \quad \text{e} \\ J &:= (\{1,2,3,4,8\},\{\{1,2\},\{3,4\},\{1,8\}\}) \end{aligned}$$

não são subgrafos de G pois: H não é grafo, em I não vale $V(I) \subseteq V(G)$ e em J não vale $E(J) \subseteq E(G)$.

Dados um grafo G e um subconjunto de vértices $U \subseteq V(G)$, o ***subgrafo induzido por*** U é o subgrafo $G[U]$ dado por

$$\left(U, E(G) \cap \binom{U}{2}\right).$$

Analogamente, definimos subgrafo induzido por um subconjunto de arestas. Se

$$M = \{e_1, e_2, \ldots, e_m\} \subseteq E(G),$$

então o ***subgrafo induzido por*** M é o subgrafo $G[M]$ dado por

$$\left(\bigcup_{i=1}^{m} e_i, M\right).$$

Exemplo 1.3

Dos grafos G, H e I cujos diagramas são dados na Figura 1.2 podemos dizer que H é um subgrafo induzido de G enquanto que I é um subgrafo mas não é induzido.

Um subgrafo $H \subseteq G$ onde $V(H) = V(G)$ é chamado de ***subgrafo gerador***. No exemplo acima o grafo I é subgrafo gerador de G, enquanto que H não é subgrafo gerador de G.

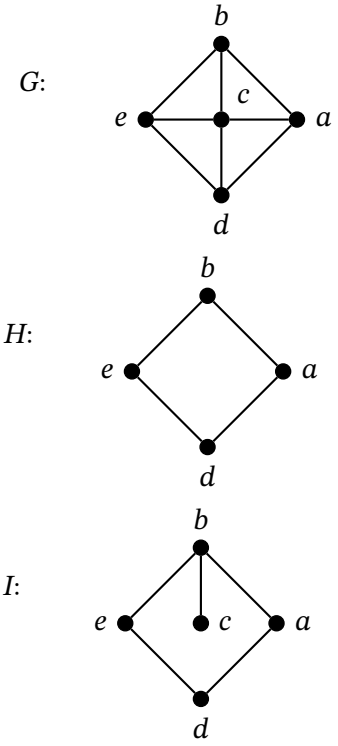


Figura 1.2: diagrama dos grafos G, H e I.

1.3 Passeios, caminhos e circuitos

Uma sequência W de vértices de um grafo G

$$W = v_1, v_2, \ldots, v_k \qquad (k \geq 1) \tag{1.4}$$

é chamada de ***passeio*** em G se v_i e v_{i+1} são adjacentes, para todo $i \in \{1, 2, \ldots, m-1\}$.

Por exemplo, no grafo do Exemplo 1.1 a sequência $1, 2, 5, 2, 1, 5$ define um passeio. Também a sequência $1, 2, 5, 2, 1$ define um passeio que, nesse caso, chamamos de ***passeio fechado***, ou seja, um passeio como W na equação (1.4) acima com $v_1 = v_k$.

No grafo G do Exemplo 1.2 temos o passeio fechado a, c, b, a, c, d, a no qual os vértices c e a ocorrem mais de uma vez. Também o par $\{a, c\}$ é uma aresta *percorrida duas vezes*. Restringido a possibilidade de repetição de vértices e arestas temos tipos especiais de subgrafos, que merecem destaque e, portanto, um nome especial.

Definição 1.3: caminho em G

Um ***caminho*** $P \subseteq G$ é um passeio

$$P = v_1, v_2, \dots, v_k \qquad (k \geq 1)$$

em G tal que $v_i \neq v_j$, sempre que $i \neq j$, ou seja, vértices não são repetidos.

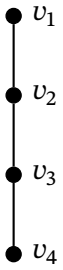


Figura 1.3: caminho.

Denotamos por P^k um caminho com k vértices.

Exemplo 1.4

Para o grafo G representado no diagrama da Figura 1.4, o passeio $P^4 = a, b, f, i$ é um caminho no grafo G. Ainda, esse mesmo caminho é definido pelo subgrafo induzido $G[\{a, b, f, i\}]$.
Também o subgrafo definido pelo par de subconjuntos $\big(\{a, d, c, h, i\}, \big\{\{a, d\}, \{d, c\}, \{c, h\}, \{h, i\}\big\}\big)$ é um caminho em G.
O subgrafo induzido pelo conjunto de vértices $\{m\}$ é um caminho P^1 e o subgrafo induzido pela aresta $\{j, l\}$ é um caminho P^2.

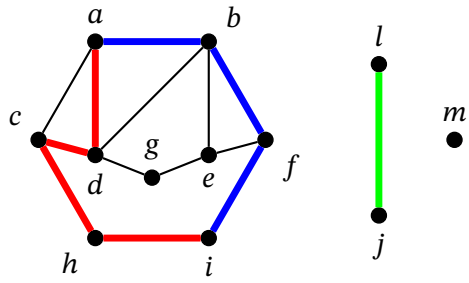


Figura 1.4: exemplos de caminhos em um grafo.

Num caminho $v_1, \dots, v_k$, dizemos que os vértices v_1 e v_k são os seus ***extremos*** e dizemos que os vértices v_i, $1 < i < k$, são os seus ***vértices internos***.

Definição 1.4: circuito em G

Um ***circuito*** $C \subseteq G$ é um passeio

$$C = v_1, v_2, \dots, v_k, v_1 \qquad (k \geq 3)$$

em G tal que $v_i \neq v_j$, sempre que $i \neq j$, ou seja, vértices não são repetidos exceto nos extremos do passeio.

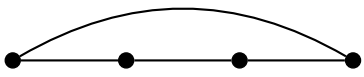


Figura 1.5: um circuito.

Exemplo 1.5: o grafo caminho e o grafo circuito

O grafo (V, E) dado por

$$V = \{1, \dots, k\}$$
$$E = \big\{\{i, i+1\} : 1 \leq i \leq k-1\big\}$$

para algum $k \geq 1$, é um caminho com k vértices (e $k-1$ arestas).

Analogamente, o grafo (V, E) dado por

$$V = \{1, \dots, k\}$$
$$E = \big\{\{i, i+1\} : 1 \leq i \leq k-1\big\} \cup \big\{\{k, 1\}\big\},$$

para algum $k \geq 3$, é um circuito com k vértices (e k arestas).

Um circuito com quatro vértices é representado pelo diagrama da Figura 1.5.

O número de arestas (e, portanto, o número de vértices) em um circuito é o ***comprimento*** desse circuito e denotamos um circuito de comprimento k por C^k. Por exemplo, o grafo G definido a partir da Figura 1.4; nele temos que $C^4 = G[\{b, d, e, g\}]$ é um circuito, bem como o subgrafo $C^5 = (V, E)$ dado por $V = \{a, b, d, e, g\}$ e $E = \{\{a, b\}, \{b, e\}, \{e, g\}, \{g, d\}, \{d, a\}\}$. Também é um circuito o subgrafo induzido pelas arestas cuja cor seja ou azul ou vermelha.

Denotamos por $\delta(G)$ o menor grau de um vértice de G, isto é,

$$\delta(G) = \min_{v \in V(G)} \mathrm{d}_G(v)$$

chamado ***grau mínimo*** de G.

Proposição 1.1

Todo grafo G contém um caminho de comprimento pelo menos $\delta(G)$ e, se $\delta(G) \geq 2$, contém um circuito de comprimento pelo menos $\delta(G) + 1$.

Demonstração. Dado um grafo G, vamos exibir um caminho com pelo menos $\delta(G)$ arestas e um circuito com pelo menos $\delta(G) + 1$ arestas nos casos em que $\delta(G) \geq 2$.

Seja $P = x_0, x_1, \dots, x_k$ um caminho de comprimento máximo em G. De P ter comprimento máximo em G temos que $N_G(x_0) \subsetneq V(P)$, caso

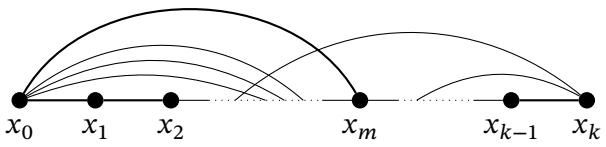


Figura 1.6: se $P = x_0, \dots, x_k$ é de comprimento máximo em G então seus extremos devem ter a vizinhança em $V(P)$.

contrário, haveria caminho mais longo que P em G. Logo, $|N_G(x_0)| \leq |V(P)| - 1$ e, como $|N_G(x_0)| \geq \delta(G)$ e $|E(P)| = |V(P)| - 1$, segue que $|E(P)| \geq \delta(G)$, isto é, P é um caminho de comprimento pelo menos $\delta(G)$.

