

Textos de

# Matemática Discreta

Teoria de Grafos

Eliana Costa e Silva

[eos@estg.ipp.pt](mailto:eos@estg.ipp.pt)

Para os cursos de:

Licenciatura em Segurança Informática

em Redes de Computadores

Licenciatura em Engenharia Informática

O uso destes apontamentos como **único** material de estudo é fortemente desaconselhado.

O estudante deve também consultar a bibliografia recomendada indicada na Ficha da Unidade Curricular e disponível neste documento, nomeadamente, [4], [3], [1], [2].

# Conteúdo

<b>1</b>	<b>Teoria de Grafos</b>	<b>1</b>
1	Grafos e a sua representação . . . . .	2
1.1	Terminologia . . . . .	5
1.2	Representação de Grafos . . . . .	7
1.3	Conectividade . . . . .	13
2	Caminhos Eulerianos e Hamiltonianos . . . . .	17
3	Árvores e suas aplicações . . . . .	30



# Teoria de Grafos

**Grafos** (*Graphs*) são estruturas discretas que consistem num conjunto de vértices (*nodes*) e arestas (*edges*) que ligam esses vértices.

Árvores são casos particulares de grafos onde as ligações entre os vértices não são circulares.

Grafos podem ser utilizados como modelos numa grande variedade de áreas, podendo ser utilizados, por exemplo, para:

- representar a competição entre diversas espécies de um eco sistema;
- representar quem influencia quem numa organização;
- modelar redes de conhecimentos entre pessoas;
- modelar a relação entre chamadas telefónicas e números de telefone;
- modelar a relação entre os *links* e os *websites*;
- determinar se dois computadores estão ligados numa rede de computadores;
- ...

Devido ao interesse relativamente recente em Teoria de Grafos, e devido à sua aplicabilidade numa ampla variedade de áreas, foram introduzidas muitas terminologias diferentes, pelo que é necessário ter atenção ao significado de cada termo que está a ser usado. De facto, embora a terminologia usada pelos matemáticos tenha vindo a ser cada vez mais padronizada, noutras áreas esta é ainda muito diversificada. No entanto, existem três questões-chave que nos podem ajudar a perceber a estrutura de um grafo:

- As arestas dos grafos são orientadas ou não?
- Se o grafo é não orientado, existem várias arestas que ligam o mesmo par de vértices? Se o grafo é orientado, existem arestas múltiplas?
- Existem lacetes?

# 1 Grafos e a sua representação

## Definição 1:

Um **grafo** não orientado  $G = (V, E)$  consiste num conjunto  $V$ , não vazio, de vértices (ou nodos), e num conjunto  $E$  de arestas, as quais estão associadas a um ou dois vértices.

Se o conjunto  $V$  é:

- finito dizemos que o grafo é **finito**;
- infinito dizemos que o grafo é **infinito**.

## Exemplo 1:

(Adaptado de (Rosen, 2012))

Considere uma rede de computadores composta por **data centers** e *links* entre os mesmos (Figura 1.1). Os vértices do grafo representam os *data centers* e as arestas representam os *links* (comunicação entre os *data centers*).

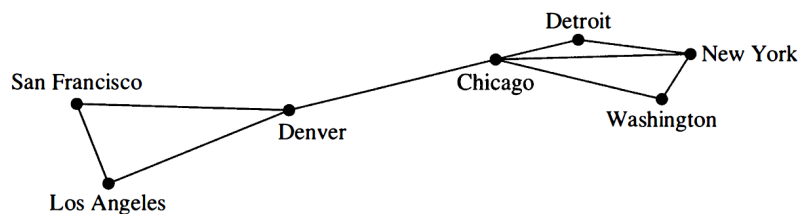


Figura 1.1: Exemplo de uma rede de computadores (retirado de (Rosen, 2012))

Neste exemplo da Figura 1.1:

- cada aresta liga dois vértices diferentes;
- nenhuma aresta liga um vértice a si mesmo;
- não existem duas arestas diferentes que ligam os mesmos vértices.

Este é um exemplo de um **grafo simples** (*simple graph*).

No entanto, numa rede de computadores podemos ter várias ligações entre os mesmos *data centers* (Figura 1.2). Neste caso dizemos que temos um **multigrafo** (*multigraph*).

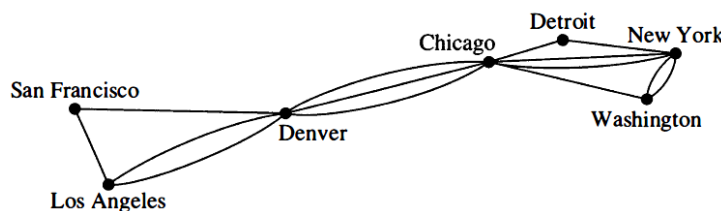


Figura 1.2: Uma rede de computadores com múltiplos *links* entre *data centers* (Retirado de (Rosen, 2012))

Quando temos  $m$  arestas diferentes associadas ao mesmo par não ordenado de vértices  $(u, v)$ , dizemos que  $(u, v)$  é uma aresta de multiplicidade  $m$ .

Por exemplo, quando temos 4 arestas diferentes associadas ao mesmo par de vértices dizemos que a aresta tem multiplicidade 4.

Por vezes temos *links* de um *data center* para si mesmo (eventualmente utilizado para diagnóstico de problemas!). Um exemplo de tal rede é apresentada na Figura 1.3. Estas arestas são chamadas de **lacete** (*loop*), e por vezes podemos ter mais do que um lacete em cada vértice. Grafos que incluem vértices de multiplicidade superior a 1 e lacetes podem designar-se por **pseudografos** (*pseudographs*).

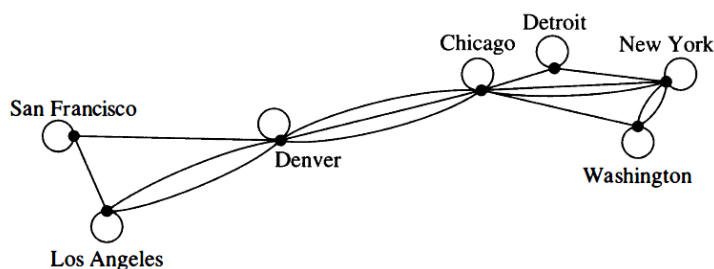


Figura 1.3: Uma rede de computadores com diagnóstico (Retirado de (Rosen, 2012))

Os grafos apresentados nas Figuras 1.1, 1.2 e 1.3 são exemplos de **grafos não orientados**. Dizemos que as suas arestas são não orientadas.

No entanto, numa rede de computadores podemos ter *links* que operam num só sentido. Por exemplo, quando há uma grande quantidade de dados que alguns *data centers* enviam, mas onde há poucos ou nenhuns dados enviados no sentido oposto! Este é um exemplo de um **grafo orientado** ou **digrafo** (ver Figura 1.4). Note que, a cada aresta de um grafo orientado está associada a um par ordenado.

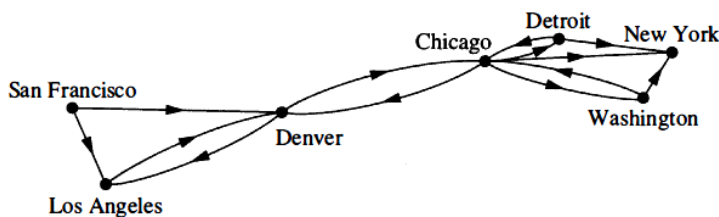


Figura 1.4: Uma rede de computadores com *links* de comunicação num só sentido (Retirado de (Rosen, 2012))

#### Nota:

Anteriormente utilizamos digrafos para representar relações. No entanto, a definição de digrafo é mais geral.

**Definição 2:**

Um **grafo orientado** (ou **digrafo**) é um par  $\vec{G} = (V, E)$ , onde  $V$  é um conjunto não vazio de vértices e  $E$  é um conjunto de **arestas orientadas** ou **arcos** (*directed edges* ou *arcs*). Cada aresta está associada a um par ordenado de vértices, portanto,  $E$  é subconjunto de  $V \times V$ .

Se  $e = (u, v)$  é uma aresta de  $\vec{G}$ , dizemos que  $u$  é a **cauda** (*start*) e  $v$  é a **cabeça** (*end*) da aresta  $e$ .

- Um digrafo que não possui arestas múltiplas e não tem lacetes é designado por **grafo orientado simples**;
- Um digrafo que possui arestas múltiplas é designado por **multigrafo orientado** ou multidigrafo;
- Em alguns modelos temos algumas arestas não orientadas enquanto que outras são orientadas. Nesses casos temos um **mixed graph**.

A Tabela 1.1 sumariza a terminologia usada.

Tipo	Arestas	Arestas múltiplas?	Lacetes?
Grafo simples	não orientadas	Não	Não
Multigrafo	não orientadas	Sim	Não
Pseudografo	não orientadas	Sim	Sim
Grafo orientado simples	orientadas	Não	Não
Multigrafo orientado	orientadas	Sim	Sim
<i>Mixed graph</i>	orientadas e não orientadas	Sim	Sim

Tabela 1.1: Terminologia.

**Observação:**

A representação de um grafo não orientado e um digrafo não é única, visto que a disposição dos vértices e o traçado das arestas é livre.

No entanto, toda a representação gráfica determina um único grafo!

**Exemplo 2:**

*The Web Graph* (adaptado de (Rosen, 2012))

A *World Wide Web* pode ser modelada como um digrafo onde cada *webpage* é representada por um vértice e onde cada aresta começa na *webpage a* e termina na *webpage b*, se existe um *link* em *a* que aponte para *b* (Figura 1.5).

Praticamente a cada segundo novas webpages são criadas e removidas, portanto, a alteração do grafo que representa a Web muda de forma praticamente contínua. Estima-se que em 2010 o *Web Graph* tinha pelo menos 55 biliões de vértices e 1 trilião de arestas, daí o grande interesse em estudar as propriedades deste grafo para melhor perceber a Web.

Este grafo é usado, por exemplo, para calcular o *PageRank* das páginas www. Para mais informações ver, por exemplo, <http://webgraph.dsi.unimi.it/> ou <http://webgraph.di.unimi.it/>



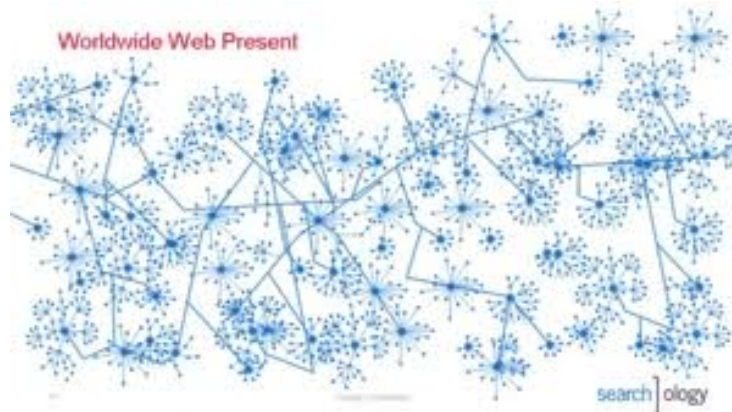


Figura 1.5: *The Web Graph* (Retirada de: <http://driverlayer.com/img/webgraph/11/>)

## 1.1 Terminologia

Com base no exemplo seguinte é introduzido algum vocabulário básico de Teoria de Grafos.

### Exemplo 3:

Consideremos o grafo não orientado  $G$  e o grafo orientado  $\vec{G}$ , cujos conjuntos dos vértices e das arestas são, respetivamente,

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

onde  $e_1 = (2, 1)$  e  $e_2 = (2, 1)$ , e  $\vec{e}_1 = (2, 1)$  e  $\vec{e}_2 = (2, 1)$ .

A sua representação é apresentada na Figura 1.6.

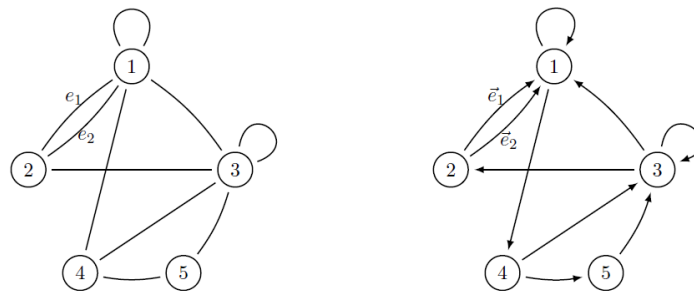


Figura 1.6: Grafo orientado e grafo não orientado

- Dizemos que uma aresta é **incidente** nos seus vértices extremos e que tais vértices são **adjacentes** (ou vizinhos).
- O conjunto de todos os vértices adjacentes a um dado vértice  $v$  designa-se de **vizinhança** de  $v$  e denota-se por  $\mathcal{N}_G(v)$ .

- Na Figura 1.6, os vértices 1 e 3 são adjacentes e  $\mathcal{N}_G(1) = \{1, 2, 3, 4\}$ .
- As arestas  $(1, 1)$  e  $(3, 3)$  são lacetes.
- Duas ou mais arestas unindo o mesmo par de vértices designam-se de **arestas múltiplas** (ou arestas paralelas).

Note-se que, num digrafo, duas arestas múltiplas têm a mesma orientação de um vértice  $u$  para um vértice  $v$ .

- As arestas que unem o vértice 2 ao vértice 1 são arestas múltiplas. De modo a distinguirmos duas arestas múltiplas, etiquetamos as arestas, fazendo  $e_1 = (2, 1)$  e  $e_2 = (2, 1)$ .
- A **ordem** de um grafo  $G$  é o número de vértices que o constitui, ou seja,  $|V(G)|$ .
- A **dimensão** de um grafo  $G$  é o número de arestas que ele possui, ou seja,  $|E(G)|$ .
- Os grafos apresentados na Figura 1.6 têm ordem 5 e dimensão 10, respetivamente.
- O **grau** de um vértice (de um **grafo não orientado**,  $G$ ) é o número de arestas incidentes no vértice  $v$ , onde os lacetes, caso existam, contam duas vezes para o grau do vértice. Assim, no grafo à esquerda na Figura 1.6 temos

$$\text{grau}(1) = 6, \text{grau}(2) = 3, \text{grau}(3) = 6, \text{grau}(4) = 3 \text{ e } \text{grau}(5) = 2.$$

- Num **grafo orientado**,  $\vec{G}$ , designamos por **grau de entrada** do vértice  $v$ , e denotamos por  $\text{grau}^e(v)$ , o número de arestas que entram no vértice  $v$ .

Dualmente, designamos por **grau de saída** do vértice  $v$ , e denotamos por  $\text{grau}^s(v)$ , o número de arestas que saem do vértice  $v$ .

No grafo apresentado à direita na Figura 1.6 temos:

$$\begin{aligned} \text{grau}^e(1) &= 4, \text{grau}^s(1) = 2, \\ \text{grau}^e(2) &= 1, \text{grau}^s(2) = 2, \\ \text{grau}^e(3) &= 3, \text{grau}^s(3) = 3, \\ \text{grau}^e(4) &= 1, \text{grau}^s(4) = 2, \\ \text{grau}^e(5) &= 1 \text{ e } \text{grau}^s(5) = 1. \end{aligned}$$

É fácil notar que, qualquer que seja o grafo (orientado ou não), cada aresta contribui duas vezes para a soma dos graus dos vértices (pois é incidente nos dois vértices extremos!), mesmo quando se trata de um lacete.

Portanto, **a soma dos graus de todos os vértices é o dobro do número de arestas do grafo.**

### Teorema 1:

Seja  $G = (V, E)$  um **grafo não orientado**. Então,

$$2|E| = \sum_{v \in V} \text{grau}(v).$$

Note que este resultado se aplica mesmo na presença de lacetes e arestas múltiplas! □

**Exemplo 4:**

Quantas arestas tem um grafo com 10 vértices sendo cada um de multiplicidade 6?

**Resposta:**

Como a soma dos graus dos vértices é  $6 \times 10 = 60$ , temos  $\frac{60}{2} = 30$  arestas.

**Teorema 2:**

Um grafo **não orientado** tem um número par de vértices de grau ímpar. □

(Pode consultar a demonstração deste resultado na página 653 de (Rosen, 2012).)

**Teorema 3:**

Seja  $\vec{G} = (V, E)$  um **grafo orientado**. Então,

$$|E| = \sum_{v \in V} \text{grau}^e(v) = \sum_{v \in V} \text{grau}^s(v).$$

□

O Teorema anterior afirma que a soma dos graus de saída dos vértices de um grafo orientado é igual à soma dos graus de entrada, e igual ao número de arestas do grafo.

Para o grafo orientado apresentado na Figura 1.6 temos 10 arestas e:

$$\sum_{v \in V} \text{grau}^e(v) = 4 + 1 + 3 + 1 + 1 = 10$$

e

$$\sum_{v \in V} \text{grau}^s(v) = 2 + 2 + 3 + 2 + 1 = 10.$$

## 1.2 Representação de Grafos

Existem muitas formas de representar um grafo. Conhecer as diferentes representações e saber como escolher a mais adequada é muito importante. De seguida apresentamos algumas formas de representar grafos.