Agora, tratando o caso de circuito, vamos supor que $\delta(G) \geq 2$. Se o grau mínimo é pelo menos dois, então o vértice x_0 tem um vizinho em P diferente de x_1 e assim está bem definido o número

$$m := \max\{\, j \in \{2, \dots, k\} : \; x_j \in N_G(x_0) \,\},$$

que é o maior índice em $\{2, \dots, k\}$ de um vértice vizinho de x_0 em P.

Basta notarmos que o circuito $x_0, x_1, \dots, x_m, x_0$ tem comprimento $m+1$ e que $N_G(x_0) \subseteq \{x_1, x_2, \dots, x_m\}$, logo $m \geq |N_G(x_0)|$, portanto,

$$m \geq |N_G(x_0)| \geq \delta(G).$$

Portanto, $m + 1 \geq \delta + 1$, ou seja, o circuito C tem comprimento pelo menos $\delta(G) + 1$. □

Distância

Uma consequência importante da definição de caminho num grafo é que, com ela, temos uma definição de distância entre vértices do grafo, definida pelo caminho mais curto. Vamos formalizar essa definição.

O ***comprimento*** de um caminho P é número de arestas no caminho. Um caminho com n vértices tem $n-1$ arestas, portanto, comprimento $n-1$.

Definição 1.5: distância em G

Em um grafo $G = (V, E)$, a ***distância*** entre dois vértices quaisquer $u, v \in V$, denotada por $\operatorname{dist}_G(u, v)$, é definida como o comprimento do menor caminho P com extremos u e v, quando existe algum caminho

$$\operatorname{dist}_G(u, v) = \min_{k \in \mathbb{Z}_+} \{\, G \supseteq P^k \text{ caminho com extremos } u, v \,\}.$$

Se u e v não são extremos de algum caminho em G, então convencionamos

$$\operatorname{dist}_G(u, v) = \infty.$$

Convencionamos, como é usual, que ∞ satisfaz

$$n + \infty = \infty \quad \text{e} \quad n < \infty,$$

para todo $n \in \mathbb{N}$.

No Exemplo 1.4, temos $\operatorname{dist}_G(a, j) = \infty$, $\operatorname{dist}_G(a, b) = 1$, $\operatorname{dist}_G(a, i) = 3$, $\operatorname{dist}_G(m, j) = \infty$.

Definição 1.6: grafo conexo

Um grafo G é ***conexo*** se para todos $u, v \in V(G)$ vale que

$$\operatorname{dist}_G(u, v) < \infty.$$

Dados vértices $u, v \in V(G)$, dizemos que v ***alcança*** u em G se, e somente se, existe um caminho com extremos u e v em G. Assim, "alcança" é uma relação de equivalência em $V(G)$.

As classes de equivalência dessa relação são os subconjuntos dos vértices alcançáveis entre si, chamadas de ***componentes conexas*** do grafo. Equivalentemente, um grafo é conexo se e somente se tem uma única classe de equivalência pela relação "alcança".

Proposição 1.2: $(V(G), \operatorname{dist}_G)$ é espaço métrico

A função $\operatorname{dist}_G : V(G) \times V(G) \to \mathbb{N}$ é uma ***métrica*** em $V(G)$ pois satisfaz

1 $\operatorname{dist}_G(u, v) = \operatorname{dist}_G(v, u) \geq 0$ para todos $u, v \in V(G)$;
2 $\operatorname{dist}_G(v, u) = 0$ se e somente se $u = v$;
3 a ***desigualdade triangular***

$$\operatorname{dist}_G(u, w) \leq \operatorname{dist}_G(u, v) + \operatorname{dist}_G(v, w) \tag{1.6}$$

para quaisquer $u, v, w \in V(G)$.

1.4 Outras noções de grafos

Em algumas situações precisamos de um modelo para um problema a ser resolvido e esse modelo seria um grafo se desconsiderássemos algumas peculiaridades da situação. Por exemplo, um mapa rodoviário pode ser modelado definindo-se um vértice para cada cidade e duas cidades formam uma aresta no grafo (modelo) se existe rodovia ligando as cidades correspondentes aos vértices. Normalmente, distância é um parâmetro importante nesses mapas e assim as arestas devem ter um comprimento associado a elas, entretanto, "comprimento de aresta" não faz parte da definição de um grafo. Num outro exemplo, se estamos interessados em rotas de tráfego dentro de uma cidade podemos definir um vértice por esquina e duas esquinas consecutivas numa mesma rua formam uma aresta. Nesse caso, as ruas têm sentido (mão e contramão) e as arestas também deveriam ter mas, novamente, essa característica não faz parte da definição de grafos. Esses problemas e muitos outros podem ser modelados com "outros tipos" de grafos. Alguns desses outros tipos são

- ***grafo com pesos nas arestas*** é definido por um tripla (V, E, ρ) de modo que (V, E) é um grafo e $\rho : E \to R$ é uma função que assume valores em R; usualmente $R \subseteq \mathbb{R}$;

- ***grafo orientado*** têm orientação nas arestas, de modo que as arestas, ou ***arcos***, agora são pares ordenados. Um grafo orientado fica definido por um par (V, E) com $E \subseteq V \times V$ sendo uma relação irreflexiva e assimétrica;
- ***grafo dirigido ou digrafo*** é dado por um par (V, E) onde $E \subseteq V \times V \setminus \{(v, v)\, v \in V\}$;
- ***multigrafo*** é dado por um conjunto V de multigrafo vértices, um conjunto E de arestas e uma função de E em $\binom{V}{2}$ que diz quem são os extremos de cada aresta; nesse caso podemos ter mais de uma aresta com os mesmos extremos.

Em cada um dos casos acima, podemos tomar o grafo naturalmente definido por essas estruturas, chamado ***grafo subjacente***. Outros tipos podem ser derivados combinando esses tipos de grafos, por exemplo, grafo dirigido com pesos nas arestas. Um exemplo de diagrama de grafo orientado com pesos nas arestas é dado na figura abaixo. Usamos

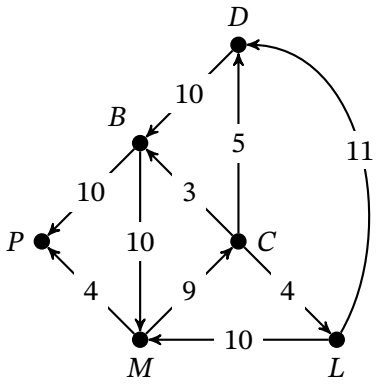


Figura 1.7: exemplo de grafo orientado.

para representar a aresta orientada (a, c).

Exercícios

Exercício 1. *Demonstre que em todo grafo o número de vértices com grau ímpar é par.*

Exercício 2. *Um químico deseja embarcar os produtos P, Q, R, S, T, O, X usando o menor número de caixas. Alguns produtos não podem ser colocados numa mesma caixa porque reagem. Os produtos P, Q, R, X reagem entre si; P reage com O e com S e vice-versa; T também reage com O e com S e vice-versa. Descreva o grafo que modela essa situação, mostre um diagrama desse grafo e use-o para descobrir o menor número de caixas necessárias para embarcar os produtos com segurança.*

Exercício 3. *Adaltina esperava as amigas Brandelina, Clodina, Dejaina e Edina para um lanche em sua casa. Enquanto esperava preparou os lanches: Bauru, Misto quente, Misto frio e X-salada. Brandelina gosta de Misto frio e de X-salada; Clodina de Bauru e X-salada; Dejaina gosta*

de Misto quente e Misto frio; Edina gosta de de Bauru e Misto quente. Descreva o grafo que modela essa situação, mostre um diagrama desse grafo e use-o para descobrir se é possível que cada amiga de Adaltina tenha o lanche que gosta. Se possível, determine o número de soluções.

Exercício 4. *Defina todos os grafos sobre o conjunto de vértices* $\{1,2,3,4\}$ *e tamanho* 6.