### Listas e Matrizes de adjacências

Para representar grafos sem arestas múltiplas podemos usar **lista de adjacências**. Estas listas especificam os vértices adjacentes a cada vértice de um grafo. No entanto, quando os grafos têm um número elevado de arestas escrever tal lista pode ser extremamente pesado.

Para simplificar podemos usar matrizes para representar grafos. As mais usadas são as **matrizes de adjacências**.

**Definição 3:**

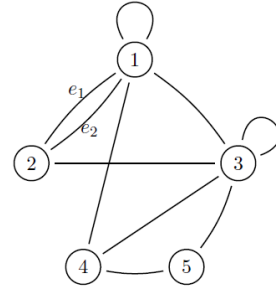
A **matriz de adjacências** de um **grafo não orientado**  $G(V, E)$ , que denotamos por  $M_G(a_{ij})$  (ou, simplesmente,  $M_G$ ), é a matriz de dimensão  $|V| \times |V|$  onde  $a_{ij}$  é igual ao número de arestas incidentes nos vértices  $v_i$  e  $v_j$ .

**Exemplo 5:**

Considerando novamente o grafo não orientado apresentado na Figura 1.6, definido por:

$$V(G) = \{1, 2, 3, 4, 5\}$$

$$E(G) = \{(1, 1), (1, 4), (2, 1), (2, 1), (3, 1), (3, 2), (3, 3), (4, 3), (4, 5), (5, 3)\}$$



temos a seguinte matriz de adjacência:

$$M_G = \begin{bmatrix} 1 & 2 & 1 & 1 & 0 \\ 2 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

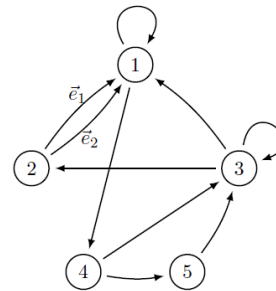
**Definição 4:**

A matriz de adjacência de um **digrafo**  $\vec{G} = (V, E)$ ,  $M_{\vec{G}}(a_{ij})$  é tal que  $a_{ij}$  é igual ao número de arestas com cauda em  $v_i$  e cabeça em  $v_j$ .

Considerando o grafo orientado apresentado na Figura 1.6, definido por:

$$V(\vec{G}) = \{1, 2, 3, 4, 5\}$$

$$E(\vec{G}) = \{(1, 1), (1, 4), \vec{e}_1, \vec{e}_2, (3, 1), (3, 2), (3, 3), (4, 3), (4, 5), (5, 3)\}$$

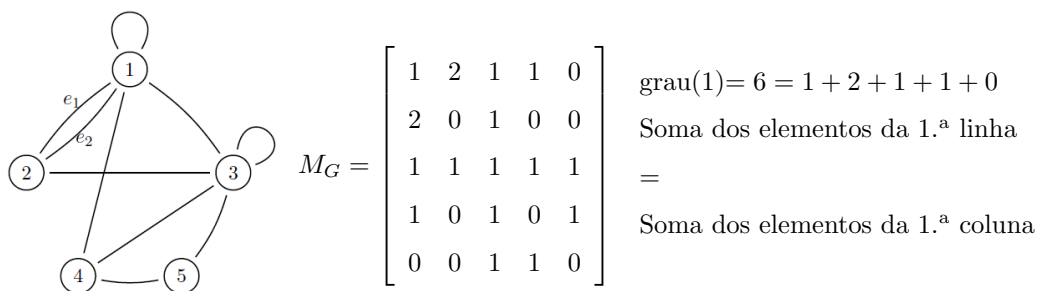


temos a seguinte matriz de adjacência:

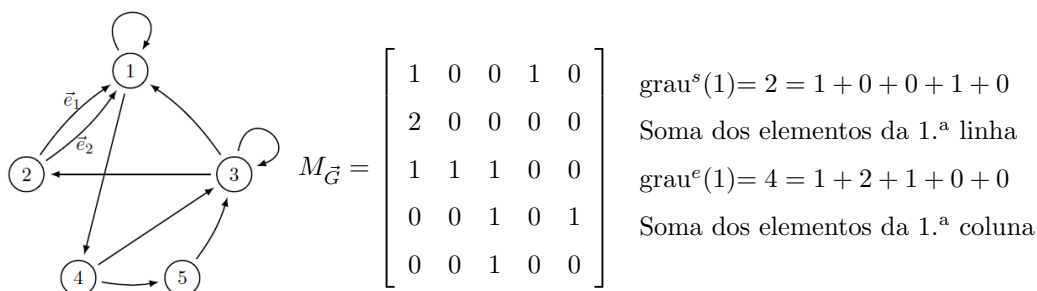
$$M_{\vec{G}} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

**Note que:**

- Num **grafo não orientado**,
  - a matriz de adjacências é **simétrica**;
  - a soma das entradas de cada linha da matriz de adjacências, multiplicando por 2 a entrada da diagonal principal, é igual ao grau do vértice correspondente a essa linha.  
(Tem-se o mesmo resultado em relação às colunas da matriz.)



- Num **digrafo**,
  - a soma das entradas de cada **linha** da matriz de adjacências é igual ao grau de **saída** do vértice correspondente e
  - a soma das entradas de cada **coluna** da matriz de adjacências é igual ao grau de **entrada** do vértice correspondente.

**Isomorfismo de grafos**

Muitas vezes precisamos de saber se é possível representar dois grafos da mesma forma. Isto é, determinar se os grafos têm a mesma estrutura (se ignorarmos as identidades de seus vértices).

Por exemplo, em química, os grafos são usadas para modelar compostos químicos. Compostos diferentes podem ter a mesma fórmula molecular, mas podem diferir em estrutura. Tais compostos podem ser representados por grafos que não podem ser extraídos da mesma maneira. Podemos verificar se um determinado composto é conhecido ou não recorrendo à representação dos compostos previamente conhecidos.

**Definição 5:**

Dois grafos  $G = (V(G), E(G))$  e  $H = (V(H), E(H))$  dizem-se **isomorfos** se existir uma **bijeção** entre o conjunto dos vértices de  $G$  e o conjunto dos vértices de  $H$  e uma **bijeção** entre o conjunto das arestas de  $G$  e o conjunto das arestas de  $H$  que preservam a relação de adjacência e de incidência, ou seja, existem

$$\phi : V(G) \longrightarrow V(H) \text{ e } \psi : E(G) \longrightarrow E(H)$$

tais que

$$e = (u, v) \in E(G) \text{ se e só se } \psi(e) = (\phi(u), \phi(v)) \in E(H).$$

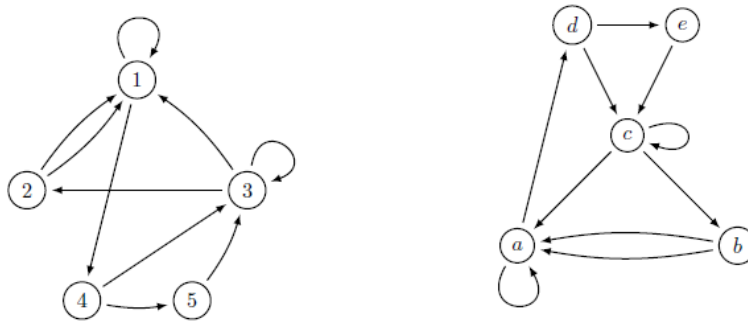


Figura 1.7: Grafos isomorfos

**Exemplo 6:**

Os grafos orientados apresentados na Figura 1.7 são isomorfos.

Basta observar que a **bijeção**  $\phi$ , entre os vértices, preserva a relação de adjacência e a **bijeção**, entre as arestas, satisfaz a relação de incidência:

$\phi : V(G) \longrightarrow V(H)$	$\psi : E(G) \longrightarrow E(H)$
1 $\mapsto$ a	(1, 1) $\mapsto$ (a, a)
2 $\mapsto$ b	(1, 4) $\mapsto$ (a, d)
3 $\mapsto$ c	(2, 1) $\mapsto$ (b, a)
4 $\mapsto$ d	(3, 1) $\mapsto$ (c, a)
5 $\mapsto$ e	(3, 2) $\mapsto$ (c, b)
	(4, 3) $\mapsto$ (d, c)
	(4, 5) $\mapsto$ (d, e)
	(5, 3) $\mapsto$ (e, c)

Muitas vezes **não é fácil** determinar se 2 grafos simples são isomorfos. De facto existem  $n!$  correspondências bijetivas entre dois grafos com  $n$  vértices, o que pode tornar impraticável encontrar tal bijeção para  $n$  grande. Os softwares NAUTY e Traces, disponíveis em <http://pallini.di.uniroma1.it/>, permitem determinar se dois grafos são isomorfos.