Exercício 5. *Para todo* $n \in \mathbb{N}$*, qual é o número máximo de arestas que pode ter um grafo com* n *vértices?*

Exercício 6. *O* **complemento** *de um grafo* G*, denotado por* $G^{\complement}$*, é o grafo que tem o mesmo conjunto de vértices de* G *e dois vértices formam uma aresta em* $G^{\complement}$ *se e somente se* **não** *formam uma aresta de* G*:*

$$\begin{aligned} V(G^{\complement}) &= V(G), \\ E(G^{\complement}) &= \binom{V(G)}{2} \setminus E(G) = \big\{\{u,v\} \subset V(G) : \{u,v\} \notin E(G)\big\}. \end{aligned}$$

Dê o complemento dos seguintes grafos

(i) G *dado por*

$$\begin{aligned} V(G) &= \{1,2,3,4,5\}, \\ E(G) &= \{\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{2,3\},\{2,4\},\{2,5\},\{3,4\},\{3,5\},\{4,5\}\}. \end{aligned}$$

(ii) H *dado por*

$$\begin{aligned} V(H) &= \{1,2,3,4,5,6,7\}, \\ E(H) &= \{\{1,2\},\{1,3\},\{2,4\},\{2,5\},\{3,6\},\{3,7\}\}. \end{aligned}$$

(iii) B *dado por*

$$\begin{aligned} V(B) &= \{a,b,c,1,2,3\}, \\ E(B) &= \{\{a,1\},\{a,2\},\{a,3\},\{b,1\},\{b,2\},\{b,3\},\{c,1\},\{c,2\},\{c,3\}\}. \end{aligned}$$

Exercício 7. *Se definirmos a* **união** *dos grafos* G *e* H *por* $\mathbf{G \cup H} := \big(V(G) \cup V(H), E(G) \cup E(H)\big)$*, a união* $G \cup H$ *é um grafo?*

Exercício 8. *Se definirmos* **intersecção** *dos grafos* G *e* H *por* $\mathbf{G \cap H} := \big(V(G) \cap V(H), E(G) \cap E(H)\big)$*, a intersecção* $G \cap H$ *é um grafo?*

Exercício 9. *Um grafo é chamado de* **completo** *sobre* V *se todo par de vértices de* V *é uma aresta do grafo, ou seja* $E = \binom{V}{2}$*. Um grafo completo com* n *vértices é denotado por* K^n*. Prove que para qualquer grafo* G *vale que* $G \cup G^{\complement} = K^n$*.*

Exercício 10. *Considere o caso geral do exercício 2: Um químico deseja embarcar os produtos* $p_1, p_2, \ldots, p_n$ *usando o menor número de caixas. Alguns produtos não podem ser colocados numa mesma caixa porque reagem e o químico conhece uma listagem com* m *pares de produtos que*

reagem. Seja G o grafo que modela esse problema, onde vértices são produtos e arestas os pares que reagem. Denotemos por $\chi(G)$ o número de mínimo de caixas de modo que seja possível encaixotar os produtos com segurança. Prove que

$$\chi(G) \leq \frac{1}{2} + \sqrt{2m + \frac{1}{4}}.$$

(Dica: em uma distribuição mínima de caixas, a cada duas caixas, precisa existir pelo menos um produto em uma caixa reagindo com um produto da outra caixa. Assim podemos garantir um número mínimo m de arestas para o grafo.)

Exercício 11. *Vimos que $\delta(G)$ denota o grau mínimo de G. Analogamente, definimos $\Delta(G)$, o* **grau máximo** *de G, por*

$$\Delta(G) = \max_{v \in V(G)} \mathrm{d}_G(v)$$

e definimos $\mathrm{d}(G)$, o **grau médio** *de G, por*

$$\mathrm{d}(G) = \frac{1}{|V(G)|} \sum_{v \in V(G)} \mathrm{d}_G(v)$$

para todo G com pelo menos 1 vértice, caso contrário $\mathrm{d}(G) := 0$.

Demonstre que $\delta(G) \leq \mathrm{d}(G) \leq \Delta(G)$ para todo grafo G.

Exercício 12. *Decida se pode existir um grafo G com vértices que têm graus $2, 3, 3, 4, 4, 5$, respectivamente. E graus $2, 3, 4, 4, 5$? Se sim, descreva-os.*

Exercício 13. *Seja G um grafo com 14 vértices e 25 arestas. Se todo vértice de G tem grau 3 ou 5, quantos vértices de grau 3 o grafo G possui?*

Exercício 14. *Chico e sua esposa foram a uma festa com três outros casais. No encontro deles houveram vários apertos de mão. Ninguém apertou a própria mão ou a mão da(o) esposa(o), e ninguém apertou a mão da mesma pessoa mais que uma vez. Após os cumprimentos Chico perguntou para todos, inclusive para a esposa, quantas mãos cada um apertou e recebeu de cada pessoa uma resposta diferente. Quantas mãos Chico apertou?*

Exercício 15. *Prove que existem pelo menos dois vértices com o mesmo grau em todo grafo de ordem maior ou igual a dois.*

Exercício 16. *Para um número natural r, um grafo é r-***regular** *se todos os vértices têm grau r. Para um grafo r-regular com n vértices e m arestas, expresse m em função de n e r. Dê exemplo de um grafo 3-regular que não é completo.*

Exercício 17. *Demonstre que em todo grafo G com $\delta(G) > 0$ e $|E(G)| < |V(G)|$ existem pelo menos dois vértices de grau 1.*

Exercício 18. *Quantos subgrafos tem o grafo* $\big(\{1,2,3,4,5,6\},\{\{1,2\}\}\big)$*?*

Exercício 19. *Quantos subgrafos completos tem o grafo completo de ordem* n*?*

Exercício 20. *Sejam* G *um grafo e* $M \subseteq E(G)$*. Tome o subconjunto*

$$U = \bigcup_{e \in M} e$$

de vértices de G*. Prove ou dê um contraexemplo para* $G[U] = G[M]$*.*

Exercício 21. *Descubra um subgrafo induzido de*

$$\begin{aligned} V(G) &= \{1,2,3,4,5,6,7,8\} \text{ e} \\ E(G) &= \big\{\{1,2\},\{1,3\},\{2,3\},\{2,5\},\{3,6\},\{8,5\},\{8,6\},\{5,6\},\{3,4\},\{5,7\}\big\} \end{aligned}$$

que seja 1*-regular e tenha o maior número possível de arestas. (Qual a relação com a resolução do exercício 3?)*

Exercício 22. *Mostre que em qualquer coloração das arestas do grafo* K^6 *com as cores preta e branca vale: ou* K^6 *contém um* k^3 *com todas as arestas pretas ou* K^6 *contém um* k^3 *com todas as arestas brancas.*

Exercício 23. *Dados um grafo* G *e* $U \subseteq V(G)$*, dizemos que* U *é um* **conjunto independente***, ou* **conjunto estável***, em* G *se* $E\big(G[U]\big) = \emptyset$*, ou seja, não há par de vértices adjacentes ambos pertencentes a* U*.*

Denotamos por $\alpha(G)$ *a cardinalidade do maior conjunto independente em* G*,*

$$\alpha(G) = \max\big\{\,|A| : A \subset V(G) \text{ é um conjunto independente}\,\big\}.$$

Demonstre que se $d(G) > \alpha(G)$ *então* G *contém triângulo.*

Exercício 24. *Dados um grafo* G *e* $U \subseteq V(G)$*, dizemos que* U *é um* **clique** *em* G *se* $G[U]$ *é completo.*

Denotamos por $\omega(G)$ *a cardinalidade do maior clique em* G

$$\omega(G) = \max\big\{\,|A| : A \subset V(G) \text{ é um clique}\,\big\}.$$

Demonstre que $\omega(G) = \alpha(G^{\complement})$*.*

Exercício 25. *Demonstre que as desigualdades abaixo valem para todo grafo* G

(i) $\alpha(G) \geq |V(G)|/(\Delta(G)+1)$*;*
(ii) $\alpha(G) \leq |E(G)|/\delta(G)$*, se* $\delta(G) \neq 0$*;*
(iii) $\omega(G) \leq \Delta(G)+1$*.*

Exercício 26. *Suponha* $H \subseteq G$*. Prove ou refute as desigualdades:*

(i) $\alpha(H) \leq \alpha(G)$;
(ii) $\alpha(G) \leq \alpha(H)$;
(iii) $\omega(G) \leq \omega(H)$;
(iv) $\omega(H) \leq \omega(G)$.