Para verificar que dois grafos **não** são isomorfos podemos verificar se satisfazem algumas condições como:

- Os grafos não têm o mesmo número de vértices.
- Os grafos não têm o mesmo número de arestas.
- Um dos grafos tem arestas múltiplas e o outro não.
- Um dos grafos tem lacetes e o outro não.
- Os vértices dos dois grafos não têm o mesmo grau.

### Algumas aplicações de grafos isomorfos

**Química:** multigrafos (conhecidos como grafos moleculares) são usados para modelar compostos químicos. Os vértices representam átomos e as arestas representam ligações químicas entre estes átomos.

**Circuitos eletrônicos:** são modelados usando grafos em que os vértices representam os componentes e as arestas representam as ligações entre eles. A isomorfia entre grafos pode ser usada para determinar se um determinado chip de um fabricante inclui propriedade intelectual de outro fabricante.

**Computer vision:** **Investigue!**

### Alguns Grafos Simples Especiais

- **Grafos completos  $K_n$ :**

grafos simples com  $n > 0$  vértices que contêm exatamente uma aresta entre dois vértices distintos (Figura 1.8). Portanto, todos os vértices são adjacentes a todos os outros.

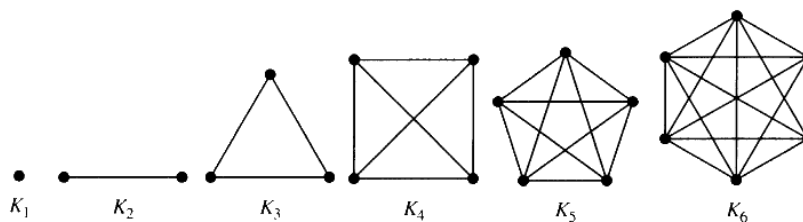


Figura 1.8:  $K_n, n = 1, 2, \dots, 6$  (retirada de (Rosen, 2012))

- **Ciclos  $C_n, n \geq 3$ :**

consistem em  $n$  vértices,  $v_1, v_2, \dots, v_n$ , e arestas  $(v_1, v_2), (v_2, v_3), \dots, (v_{n-1}, v_n), (v_n, v_1)$ .

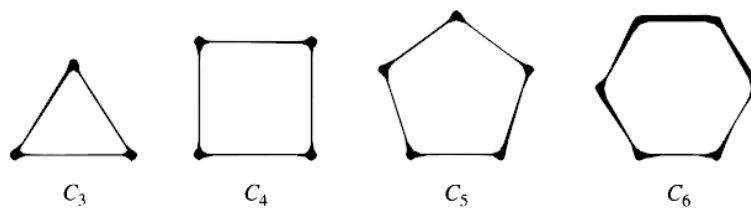


Figura 1.9:  $C_n, n = 3, \dots, 6$  (retirada de (Rosen, 2012))

- **Rodas (Wheels)**  $W_n, n \geq 4$ :

obtido a partir de um ciclo acrescentando um novo vértice que une cada um dos  $n$  vértices de  $C_n$ . (Figura 1.10).

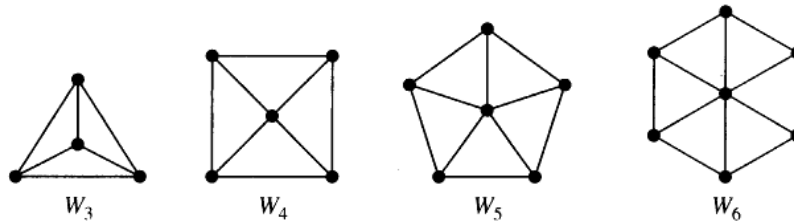


Figura 1.10:  $W_n, n = 3, \dots, 6$  (retirada de (Rosen, 2012))

- **$n$ -Cubos**  $Q_n, n = 1, 2, \dots$ :

é um grafo que tem vértices que representam  $2^n$  strings de bits de comprimento  $n$ . Dois vértices são adjacentes se e só se as strings que representam diferem apenas numa posição (Figura 1.11).

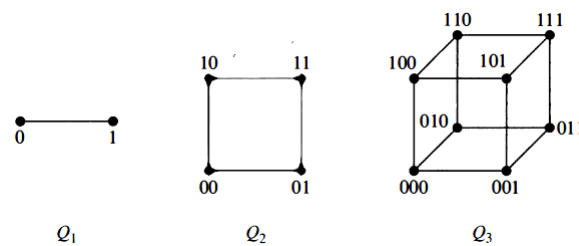


Figura 1.11:  $n$ -Cubos,  $n = 1, 2, 3$  (retirada de (Rosen, 2012))

### Definição 6:

Um grafo é chamado um **grafo bipartido completo** se o seu conjunto dos vértices  $V$  pode ser dividido em dois conjuntos, não vazios, disjuntos  $V_1$  e  $V_2$  (i.e.,  $V_1 \cap V_2 = \emptyset$  e  $V_1 \cup V_2 = V$ ), tais que:

dois vértices  $x$  e  $y$  são adjacentes se e só se  $x \in V_1$  and  $y \in V_2$ .

Se  $|V_1| = m$  e  $|V_2| = n$  denotamos esse grafo por  $K_{m,n}$ .

### Exemplo 7:

O grafo da Figura 1.12 não é completo uma vez que, nem todos os vértices são adjacentes.

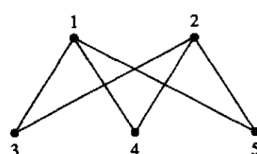


Figura 1.12: Grafo bipartido completo  $K_{2,3}$ .



Por exemplo, não existe uma aresta incidente nos vértices 1 a 2.

No entanto, podemos dividir os vértices deste grafo em dois conjuntos  $V_1 = \{1, 2\}$  e  $V_2 = \{3, 4, 5\}$  tais que dois vértices de  $V_1$  (ou  $V_2$ ) não são adjacentes e dois vértices de conjuntos  $V_1$  e  $V_2$  diferentes são adjacentes.

Ou seja,

$1 \in V_1$  não é adjacente a  $2 \in V_1$ ;

$3 \in V_2$  não é adjacente a  $4, 5 \in V_2$ ;

$4 \in V_2$  não é adjacente a  $3, 5 \in V_2$ ;

mas

$1 \in V_1$  é adjacente a  $3, 4, 5 \in V_2$  (o mesmo para  $2 \in V_1$ );

$3 \in V_2$  é adjacente a  $1, 2 \in V_1$  (o mesmo para  $4, 5 \in V_2$ ).

#### Teorema 4:

Um grafo  $G$  é bipartido se e só se não tem circuitos de comprimento ímpar.  $\square$

#### Definição 7:

Um **subgrafo** de um grafo  $G = (V(G), E(G))$  é um grafo  $H = (V(H), E(H))$  onde  $V(H) \subseteq V(G)$  e  $E(H) \subseteq E(G)$ . Se  $H$  for um subgrafo de  $G$  diferente de  $G$ , então  $H$  denomina-se de **subgrafo próprio**.

## 1.3 Conectividade

O problema de determinar se uma mensagem pode ser enviada de um computador para outro através de *links* intermédios pode ser estudada recorrendo a um grafo. O mesmo acontece para problemas de planear a recolha do lixo, o diagnóstico em redes de computadores, planear a entrega de correio, entre outros.

De seguida introduz-se alguma notação relativa a caminhos que percorrem as arestas de um grafo.

#### Caminhos

#### Definição 8:

Dado um grafo não orientado  $G = (V(G), E(G))$ , designamos por **caminho** em  $G$  toda a sequência não vazia

$$P = (v_0, e_1, v_1, e_2, v_2, \dots, e_k, v_k)$$

com  $v_0, v_1, \dots, v_k \in V(G)$ ,  $e_1, e_2, \dots, e_k \in E(G)$  e os vértices  $v_{i-1}$  e  $v_i$  são os extremos da aresta  $e_i$ , para todo  $i = 1, \dots, k$ .

- Designamos o vértice  $v_0$  de **vértice inicial**, o vértice  $v_k$  de **vértice final** e os vértices  $v_1, \dots, v_{k-1}$  de **vértices intermédios** do caminho  $P$ .
- Dizemos ainda que  $P$  é um **caminho** entre os vértices  $v_0$  e  $v_k$ .