Exercício 27. *É verdade que todo grafo com pelo menos três vértices, exatamente dois vértices de grau* 1 *e os demais vértices com grau* 2 *é um caminho?*

Exercício 28. *Escreva um algoritmo que recebe uma grafo G e decide se G é um caminho. Determine a complexidade do algoritmo.*

Exercício 29. *Sejam $P = u_1, u_2, \dots, u_k$ e $Q = v_1, v_2, \dots, v_\ell$ duas sequências de vértices de um grafo. Definimos a* **concatenação** *dessas sequências como a sequência dos vértices de P seguidos dos vértices de Q, denotada $P, Q = u_1, u_2, \dots, u_k, v_1, v_2, \dots, v_\ell$. Supondo que P e Q sejam caminhos no grafo, sob que condições a sequência P, Q é caminho?*

Exercício 30. *Prove que se $u_1, u_2, \dots, u_k$ é um passeio em G então existem índices $i_1, i_2, \dots, i_t$ tais que a subsequência $u_1, u_{i_1}, u_{i_2}, \dots, u_{i_t}, u_k$ é um caminho em G. Também, prove que se $u_1, v_2, \dots, v_{k-1}, u_k$ é um passeio distinto do anterior, então G contém circuito.*

Exercício 31. *Prove a desigualdade triangular (equação (1.6)).*

Exercício 32. *Mostre que a relação $e \sim f$ em $E(G)$ dada por $e = f$ ou existe um circuito $C \subset G$ tal que e e f pertencem a $E(C)$ é uma relação de equivalência.*

Exercício 33. *Seja k um número natural. O* ***k*-cubo** *é o grafo cujo conjunto de vértices são as sequências binárias de k bits e dois vértices são adjacentes se e somente se as k-tuplas correspondentes diferem exatamente em uma posição. Determine o número de vértices e de arestas do k-cubo. Determine os parâmetros α (conjunto independente máximo) e ω (clique máximo) do k-cubo.*

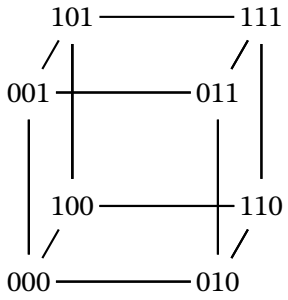


Figura 1.8: um diagrama do 3-cubo.

Exercício 34. *Definimos o* **diâmetro** *do grafo G como a maior distância entre dois vértices quaisquer de G, ou seja*

$$\operatorname{diam}(G) = \max_{u,v \in V(G)} \operatorname{dist}_G(u, v).$$

Naturalmente, $\operatorname{diam}(G) = \infty$ *se existirem dois vértices no grafo* G *que não são extremos de um caminho.*

Determine o diâmetro do k*-cubo.*

Exercício 35. *Definimos a* **cintura** *do grafo* G *como o comprimento do menor circuito contido em* G*, ou seja*

$$\operatorname{cin}(G) = \min\{|E(C)| : \text{ existe um circuito } C \subseteq G\},$$

quando existe um circuito, caso contrário convencionamos que $\operatorname{cin}(G) = \infty$*.*

Determine a cintura do k*-cubo.*

Exercício 36. *Seja* k *um número natural. O* **grafo de Fibonacci** F_k *é o grafo cujo conjunto de vértices são as sequências binárias de tamanho* k *sem ocorrências de 1 consecutivos; dois vértices são adjacentes se e somente se as* k*-tuplas correspondentes diferem exatamente em uma posição.*

Desenhe um diagrama do F_3*.*

Determine o número de vértices, arestas, o diâmetro e a cintura do F_k*. Determine os parâmetros* α *(conjunto independente máximo) e* ω *(clique máximo) do grafo* F_k*.*

Exercício 37. *Seja* G *um grafo conexo. Mostre que se existem* $w \in V(G)$ *e* $\{u, v\} \in E(G)$ *tais que* $\operatorname{dist}(w, u) = \operatorname{dist}(w, v)$ *então* $\operatorname{cin}(G) \leq \operatorname{dist}(w, u) + \operatorname{dist}(w, v) + 1$*.*

Exercício 38. *Seja* p *um número primo e tome* $V = \{0, 1, \ldots, p-1\}$*. Defina o grafo 3-regular* $G = (A \cup B, E)$ *com* $A = V \times \{a\}$ *e* $B = V \times \{b\}$*, ou seja, o que queremos é que* A *e* B *sejam duas cópias disjuntas de* V*.*

Cada vértice $(x, a) \in A$ *é adjacente aos vértices* (x, b)*,* $(x + 1, b)$ *e* $(x + y, b)$ *de* B *onde* $y \neq 0, 1$ *é fixo e a soma é feita módulo* p*.*

Demonstre que qualquer subgrafo induzido de G *de ordem* $p + 1$ *contém um caminho de comprimento 2 quando* $y = 2, (p + 1)/2, p - 1$*. O mesmo vale para qualquer* $y \neq 0, 1$ *de* V*?*

Representação 2

2.1 Isomorfismo

Dois grafos são isomorfos quando têm exatamente a mesma estrutura de conexões, embora seus vértices possam estar nomeados de maneiras diferentes. Ou seja, quando há uma relação 1-1 entre seus vértices que preserva a relação de adjacência. A definição forma é a seguinte.

Definição 2.1: grafos isomorfos

Dizemos que os grafos G e H são ***isomorfos*** se existe uma função bijetora

$$f : V(G) \to V(H) \tag{2.1}$$

tal que

$$\{u, v\} \in E(G) \text{ se, e somente se, } \{f(u), f(v)\} \in E(H)$$

para todos $u, v \in V(G)$. Nesse caso, escrevemos $G \simeq H$.

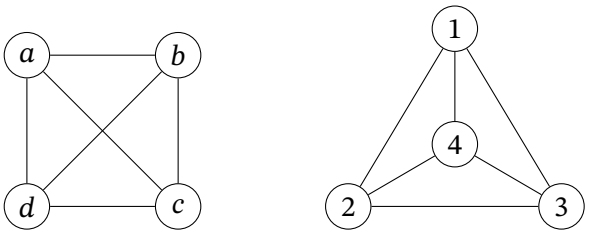
Grafos isomorfos diferem apenas na maneira como sua estrutura é representada, não na estrutura propriamente dita.

Uma função f como (2.1) acima é chamada de ***isomorfismo*** e, por abuso de notação, as vezes escrevemos

$$f : G \to H$$

para um isomorfismo como definido acima.

Por exemplo, são (trivialmente) isomorfos os grafos definidos pelas representações:



e um isomorfismo é a função f dada por

$$f(a) = 1,\ f(b) = 2,\ f(c) = 3 \text{ e } f(d) = 4.$$

Observe que há outros isomorfismos possíveis.

Notemos que quaisquer dois grafos completos G e H com n vértices são isomorfos. Mais que isso, entre grafos completos, qualquer uma das $n!$ bijeções entre $V(G)$ e $V(H)$ define um isomorfismo entre eles.

Nesse caso, dizemos que o grafo completo é ***único a menos de isomorfismos***. Ingenuamente, isso quer dizer que o grafo é único a

menos dos nomes dados aos vértices. Esse fato justifica usarmos a mesma notação para todos eles: K^n.

Exemplo 2.1

Há oito grafos distintos sobre o conjunto de vértices $\{1, 2, 3\}$, eles estão descritos nas representações da Figura 2.1 abaixo.

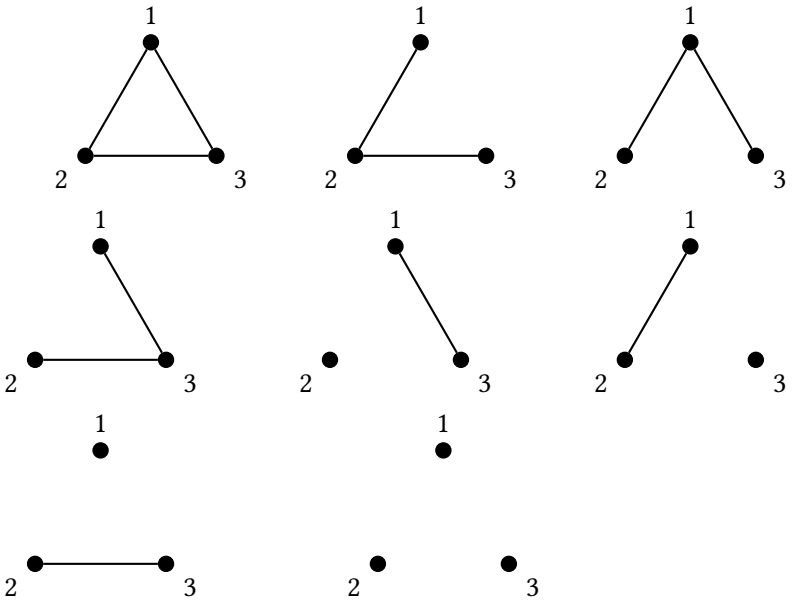


Figura 2.1: grafos distintos de ordem 3.