- O caminho  $P$  designa-se por **caminho simples** se todas as arestas forem distintas.

Se o grafo  $G$  for um **grafo simples** (i.e., sem arestas múltiplas), então, a existir, existe uma única aresta de  $u$  para  $v$ , quaisquer que sejam os vértices  $u$  e  $v$ , pelo que podemos apresentar o caminho  $P$  simplesmente pela sequência dos seus vértices, ou seja,  $P = (v_0, v_1, v_2, \dots, v_k)$ .

### Definição 9:

Num grafo simples, se o vértice inicial for igual ao vértice final, designamos os conceitos anteriores, respetivamente, por **circuito** e **circuito simples**.

### Definição 10:

Seja um caminho  $P$  de um grafo  $G$ .

- Designa-se por **comprimento** de  $P$ , e denota-se por  $comp(P)$ , o número de arestas (com eventual repetição) que o constitui.
- Distância entre dois quaisquer vértices  $u$  e  $v$  de  $G$ , onde  $\mathcal{P}_{u,v}$  é o conjunto de todos os caminhos entre  $u$  e  $v$ :

$$dist_G(u, v) = \begin{cases} \min_{P \in \mathcal{P}_{u,v}} comp(P) & \text{se } \mathcal{P}_{u,v} \neq \emptyset \\ \infty & \text{se } \mathcal{P}_{u,v} = \emptyset \end{cases}$$

### Exemplo 8:

No grafo simples, apresentado na Figura 1.13:

- $(a, d, c, f, e)$  é um **caminho simples** de comprimento 4, pois as arestas são  $\{a, d\}$ ,  $\{d, c\}$ ,  $\{c, f\}$  e  $\{f, e\}$ ;
- $(a, b, e, d, a, b)$  é um **caminho** de comprimento 5, mas não é simples pois contém a aresta  $\{a, b\}$  duas vezes;
- $(d, e, c, a)$  **não é um caminho** porque não existe a aresta  $\{e, c\}$ ;
- $(b, c, f, e, b)$  é um **circuito** de comprimento 4, porque tem 4 arestas,  $\{b, c\}$ ,  $\{c, f\}$ ,  $\{f, e\}$  e  $\{e, b\}$ , e começa e termina no mesmo vértice  $b$ .

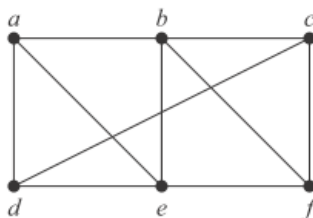


Figura 1.13: Grafo simples não orientado.

### Atenção:

Existe uma grande variabilidade na terminologia usada (Ver Apêndice 3).

**Definição 11:**

- Um grafo **não orientado** diz-se **conexo** se, para cada par de vértices, existe um caminho que os une. Caso contrário, o grafo diz-se **desconexo**.
- Um **grafo orientado** diz-se **conexo** se o seu grafo suporte (i.e., o grafo que se obtém ignorando a orientação das arestas) for conexo.
- Um **grafo orientado** diz-se **fortemente conexo** se, para cada par de vértices, existe um caminho orientado que os une.

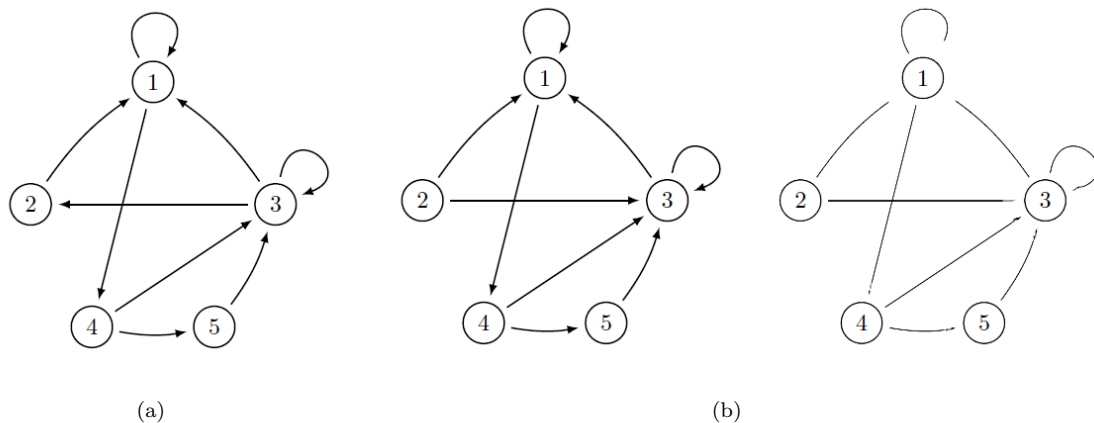


Figura 1.14: (a) Digrafo fortemente conexo e (b) digrafo conexo mas não fortemente conexo

Na Figura 1.14 são apresentados um exemplo de um grafo fortemente conexo e de um grafo conexo, mas não fortemente conexo.

De facto, no grafo à esquerda o caminho fechado  $P = (1, 4, 5, 3, 2, 1)$  passa por todos os vértices e portanto, existe um caminho entre quaisquer dois vértices.

No grafo apresentado à direita (Figura 1.14), não é possível definir nenhum caminho orientado que passe em 2, portanto o grafo não é fortemente conexo. No entanto, o seu grafo suporte é conexo, donde o grafo é conexo.

**Exemplo 9:**

O *Web graph*<sup>a</sup> não é fortemente conexo, mas tem uma componente fortemente conexa (i.e., subgrafo que é fortemente conexo mas não está contido no maior subgrafo fortemente conexo) que inclui cerca de 90% dos vértices do grafo. O subgrafo do grafo original correspondente a esta componente fortemente conexa tem uma componente fortemente conexa muito grande (*GSCC* - *giant strongly connected component*) e várias componentes fortemente conexas pequenas.

<sup>a</sup>Em 2010 para armazenar a matriz de adjacência deste grafo eram necessários 40TB de memória.

Qualquer *webpage* da GSCC pode ser atingida começando de qualquer *webpage* da GSCC (que contem mais de 53 milhões de vértices!) Os restantes vértices na componente conexa do grafo não orientado são:

páginas que podem ser atingidas a partir de qualquer página na GSCC mas não apontam de volta a estas páginas; páginas que apontam de volta para páginas no GSCC, mas não pode ser alcançadas seguindo os links nas páginas da GSCC; e páginas que não podem alcançar páginas na GSCC e não pode ser alcançadas a partir de páginas na GSCC (após uma série de links).

Pode encontrar mais pormenores em [4].

O número de caminhos entre dois vértices de um grafo pode ser determinado recorrendo à matriz de adjacências.

### Teorema 5:

Seja  $G$  um grafo com matriz de adjacências  $M$ , com respeito à ordem dos vértices  $v_1, v_2, \dots, v_n$ . O número de diferentes caminhos de comprimentos  $c$  de  $v_i$  para o  $v_j$  é igual a  $a_{ij}$  da matriz  $M^c$ .  $\square$

### Observação:

O **Teorema 5** é válido para grafos não orientados, digrafos, grafos com arestas múltiplas ou lacetes.

### Exemplo 10:


Dado o grafo da Figura 1.15 cuja matriz de adjacências é:

$$M = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

o número de caminhos de comprimento 4, entre dois quaisquer vértices é dado pela matriz  $M^4$ ,

$$M^4 = \begin{bmatrix} 6 & 3 & 5 & 2 & 1 \\ 2 & 1 & 3 & 1 & 1 \\ 8 & 2 & 6 & 5 & 3 \\ 8 & 2 & 3 & 4 & 1 \\ \mathbf{5} & 1 & 2 & 3 & 1 \end{bmatrix}$$

Por exemplo, existem 5 caminhos de comprimentos 4 do vértice 5 para o vértice 1.

**Observação:** Para o cálculo de  $M^4$  use o  Scilab.

Para verificar se um grafo é fortemente conexo podemos recorrer ao **fecho transitivo** da matriz de adjacências de um grafo.

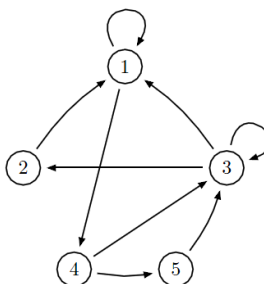


Figura 1.15: Grafo orientado

**Definição 12:**

O **fecho transitivo** de uma matriz de adjacências  $M$ , de ordem  $n$ , de um grafo  $G$  é a matriz  $F$ :

$$F = M + M^2 + \cdots + M^n.$$

**Corolário 1:**

Um grafo não orientado é conexo (fortemente conexo) se o fecho transitivo da sua matriz de adjacências não tiver entradas nulas.