Exemplo 2.2

Dentre os 8 grafos distintos do exemplo anterior, há apenas 4 grafos não-isomorfos com três vértices, representados pelos diagramas da Figura 2.2. Nesse caso dizemos que ***a menos de isomorfismos*** há 4 grafos de ordem 3.

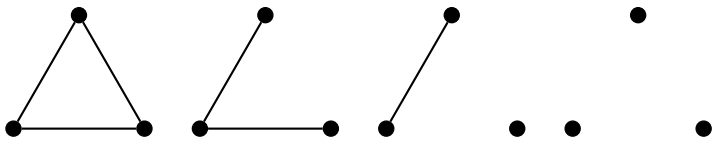


Figura 2.2: os "tipos de grafos" não-isomorfos de ordem 3.

Exemplo 2.3: Grafo de Petersen

Os grafos representados na Figura 2.3 são isomorfos pelo isomorfismo na Tabela 2.1. Qualquer grafo isomorfo a eles é chamado de ***Grafo de Petersen***, é um dos grafos mais conhecidos na teoria dos grafos. Isso se deve ao fato dele ser o menor exemplo ou o menor contraexemplo para muitos problemas em grafos. Júlio Petersen (1839–1910) foi um matemático dinamarquês que por volta de 1898 construiu o grafo que leva seu nome como o menor contraexemplo para afirmação: ***todo***

grafo conexo sem ponte admite uma 3-coloração própria das arestas. Voltaremos a essa afirmação quando tivermos conhecimento dos termos técnicos que ela contém. O grafo de Petersen é retratado nas capas de vários periódicos científicos e livros sobre teoria dos grafos e matemática discreta.

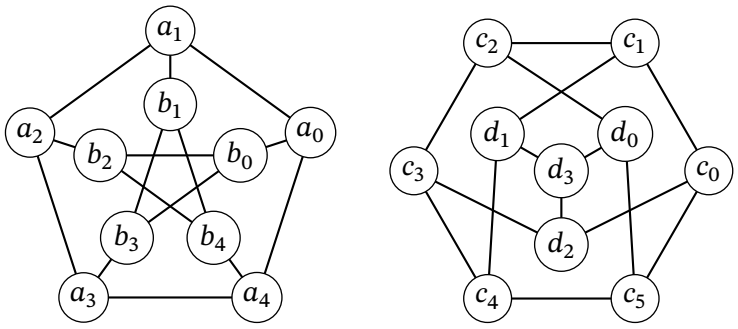


Figura 2.3: grafos isomorfos (grafo de Petersen).

v	$f(v)$	v	$f(v)$
a_0	c_3	b_0	c_4
a_1	d_2	b_1	d_3
a_2	c_0	b_2	c_5
a_3	c_1	b_3	d_1
a_4	c_2	b_4	d_0

Tabela 2.1: isomorfismo entre os grafos da Figura 2.3.

Não isomorfismo

Nenhum par dentre os grafos G, H e K representados na Figura 2.4 são isomorfos.

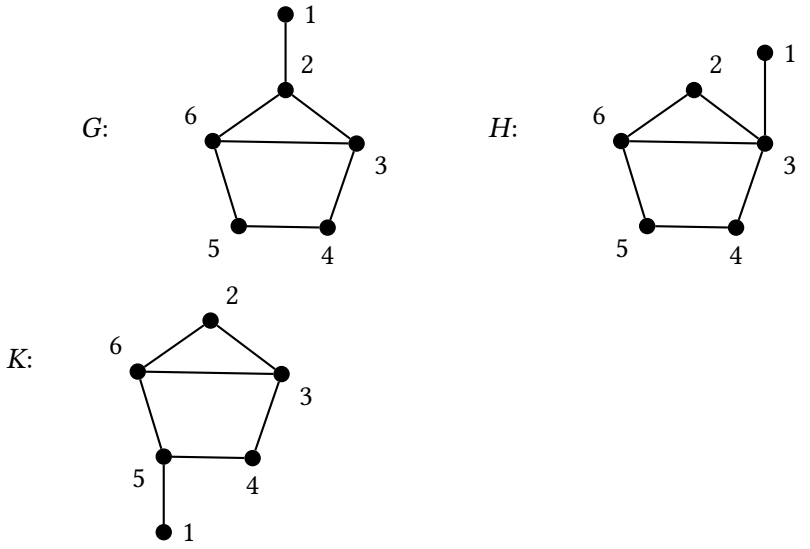


Figura 2.4: grafos não-isomorfos.

Temos que G não é isomorfo a H porque G não tem um vértice de grau quatro enquanto que o vértice 5 em H tem grau quatro, portanto não há como haver uma bijeção entre os vértices desse grafo que preserve as adjacências.

Pelo mesmo motivo H não é isomorfo a K.

Agora, G não é isomorfo a K porque caso existisse um isomorfismo $f : G \to K$ então $f(1) = 4$ e $f(2) = 5$, portanto, a imagem por f do conjunto $\{2, 3, 6\} \subset V(G)$ é, obrigatoriamente, o conjunto $\{4, 5, 6\} \subset V(K)$ o que é contradição pois $\{2, 3, 5\}$ em G induz um triângulo e $f(\{2, 3, 5\})$ em K não induz triângulo (veja o Exercício 43 abaixo).

2.2 Representações computacionais

Em problemas computacionais envolvendo grafos, é conveniente supor que o conjunto de vértices é

$$V = [n] = \{1, 2, \ldots, n\}.$$

Essa convenção não restringe a classe de grafos que podemos considerar, pois a escolha dos nomes dos vértices é irrelevante quando estamos interessados em propriedades que dependem apenas da estrutura do grafo.

Essa observação é especialmente importante do ponto de vista computacional. Uma vez que estamos interessados em propriedades invariantes por isomorfismo, podemos substituir qualquer grafo por um representante isomorfo cujos vértices sejam numerados de 1 a n. Com isso facilitamos o acesso às informações usando estruturas de dados indexadas disponíveis nas linguagens de programação estruturadas.

Dado um grafo $G = (V, E)$, podemos substituir sua representação por uma representação isomorfa na qual os vértices são inteiros. Mais precisamente, se $|V| = n$, escolhemos uma bijeção $f : V \to [n]$ e definimos o grafo $G' = ([n], E')$, onde

$$E' = \big\{\{f(u), f(v)\} : \{u, v\} \in E\big\}.$$

A função f é, por construção, um isomorfismo entre G e G'.

Para os problemas computacionais considerados nesta disciplina, essa substituição não altera o problema: as propriedades de interesse são invariantes por isomorfismo. Portanto, se G é a entrada de um problema, podemos primeiro construir sua cópia isomorfa G' e executar o algoritmo sobre G'. Uma vez obtida a solução para G', podemos transportá-la de volta para G usando o isomorfismo f^{-1}.

Por exemplo, se o algoritmo produz um caminho $(v_1, v_2, \ldots, v_k)$ em G', então $(f^{-1}(v_1), f^{-1}(v_2), \ldots, f^{-1}(v_k))$ é o caminho correspondente em G. O mesmo princípio se aplica a árvores, ciclos, conjuntos de arestas, fluxos e outras estruturas produzidas pelos algoritmos.

Na implementação, a bijeção f e sua inversa podem ser mantidas por estruturas de dados que associem os rótulos originais aos inteiros $1, \ldots, n$. Por exemplo, uma tabela de espalhamento (*hash table*) permite, em geral, realizar essas consultas em tempo esperado constante, enquanto uma árvore binária de busca permite realizá-las em tempo logarítmico quando balanceada. Não trataremos dos detalhes dessa conversão.

Essa convenção é apenas uma escolha de representação. Ela não restringe os grafos considerados, pois todo grafo com n vértices é isomorfo a um grafo com esse conjunto de vértices.

No que segue, sempre que tratamos problemas computacionais so-

bre grafos estamos assumindo os grafos são definidos sobre o conjunto de vértices $\{1, 2, \ldots, n\}$, para algum $n \in \mathbb{N}$.

2.2.1 Matriz de adjacências de G

A matriz de adjacências de $G = (V, E)$ é a matriz $|V| \times |V|$ denotada por A_G, ou A simplesmente, definida por

$$A(i, j) = \begin{cases} 1, & \text{se } \{i, j\} \in E(G) \\ 0, & \text{caso contrário.} \end{cases}$$

Para o grafo G representado na Figura 2.4, a matriz de adjacências é

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

2.2.2 Listas de adjacências de G

Outro modo menos óbvio que usa o conceito de lista ligada, ou lista encadeada. Uma lista ligada é uma coleção de elementos com uma ordem dada pelas "ligações" chamadas ponteiros. Cada elemento dessa coleção, chamado nó, tem uma informação e uma referência para o próximo elemento. Na Figura 2.6 abaixo representamos num diagrama uma lista ligada para o conjunto $\{12, 37, 99\}$. A ideia agora

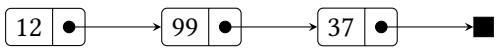


Figura 2.5: lista ligada.

é usar uma lista para a vizinhança de cada vértice de um grafo.

Lista de adjacência é um vetor $N(\,)$ de listas ligadas. A lista $N(i)$ contém os vizinhos do vértice i e cada nó dessa lista é composto por uma variável que armazena um vértice e um ponteiro que aponta para o próximo nó da lista.

A representação de uma grafo por listas de adjacências não é única, pois qualquer permutação dos nós em $N(i)$ define uma lista válida. Uma lista de adjacências para o G grafo representado na Figura 2.4 é representada na Figura 2.6.

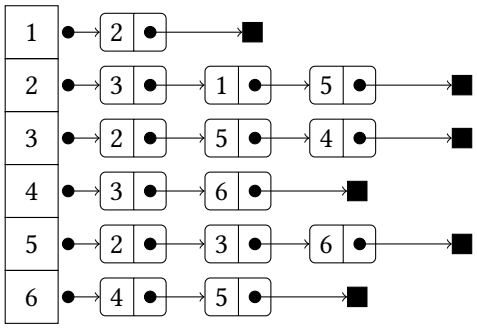


Figura 2.6: Lista de adjacências do grafo G na Figura 2.4.

2.3 Complexidade de algoritmos em grafos

Usualmente, a expressão que estabelece a complexidade de um algoritmo é escrita em função do ***tamanho da entrada***, isto é, o tamanho da representação de uma entrada para o algoritmo, em notação assintótica.

Se um algoritmo recebe um grafo (V, E) como entrada, e nada mais, então o parâmetro "tamanho da entrada" é o ***tamanho da representação*** do grafo. No caso de matriz de adjacências é razoável admitirmos que o tamanho da representação é $|V|^2$ e no caso de listas de adjacências $|V| + |E|$.

Neste texto entendemos por tempo de execução do algoritmo a ***complexidade de tempo no pior caso***, isto é, uma estimativa para o número máximo, dentre todas as entradas do mesmo tamanho, de instruções executadas e que é expressa em função do tamanho n da representação

$$T(n) = \max\{T(I) : I \text{ tem representação de tamanho } n\}$$

onde $T(I)$ denota o número de instruções executadas pelo algoritmo com entrada I.

Exemplo 2.4

Sejam A e B dois algoritmos distintos que resolvem o mesmo problema sobre um grafo $G = (V, E)$. O algoritmo A usa, no pior caso, exatos $|V|^2$ passos para executar a tarefa e o algoritmo B usa exatos $(|V| + |E|)\lfloor \log |E| \rfloor$ passos no pior caso. Para grafos com $|V|^2/4$ arestas o primeiro algoritmo é sempre melhor enquanto que para grafos com $1.000|V|$ arestas o segundo algoritmo é ***assintoticamente*** melhor (melhor para todo grafo com $|V| \geq 16.645$ vértices; o primeiro é melhor para $|V| \leq 16.644$).

Definição 2.2: notação O

Sejam f e g duas funções definidas nos naturais e que assumem valores naturais (ou, reais positivos), então

$$f(n) = O(g(n)), \quad \text{para } n \text{ suficientemente grande}$$

significa que *existem* constantes positivas c e n_0 tais que *para todo* $n \geq n_0$ vale

$$f(n) \leq c \cdot g(n).$$

Vejamos a complexidade de alguns algoritmos que executam tarefas simples comparando cada uma das representações dadas acima para um grafo fixo $G = (V, E)$

Tarefa	Matriz de adjacências	Lista de adjacências
$\{i, j\} \in E$?	$O(1)$	$O(\|V\|)$
Computar $d(i)$	$O(\|V\|)$	$O(\|V\|)$
Escrever E	$O(\|V\|^2)$	$O(\|E\|)$

Um algoritmo em grafo é de ***complexidade de tempo polinomial*** se o tempo de execução no pior caso é expressa por uma função polinomial no tamanho da representação. É importante notar que "ter complexidade polinomial" é independente da representação do grafo ser por matriz ou lista de adjacências pois essas representações são ***polinomialmente relacionadas*** (isto é, uma pode ser obtida da outra por algoritmo de tempo polinomial). Ademais, entendemos por ***algoritmo eficiente*** um algoritmo com complexidade polinomial.

Notação assintótica

As estimativas para o custo de um algoritmo são expressas usando notação assintótica. A análise assintótica foca no comportamento do algoritmo para entradas "suficientemente grandes" e simplifica a expressão da complexidade de tempo ao ignorar constantes multiplicativas e termos aditivos de ordem inferior, uma vez que essas diferenças não alteram a ordem de grandeza do crescimento da função. A análise assintótica permite uma avaliação de desempenho representativa, independente de tecnologia ou detalhes de implementação, tornando mais simples a comparação direta entre algoritmos.

Sejam $f, g : \mathbb{N} \to \mathbb{N}$ com $g(n) > 0$ para todo n suficientemente grande. Dizemos que f é ***assintoticamente muito menor*** que g e escrevemos

$$f(n) = o(g(n)) \text{ quando } n \to \infty \tag{2.2}$$

se, e só se,

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

Por exemplo,

1. $1 = o(\log(\log(n)))$.
2. $\log(\log(n)) = o(\log(n))$.
3. $\log(n) = o(n^\varepsilon)$ para todo $\varepsilon > 0$.
4. $n^\varepsilon = o(n^c)$ para quaisquer $0 < \varepsilon < 1 \leq c$.
5. $n^c = o(n^{\log n})$ para todo $1 \leq c$.
6. $n^{\log n} = o(\exp(n))$.
7. $\exp(n) = o(n^n)$.
8. $n^n = o(\exp(\exp(n)))$.

Também usamos a notação $f \ll g$ o que nos permite escrever, a partir do exemplo acima, a sequência monótona

$$1 \ll \log(\log(n)) \ll \log(n) \ll n^\varepsilon \ll n^c \ll n^{\log n} \ll e^n \ll n^n \ll e^{e^n}.$$

Dizemos que f ***assintoticamente menor*** que g se

$$f(n) = O(g(n)). \tag{2.3}$$

Na definições dadas nas equações (2.2) e (2.3) o símbolo "=" não é a igualdade no sentido usual, é um abuso da notação que permitimos em troca de algumas conveniências. Temos que $n = O(n^2)$ e $n^2+2n+1 = O(n^2)$ mas $n \neq n^2 + 2n + 1$. Quando usamos essas definições.

Proposição 2.1

Se $f(n) = o(g(n))$ então $f(n) = O(g(n))$.

A prova é imediata da definição de limite. A recíproca da Proposição 2.1 não vale, como pode ser visto tomando-se $f(n) = g(n) = n^2$.

Proposição 2.2

Se $f_1(n) = O(g_1(n))$ e $f_2(n) = O(g_2(n))$ então

1. $f_1(n) + f_2(n) = O(\max\{ g_1(n), g_2(n) \})$.
2. $f_1(n) \cdot f_2(n) = O(g_1(n) \cdot g_2(n))$.
3. $a \cdot f_1(n) = O(g_1(n))$ para toda constante $a \in \mathbb{R}$.

Demonstração. Vamos provar o item 1. Digamos que $f_1(n) \leq c_1 g_1(n)$ para todo $n \geq n_1$ e que $f_2(n) \leq c_2 g_2(n)$ para todo $n \geq n_2$, onde n_1, n_2, c_1, c_2 são as constantes dadas pela definição de notação O. Então

$$\begin{aligned} f_1(n) + f_2(n) &\leq \max\{ c_1, c_2 \}(g_1(n) + g_2(n)) \text{ com } c = \max\{ c_1, c_2 \} \\ &\leq c(2 \cdot \max\{ g_1(n), g_2(n) \}) \end{aligned}$$

para todo $n \geq \max\{ n_1, n_2 \}$, o que prova a afirmação.