Um grafo orientado é fortemente conexo se o fecho transitivo da sua matriz de adjacências não tiver entradas nulas.  $\square$

O grafo do exemplo anterior é fortemente conexo uma vez que,

$$F = M + M^2 + M^3 + M^4 + M^5 = \begin{bmatrix} 24 & 9 & 17 & 11 & 5 \\ 11 & 4 & 9 & 5 & 3 \\ 33 & 11 & 24 & 17 & 9 \\ 26 & 8 & 16 & 13 & 6 \\ 17 & 5 & 11 & 9 & 4 \end{bmatrix},$$

não tem entradas nulas.

## 2 Caminhos Eulerianos e Hamiltonianos

Os habitantes de Königsberg gostavam de dar passeios de modo a atravessar todas as pontes, sendo um passatempo tentar encontrar um caminho, com partida e chegada ao mesmo lugar, que lhes permitisse atravessar apenas uma vez cada uma das pontes (ver Figura 1.16).

Leonhard Euler (1707-1783) modelou o problema usando um grafo e mostrou a impossibilidade da existência de um tal percurso.

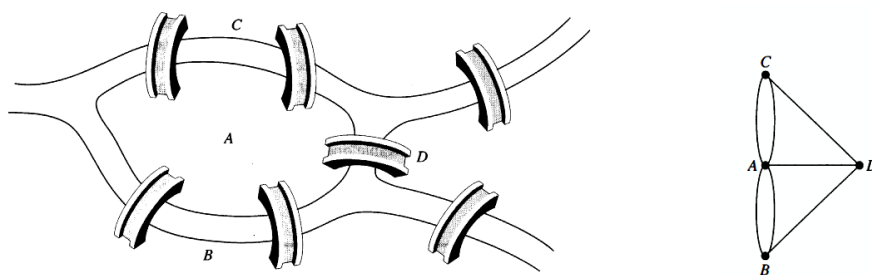


Figura 1.16: As sete pontes de Königsberg e o respetivo multigrafo (retirado de [4])

### Definição 13:

Um **circuito de Euler** num grafo  $G$  é um circuito que contém todas as arestas de  $G$ .

Um **caminho de Euler** num grafo  $G$  é um caminho simples que contém ~~cada uma das~~ arestas de  $G$ .

Um grafo diz-se **Euleriano** (ou um grafo de Euler) se admite um circuito de Euler e diz-se **semi-Euleriano** se admite um caminho de Euler.

~~Um grafo Euleriano é também semi-Euleriano!~~

simples

, mas não admite um circuito,

### Como podemos verificar a existência caminhos e circuitos de Euler?

O seguinte resultado fornece as condições suficientes e necessárias para a existência de circuitos de Euler.

### Teorema 6:

Um multigrafo conexo, com pelos menos dois vértices, tem um circuito euleriano **se e só se** todos os seus vértices têm grau par.  $\square$

Com este resultado podemos resolver o problema das sete pontes de Königsberg. Visto que o multigrafo para este problema, apresentado na Figura 1.16, tem vértices com grau ímpar podemos concluir que o grafo não é Euleriano, ou seja não existem circuitos de Euler. Portanto não era possível atravessar cada uma das sete pontes exatamente uma vez com ponto de partida e chegada no mesmo local.

Mas, será possível atravessar cada uma das pontes uma única vez começando e terminando em locais diferentes? O seguinte resultado ajuda-nos a responder a esta questão uma vez que fornece condições suficientes e necessárias para a existência de caminhos de Euler mas que não sejam circuitos de Euler.

### Teorema 7:

Um multigrafo conexo tem um caminho euleriano, mas não tem um circuito euleriano, **se e só se** possuir exatamente dois vértices de grau ímpar.  $\square$

O grafo que modela o problema das pontes de Königsberg **não é nem Euleriano, nem semi-Euleriano**, uma vez que possui quatro vértices de grau ímpar. Portanto, o problema de encontrar um caminho que passe uma única vez por todas as pontes é impossível!

**Teorema 8:**

Um multigrafo **orientado** admite um **circuito de Euler** se e só se:

- (1) for conexo;
- (2) cada vértice tem o mesmo grau de entrada e saída.

□

**Teorema 9:**

Um multigrafo **orientado** admite um **caminho euleriano**, mas não tem um circuito de Euler, se e só se:

- (1) for conexo;
- (2) todos os vértices têm o mesmo grau de entrada e saída, excepto e dois vértices que têm graus de entrada e de saída que diferem de 1, sendo que um dos vértices tem um grau de entrada a mais do que o grau de saída e o outro vértice tem um grau de saída a mais do que o grau de entrada.

□

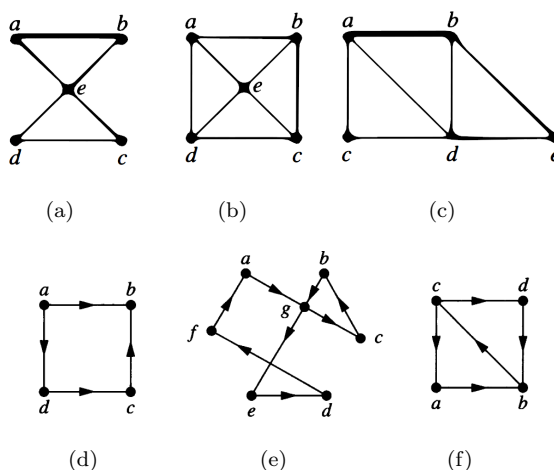


Figura 1.17: O grafo apresentado em: (a) é Euleriano; (b) não tem circuitos nem caminhos de Euler; (c) não tem circuitos de Euler mas tem caminhos de Euler. O grafo orientado apresentado em: (d) não tem circuitos nem caminhos de Euler; (e) tem circuitos de Euler; (f) não tem circuitos de mas tem caminhos de Euler.

Na Figura 1.17 são apresentados alguns exemplos de grafos, orientados e não orientados. O grafo apresentado em (a) tem circuitos de Euler (Por exemplo,  $a, e, c, d, e, b, a$ ). Por outro lado, o grafo apresentado em (b) não tem circuitos nem caminhos de Euler, pois tem quatro vértices de grau ímpar. O grafo apresentado em (c) não tem circuitos de Euler pois nem todos os seus vértices têm grau par. No entanto, como os vértices  $a$  e  $b$  que têm grau ímpar, este grafo tem caminhos de Euler (Por exemplo,  $a, c, d, e, b, d, a, b$ ). Relativamente aos grafos orientados apresentados na Figura 1.17 (d), (e) e (f), temos que todos são grafos fortemente conexos (**Verifique-o!**). O vértice  $b$  do grafo apresentado em (d) tem

grau de saída zero o grau de entrada dois, i.e.  $\text{grau}^s(b) - \text{grau}^e(b) = 2 - 0 = 2 \neq 1$ , pelo que (d) não tem circuitos nem caminhos de Euler. Todos os vértices do grafo apresentado em (e) grau de entrada e saída igual a um, portanto, o grafo tem circuitos de Euler (Por exemplo,  $a, g, c, b, g, e, d, j, a$ ). Finalmente, o grafo apresentado em (f) não tem circuitos de Euler mas tem caminhos de Euler (Por exemplo,  $c, a, b, c, d, b$ ), uma vez que  $\text{grau}^s(a) = \text{grau}^e(a)$  e  $\text{grau}^s(d) = \text{grau}^e(d)$ , e os dois outros vértices são tais que  $\text{grau}^s(c) - \text{grau}^e(c) = 2 - 1 = 1$  e  $\text{grau}^e(b) - \text{grau}^s(b) = 2 - 1 = 1$ .

O Algoritmo de **Fleury**, apresentado de seguida no Algoritmo 1, permite obter um circuito ou um caminho de Euler. Em <https://www.youtube.com/watch?v=vvP4Fg4r-Ns> (e também na página do *moodle* desta Unidade Curricular) pode encontrar um vídeo exemplificativo deste algoritmo.

*Input:* grafo  $G$ .

*Output:* circuito ou caminho de Euler  $\varepsilon$ , caso exista.