As outras propriedades são deduzidas de modo análogo e são deixadas como exercício. □

É preciso observamos alguns cuidados com o teorema acima. Para

inteiros positivos n e k não vale que $1^k + 2^k + \cdots + (n-1)^k + n^k = O(\max\{ 1^k, 2^k, \ldots, (n-1)^k, n^k \}) = O(n^k)$. O problema aqui é que o máximo só pode ser tomado sobre um número de termos que não dependa de n. De fato, temos que $1^k + \cdots + n^k = O(n^{k+1})$ e que $1^k + \cdots + n^k \neq O(n^k)$.

Fica como exercício a verificação do seguinte resultado.

Proposição 2.3: transitividade

Se $f(n) = O(g(n))$ e $g(n) = O(h(n))$ então $f(n) = O(h(n))$.

Mais alguns exemplos são dados a seguir.

1. $an^2 + bn + c = O(n^2)$ para toda constante $a > 0$.
   Primeiro, observamos que, para n suficientemente grande, $|an^2 + bn + c| \leq |a|n^2 + |b|n + |c|$ e agora usamos a Proposição 2.2 em cada operando das somas, de $an^2 = O(n^2)$, $|b|n = O(n)$ e $|c| = O(1)$ temos $an^2 + |b|n + |c| = O(\max\{ n^2, n, 1 \}) = O(n^2)$. Analogamente, para todo $k \in \mathbb{N}$

$$\sum_{i=0}^{k} a_i n^i = O(n^k).$$

2. $n \log(n!) = O(n^2 \log n)$.
   Primeiro, temos $n = O(n)$. Depois, $n! = \prod_{i=1}^{n} i < \prod_{i=1}^{n} n = n^n$. Como log é crescente $\log(n!) < \log(n^n) = n \log(n)$, portanto $\log(n!) = O(n \log(n))$. Pela Proposição 2.2 $n \log(n!) = O(n^2 \log n)$.
3. Para toda constante $a > 1$, $\log_a(n) = O(\log(n))$. De fato,

$$\log_a(n) = \frac{1}{\log a} \log n$$

   porém $\frac{1}{\log a} = O(1)$ e $\log n = O(\log n)$ e pela Proposição 2.2 $\log_a(n) = O(\log(n))$.

Um algoritmo eficiente com respeito ao tempo de execução é um algoritmo que nas instâncias de tamanho n tem tempo de execução $O(n^k)$ para algum inteiro positivo k fixo.

Convenções de uso da notação assintótica

Ao usar notação assintótica nós desconsideramos os coeficientes, por exemplo, usamos $O(n^2)$ ao invés de $O(3n^2)$ e $O(1)$ ao invés de $O(1024)$ ainda que, como classes de funções, $O(n^2) = O(3n^2)$ e $O(1) = O(1024)$.

Escrevemos no argumento de $O(\cdot)$ somente o termo mais significativo, por exemplo, usamos $O(n^2)$ ao invés de $O(2n^2 + 5n \log n + 4)$. Nesse caso, da Proposição 2.2 vale que $2n^2 + 5n \log n + 4 = O(\max\{ n^2, n \log n, 1 \}) = O(n^2)$.

Quando notação assintótica aparece em equações, na forma "expressão 1 = expressão 2" onde "expressão" são expressões algébricas que envolvem notação assintótica, os termos assintóticos em "expressão 1" são quantificados universalmente, enquanto que os termos assintóticos em "expressão 2" são quantificados existencialmente. Por exemplo, em $n^3 + O(n^2) = O(n^3) + n^2 + n$ entendemos como: existe um $n_0 > 0$ tal que para todo $n \geq n_0$

para todo $f(n)$ em $O(n^2)$, existe $g(n)$ em $O(n^3)$ tal que $n^3+f(n) = g(n)+n^2+n$.

Notação Ω e Θ

A notação Ω adaptada por Donald Knuth da notação introduzida por outros matemáticos é definida por

$$f(n) = \Omega(g(n)) \text{ se e somente se } g(n) = O(f(n)).$$

Escrevemos $f(n) = \Theta(g(n))$ se e somente se $f(n) = O(g(n))$ e $g(n) = O(f(n))$.

P×NP

Atualmente não sabemos se existe algoritmo eficiente para vários problemas em grafos.

Por exemplo, não sabemos se existe algoritmo eficiente que receba dois grafos e decida se eles são isomorfos.

- ***O problema do isomorfismo de grafos***: Dados os grafos $G = (V, E)$ e $H = (V, E')$ decidir se eles são isomorfos.

Entretanto, não é difícil projetar um algoritmo de tempo polinomial que recebe a terna (G, H, f), formada por dois grafos e uma função $f : V(G) \to V(H)$ e devolve ***sim*** caso G e H são isomorfos e f é o isomorfismo, caso contrário devolve ***não***.

Em linguagem técnica, da Teoria da Computação, dizemos que ***o problema do isomorfismo de grafos está na classe* NP *de complexidade de problemas computacionais.*** Não é sabido se esse problema é NP-difícil, caso seja, um algoritmo eficiente para esse problema implica na existência de algoritmo eficiente para todo problema computacional em NP.

O problema complementar está em coNP:

- ***O problema do não-isomorfismo de grafos***: Dados os grafos $G = (V, E)$ e $H = (V, E')$ decidir se eles são não-isomorfos.

Também não se conhece algoritmo de tempo polinomial no tamanho dos grafos para decidir se dois grafos não são isomorfos. Mais do que isso, não se conhece um algoritmo de tempo polinomial no tamanho dos grafos que receba como entrada uma terna (G, H, P) onde P é uma "prov curta" (i.e., de tamanho polinomial em $|V| + |E| + |E'|$) e que decida se P é uma prova do não-isomorfismo entre G e H. Em lingua-

gem técnica dizemos que *não se sabe se o problema do não-isomorfismo de grafos está na classe* **NP** *de complexidade computacional.*

Em 2015, Babai apresentou um algoritmo para o problema de isomorfismo de grafos com tempo de execução ***quase-polinomial***. Assim, embora não se saiba se existe um algoritmo de tempo polinomial, sabemos que o problema de isomorfismo pode ser resolvido em tempo $\exp((\log n)^c)$, onde n representa o tamanho da entrada e $c > 0$ constante. O o mesmo resultado fornece um algoritmo quase-polinomial para GNI.

Assim, a dificuldade não está simplesmente em encontrar *algum* argumento para provar que dois grafos não são isomorfos. A questão mais profunda é determinar um procedimento geral e eficiente que, dados dois grafos, decida se eles pertencem à mesma classe de isomorfismo. É justamente essa questão que dá origem ao problema de isomorfismo de grafos e à sua interessante relação com a teoria da complexidade computacional.

Exercícios

Exercício 39. *Determine quais pares dentre os grafos abaixo são isomorfos.*

(i) G_1 *tal que* $V(G_1) = \{v_1, u_1, w_1, x_1, y_1, z_1\}$ *e*
$E(G_1) = \{\{u_1, v_1\}, \{u_1, w_1\}, \{v_1, w_1\}, \{v_1, x_1\}, \{w_1, y_1\}, \{x_1, y_1\}, \{x_1, z_1\}\}$;

(ii) G_2 *tal que* $V(G_2) = \{v_2, u_2, w_2, x_2, y_2, z_2\}$ *e*
$E(G_2) = \{\{u_2, v_2\}, \{u_2, w_2\}, \{v_2, w_2\}, \{v_2, x_2\}, \{w_2, y_2\}, \{x_2, y_2\}, \{y_2, z_2\}\}$;

(iii) G_3 *tal que* $V(G_3) = \{v_3, u_3, w_3, x_3, y_3, z_3\}$ *e*
$E(G_3) = \{\{u_3, v_3\}, \{u_3, w_3\}, \{v_3, w_3\}, \{v_3, x_3\}, \{w_3, y_3\}, \{x_3, y_3\}, \{u_3, z_3\}\}$.