- (1) Determinam-se os graus de todos os vértices.
- (2) Se o grafo tiver um número de vértices de grau ímpar diferente de 2 ou 0, então o algoritmo pára e dá como resposta “O grafo não é Euleriano nem semi-Euleriano”.
- (3) Se houver 2 vértices de grau ímpar, escolhe-se um deles.  
Se não, escolhe-se qualquer vértice. Seja  $v$  o vértice escolhido e  $\varepsilon = v$ .
- (4) Se não houver nenhuma aresta incidente em  $v$ , então o algoritmo pára e devolve o caminho de Euler  $\varepsilon$ .  
Caso contrário, se houver apenas uma aresta, escolhe-se essa aresta e removem-se essa aresta e o vértice do grafo; se houver mais do que uma aresta incidente no vértice, escolhe-se uma aresta que, ao ser removida, não desconecte o grafo; remove-se essa aresta. Seja  $e = (v, w)$  a aresta escolhida.
- (5) Adiciona-se  $ew$  como sufixo de  $\varepsilon$ .
- (6) Substitui-se  $v$  por  $w$  e regressa-se a (4).

#### Algorithm 1: Algoritmo de Fleury

#### Aplicações:

Circuitos e caminhos de Euler podem ser utilizados para resolver diversos problemas de ordem prática, como por exemplo: aplicações que pedem um caminho que percorre todas as ruas de um bairro, cada rua numa rede de transporte, cada conexão numa rede elétrica, ou cada link de um sistema de comunicação rede apenas uma vez.

Circuitos e caminhos de Euler podem também ser usados no *layout* de circuitos, em *multicast* de redes, ou em biologia molecular (onde caminhos de Euler são usados para determinar sequências de DNA).



Um problema clássico que pode ser resolvido procurando um circuito Euleriano é o **Problema do Carteiro Chinês** (*Chinese postman problem*). Este problema consiste em encontrar um circuito que percorra todas as ruas pelo menos uma vez com um custo mínimo. Se existir um circuito Euleriano no grafo representativo das ruas que o carteiro tem de visitar então essa é a solução ótima deste problema e o carteiro percorre cada rua exatamente uma vez. No entanto, quando não existe um circuito Euleriano algumas ruas têm de ser percorridas mais do que uma vez. Para mais informações ver, por exemplo, [4], [1].

#### Exemplo 11:

O Puzzle “Viagem à Volta do Mundo” ou **Dodecaedro do viajante** foi inventado por Hamilton em 1857. O objetivo é começar numa cidade e viajar ao longo das arestas do dodecaedro visitando cada uma das 19 cidades uma única vez e terminar a viagem na cidade inicial. Cada um dos 20 vértices representa uma das 20 cidades (ver Figura 1.18).

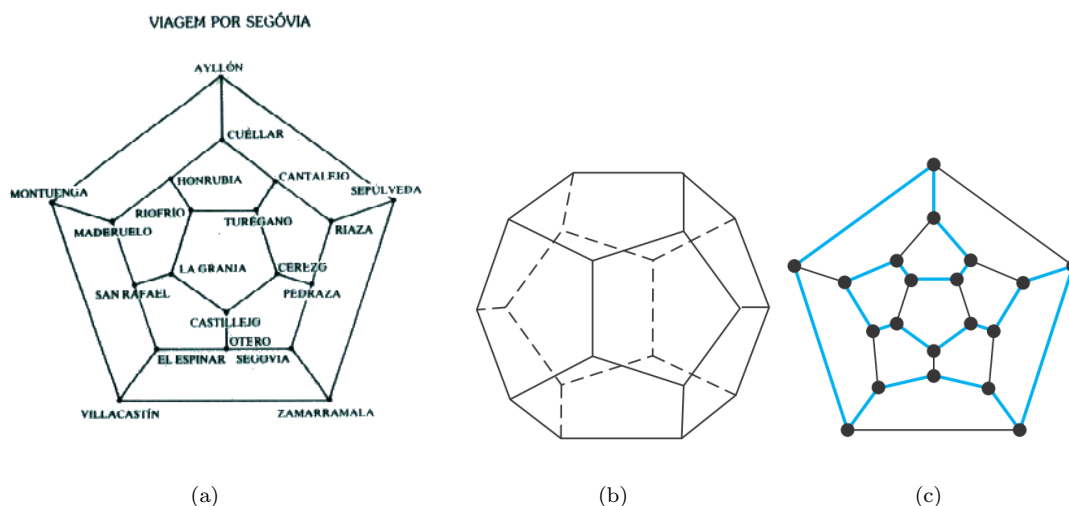


Figura 1.18: (a) Viagem à volta do mundo de Hamilton (imagem retirada de: <http://www.prof2000.pt/users/miguel/grafos/jogham.htm>); (b) dodecaedro; (c) uma solução do problema.

#### Definição 14:

Um **circuito de Hamilton** é um circuito num grafo que passa em cada vértice exatamente uma vez. Um **caminho de Hamilton** é um caminho num grafo que passa em cada vértice exatamente uma vez. Um **grafo Hamiltoniano** (ou grafo de Hamilton) é um grafo que admite um circuito de Hamilton. Um **grafo semi-Hamiltoniano** é um grafo que admite um caminho de Hamilton.

Apesar da semelhança entre caminhos Hamiltonianos e caminhos Eulerianos **não se conhecem condições necessárias e suficientes para que um grafo seja Hamiltoniano**. Existem, no entanto, alguns Teoremas que fornecem condições suficientes para a existência de circuitos de Hamilton.

Existem também algumas propriedades que podem ser utilizadas para mostrar que um grafo **não possui um circuito de Hamilton**:

- um grafo com um vértice de grau 1 não tem um circuito de Hamilton, isto porque cada vértice é incidente em duas arestas no circuito;
- se o vértice de um grafo tem grau 2 então as duas arestas incidentes nesse vértice devem fazer parte de todo o circuito de Hamilton;
- durante a construção de um circuito de Hamilton, se o circuito passou por um vértice, então as arestas incidentes nesse vértice podem ser removidas;
- um circuito de Hamilton não pode conter um circuito menor.

De seguida são apresentadas **duas das mais usadas condições suficientes para a existência de um circuito de Hamilton num grafo**.

#### Teorema 10:

##### Teorema (de DIRAC)

Seja  $G$  um grafo simples com  $n$  vértices,  $n \geq 3$ , tal que o grau de cada vértice é pelo menos  $n/2$ ,

então  $G$  tem um circuito de Hamilton.

□

#### Teorema 11:

##### Teorema (de ORE)

Seja  $G$  um grafo simples com  $n$  vértices,  $n \geq 3$ , tal que  $\text{grau}(u) + \text{grau}(v) \geq n$  para cada par de vértices  $u$  e  $v$  não adjacentes,

então  $G$  tem um circuito de Hamilton.

□

**Atenção** que os dois Teoremas anteriores apenas fornecem condições suficientes.

Podem existir circuitos de Hamilton em grafos para os quais as condições especificadas pelos Teoremas de Dirac e Ore não se verifiquem.

Por exemplo,  $C_5$  apresentado na Figura 1.9, página 11, tem um circuito de Hamilton mas:

- o grau de cada vértice é 2 que é menor que  $n/2 = 5/2$ , portanto, não estão satisfeitas as condições suficientes dadas no Teorema de Dirac;
- para qualquer par de vértices  $u$  e  $v$  de  $C_5$  temos que  $\text{grau}(u) + \text{grau}(v) = 2 + 2 = 4 < 5$ , portanto, não estão satisfeitas as condições suficientes dadas no Teorema de Ore.

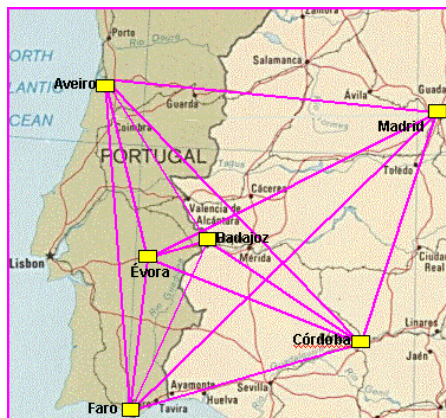


Figura 1.19: Problema do Caixeiro Viajante (retirado de <http://users.prof2000.pt/agnelo/grafos/tsp.htm>)

### Teorema 12:

#### de DIRAC (para grafos orientados)

Seja  $G$  um grafo orientado simples com  $n$  vértices, tal que o grau de saída e o grau de entrada de cada vértice é pelo menos  $n/2$ ,

então  $G$  tem um circuito de Hamilton.

□

### Teorema 13:

#### de ORE (para grafos orientados)

Seja  $G$  um grafo orientado simples com  $n$  vértices, tal que  $\text{grau}^s(u) + \text{grau}^e(v) \geq n$  para cada par de vértices  $u$  e  $v$  não adjacentes,

então  $G$  tem um circuito de Hamilton.