Exercício 40. *Decida se o grafo representado pelo diagrama na Figura 2.7 abaixo é o grafo de Petersen.*

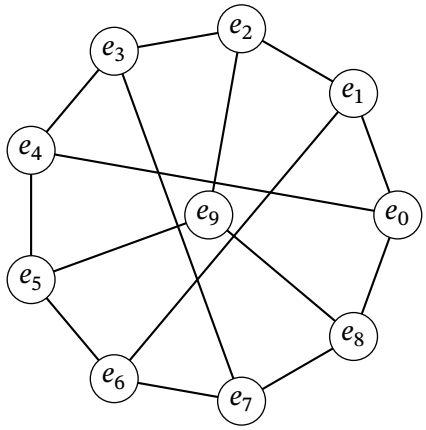


Figura 2.7: esse é o grafo de Petersen?

Exercício 41. *Decida se os dois grafos* 4*-regular representados pelos dois diagramas da Figura 2.8 abaixo são isomorfos.*

Exercício 42. *Mostre que existem* 11 *grafos não-isomorfos com* 4 *vértices.*

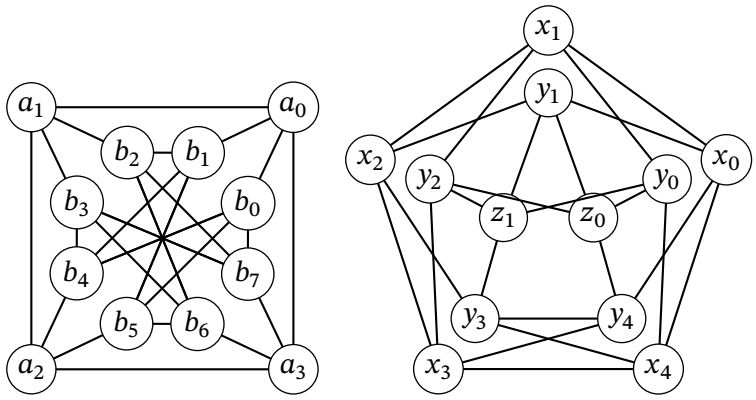


Figura 2.8: esses grafos são isomorfos?

Exercício 43. *Sejam G e H grafos isomorfos e $f : G \to H$ um isomorfismo. É verdade que $G[U]$ é isomorfo a $H[f(U)]$ para todo $U \subseteq V(G)$? Justifique.*

Exercício 44. *Mostre que o grafo de Petersen é isomorfo ao complemento do grafo linha do K^5.*

Exercício 45. *Um* **automorfismo** *de um grafo é um isomorfismo do grafo sobre ele mesmo. Quantos automorfismos tem um grafo completo?*

Exercício 46. *Mostre que o conjunto de automorfismos de um grafo com a operação de composição de funções definem um grupo.*

Exercício 47. *Qual o número de grafos distintos sobre um conjunto de vértices V de tamanho n?*

Exercício 48. *Prove que há pelo menos $2^{\binom{n}{2}}(n!)^{-1}$ grafos não isomorfos sobre um conjunto de vértices V de tamanho n.*

Exercício 49. *Um grafo G é* **vértice-transitivo** *se para quaisquer $u, v \in V(G)$ existe um automorfismo f de G com $f(v) = u$. Analogamente, G é* **aresta-transitivo** *se para quaisquer arestas $\{x, y\}, \{z, w\} \in E(G)$ existe um automorfismo f de G tal que $\{f(x), f(y)\} = \{z, w\}$.*

Dê um exemplo de grafo vértice-transitivo. Dê um exemplo de grafo aresta-transitivo. Dê um exemplo de grafo aresta-transitivo mas não vértice-transitivo.

Exercício 50. *Considere o grafo $G = (V, E)$, com $V = [n]$ e*

$$E = \{\{1,2\},\{2,3\},\{2,5\},\{3,4\},\{3,5\},\{4,6\},\{5,6\}\}.$$

(a) Escreva a matriz de adjacências A_G.
(b) Escreva uma representação de G por listas de adjacências, usando uma lista ligada para cada vértice. A ordem dos vizinhos em cada lista é livre.
(c) Quantos nós possuem, no total, as listas de adjacências? Relacione sua resposta com $|E|$.

(d) *Suponha que se queira determinar se $i, j \in E$. Explique como realizar essa consulta nas duas representações e justifique as complexidades $O(1)$ e $O(|V|)$ apresentadas no texto.*

Exercício 51. *Seja $G = (V, E)$ um grafo com $|V| = n$ e $|E| = m$.*

(a) *Mostre que uma representação de G por matriz de adjacências utiliza $\Theta(n^2)$ posições, enquanto uma representação por listas de adjacências utiliza $\Theta(n + m)$ posições.*

(b) *Para cada uma das situações abaixo, indique qual representação é mais adequada, matriz ou listas de adjacências, e justifique sua escolha comparando o custo das operações relevantes.*

- *$m = \Theta(n)$ e o algoritmo precisa percorrer todos os vizinhos de cada vértice.*
- *$m = \Theta(n^2)$ e o algoritmo realiza muitas consultas do tipo "$i, j \in E$?".*
- *$m = \Theta(n)$ e o algoritmo precisa apenas escrever todas as arestas.*
- *$m = \Theta(n^2)$ e o algoritmo precisa escrever todas as arestas.*

(c) *Explique por que, apesar das diferenças de eficiência, as duas representações não alteram a classificação de um algoritmo como sendo de tempo polinomial.*

Exercício 52. *Sejam G um grafo sobre o conjunto de vértices $[n]$, A a sua matriz de adjacências e $b(i, j)$ as entradas da matriz A^2. Que parâmetro de G está associado ao número $b(i, i)$, para cada $i \in V(G)$? Qual a relação entre $b(i, j)$ e, $N_G(i)$ e $N_G(j)$ quando $i \neq j$?*

Exercício 53. *Seja G um grafo qualquer sobre o conjunto de vértices $[n]$ e A sua matriz de adjacências. Como é possível determinar o número de triângulos de um grafo G qualquer a partir de A^3?*

Exercício 54. *O* **traço** *de uma matriz A, denotado por $\mathrm{tr}(A)$, é a soma dos elementos na diagonal da matriz. Quanto vale o traço de uma matriz de adjacências? Quanto vale $\mathrm{tr}(A^2)$ em função de $E(G)$?*

Exercício 55. *Por A ser uma matriz simétrica, sabemos da Álgebra Linear que todos os seus autovalores são números reais. Determine os autovalores da matriz de adjacências do grafo completo K^n sobre o conjunto de vértices $[n]$. Mostre que se G é um grafo r-regular sobre o conjunto de vértices $[n]$, então r é uma autovalor de A.*

Exercício 56. *Sejam G um grafo sobre o conjunto de vértices $[n]$ e A sua matriz de adjacências. Prove que se os autovalores de A são distintos então o grupo dos automorfismos (Exercício 46) é comutativo.*

Bibliografia

[1] BONDY, J. A.; MURTY, U. S. R. *Graph Theory*. New York: Springer, 2008. (Graduate Texts in Mathematics, v. 244).

[2] DIESTEL, R. *Graph Theory*. 6ª ed. Heidelberg: Springer-Verlag, 2025. (Graduate Texts in Mathematics, v. 173).

[3] ERICKSON, Jeff. *Algorithms*. 1. ed. [S. l.], 2019. Capítulos 5–11. Disponível em: https://jeffe.cs.illinois.edu/teaching/algorithms/.

[4] CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford. *Algoritmos: teoria e prática*. 3. ed. Rio de Janeiro: Elsevier, 2012.

[5] SEDGEWICK, Robert. *Algorithms in C, Part 5: Graph Algorithms*. 3. ed. Reading, USA: Addison Wesley Professional, 2002. 482 p.

[6] DASGUPTA, Sanjoy; PAPADIMITRIOU, Christos H.; VAZIRANI, Umesh V. *Algorithms*. Boston, USA: McGraw-Hill, 2008.

[7] JOYNER, David; VAN NGUYEN, Minh; COHEN, Nathann. *Algorithmic Graph Theory*. Version 0.7. 2013. Disponível em: http://code.google.com/archive/p/graphbook/. Online.

[8] ROUGHGARDEN, Tim. *Algorithms Illuminated, Part 2: Graph Algorithms and Data Structures*. New York: Soundlikeyourself Publishing, 2018. Online.

[9] BRASSARD, Gilles; BRATLEY, Paul. *Fundamentals of Algorithmics*. Englewood Cliffs: Prentice Hall, 1996.

[10] MANBER, Udi. *Introduction to Algorithms: A Creative Approach*. Addison-Wesley.