□

## Aplicações

Circuitos e caminhos de Hamilton podem ser utilizados numa grande variedade de problemas práticos, tais como, problemas em que se pretende encontrar um caminho ou um circuito que passa: em todos os cruzamentos de uma cidade, ou em todas os nós de uma rede exatamente uma vez.

Um problema famoso é o **problema do caixeiro viajante** (*TSP - Traveling Salesperson Problem*) cujo objetivo é encontrar o caminho mais curto que passe por um conjunto de cidades. Este problema reduz-se a encontrar um circuito de Hamilton num grafo tal que o peso das suas arestas (que pode ser a distância entre as duas cidades) seja o menor possível (Figura 1.19).

De facto, os avanços mais importantes da Teoria dos Grafos têm sido quase sempre motivados, pela tentativa de resolução de problemas práticos muito específicos. Por exemplo, problema das pontes de Königsberg, o problema do caminho mais curto, o problema do caixeiro viajante, o problema do carteiro chinês.

## Software

Um exemplo de um software para Teoria de Grafos é a toolbox do Scilab **metanet** disponível em <http://forge.scilab.org/index.php/p/metanet/>. Para a instalação ver [http://wiki.scilab.org/ATOMS#Install\\_a\\_module](http://wiki.scilab.org/ATOMS#Install_a_module).

Outro exemplo é o GraphTea <http://www.graphtheorysoftware.com/>.



## O problema do caminho mais curto

### Definição 15:

Consideremos que  $G$  é um **grafo ponderado**, i.e. um grafo que a cada aresta,  $e$ , é associado valor não negativo,  $w(e)$ , a que dá-nos o nome de peso ou custo. Podemos representar os pesos de cada arestas recorrendo à chamada **matriz de pesos** (ou matriz de custos),  $W = (w_{ij})$ , definida por:

$$w_{ij} = \begin{cases} w(e), & \text{se é a aresta do vértice } v_i \text{ para o vértice } v_j \\ 0, & \text{caso não exista uma aresta do vértice } v_i \text{ para o vértice } v_j \end{cases}$$

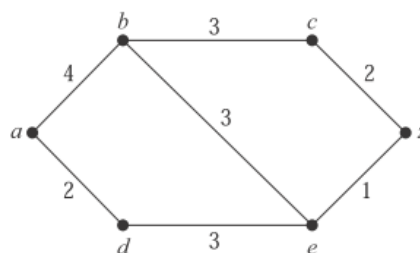


Figura 1.20: Retirada de [4].

**Exemplo 12:**

Consideremos o grafo ponderado apresentado na Figura 1.20.

Temos que  $V = \{a, b, c, d, e, z\}$ ,  $E = \{(a, b), (a, d), (b, c), (b, e), (d, e), (c, z), (e, z)\}$  e a matriz de pesos é:

$$W = \begin{bmatrix} 0 & 4 & 0 & 2 & 0 & 0 \\ 4 & 0 & 3 & 0 & 3 & 0 \\ 0 & 3 & 0 & 0 & 0 & 2 \\ 2 & 0 & 0 & 0 & 3 & 0 \\ 0 & 3 & 0 & 3 & 0 & 1 \\ 0 & 0 & 2 & 0 & 1 & 0 \end{bmatrix}$$

**Definição 16:**

O peso ou custo de um caminho é a soma dos pesos de todas as arestas que o formam.

Para o Exemplo 12 temos que:

- o peso ou custo do caminho  $a, b, e$  é  $4 + 3 = 7$
- o peso ou custo do caminho  $a, d, e$  é  $2 + 3 = 5$
- dos dois caminhos com início em  $a$  e término em  $e$ , o caminho  $a, d, e$  é o que apresenta menor custo.

**Note** que um grafo não ponderado pode ser interpretado como um grafo com todos os custos iguais a 1. Pelo que, todos os algoritmos desenvolvidos para grafos ponderados podem ser utilizados para grafos não ponderados.

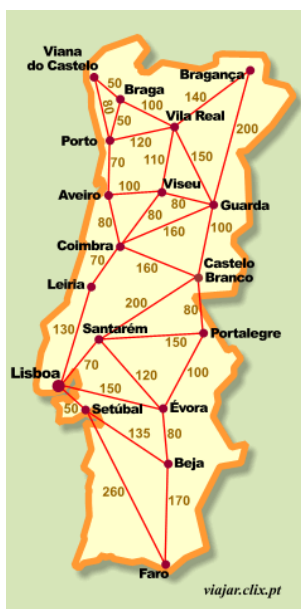


Figura 1.21: Retirada de <http://viajar.clix.pt/>.

Consideremos que um motorista deseja encontrar o caminho, mais curto possível, entre as cidades de Bragança e Faro (ver Figura 1.21). Este problema pode ser resolvido recorrendo a grafos. Considerando

que cada estrada representa uma aresta e os vértices são as cidades por onde passam (ou começam ou terminam) essas estradas, vamos atribuir a cada aresta um valor (não negativo). Neste exemplo este valor corresponde à distância entre dois vértices<sup>1</sup>. Temos, desta forma, um grafo ponderado.

Um algoritmo clássico para resolver este problema (i.e. o problema do caminho mais curto), para **grafos não orientados com pesos positivos**, é o **Algoritmo de Dijkstra**<sup>2</sup> apresentado no Algoritmo ???. Este é um algoritmo *Greedy* (guloso), de complexidade polinomial (ver Teorema 15) que é facilmente adaptado para grafos orientados. Para grafos com pesos negativos pode-se utilizar, para alguns casos, o algoritmo de **Bellman-Ford**.

O algoritmo inicialmente proposto por Dijkstra<sup>3</sup> permitia encontrar o caminho mais curto entre dois vértices. No entanto, **existem diversas variantes deste algoritmo**. Uma das mais comuns permite encontrar o caminho mais curto de um dado vértice inicial a cada um dos restantes vértices do grafo.

Dijkstra, teve em consideração que, para um vértice  $v_j$  qualquer, pertencente ao caminho mais curto de  $v_i$  para  $v_f$ , i.e.  $v_i, v_1, v_2, \dots, v_j, v_k, \dots, v_f$ , o caminho mais curto de  $v_j$  para  $v_f$  será  $v_j, v_k, \dots, v_f$ , contido no caminho anterior. O algoritmo de Dijkstra é descrito do seguinte modo:

*Input:* grafo simples, não orientado e ponderado  $G = (V, E)$ ; matriz de pesos; vértice inicial  $a = v_i$  e vértice final  $z = v_f$ .

**1. Inicializar os conjuntos:**

$$M = \emptyset$$

$X$  com os vértices adjacentes ao vértice inicial  $v_i$ ;

$X_d$  com as distâncias (ou custos ou pesos) de  $v_i$  a todos os vértices de  $X$ ;

$R$  com todos os caminhos que ligam  $v_i$  ao vértices de  $X$ .

**2. Selecionar** o vértice  $v_d$  correspondente ao elemento mínimo de  $X$ ;

**Transferir** o vértice  $v_d$  para o conjunto  $M$ ;

**Transferir** o caminho mais curto entre  $v_i$  e  $v_d$  para o conjunto  $Mc$ .

**3.** Se  $v_i = v_d$  então **TERMINAR** o algoritmo

**4. Selecionar** os vértices adjacentes de  $v_d$  não pertencentes a  $M$ .

Considerar  $A$  como o conjunto constituído por esses vértices.

4.1. **Determinar** as distâncias de  $v_i$  até cada um dos vértices do conjunto  $A$ , somando à distância de  $v_i$  a  $v_d$  as distâncias de  $v_d$  a esses vértices.

4.2. **Introduzir**

no conjunto  $X$  os vértices do conjunto  $A$  que não pertençam a  $X$ ;

no conjunto  $X_d$  as distâncias de  $v_i$  a esses vértices;

no conjunto  $R$  os respetivos caminhos.

4.3. Se houver, no conjunto  $A$ , vértices que já pertenciam ao conjunto  $X$ , mas com distâncias em  $X_d$  superiores às agora determinadas, então, para cada um desses vértices,  $v_e$ , **substituir**:

<sup>1</sup>Também poderia corresponder, por exemplo, ao custo para ir de um vértice a outro ou o tempo gasto.

<sup>2</sup>Este algoritmo foi descoberto em 1959 pelo Holandês Edsger

<sup>3</sup>Ver em <http://www-m3.ma.tum.de/foswiki/pub/MN0506/WebHome/dijkstra.pdf>.

no conjunto  $Xd$  a distância associada ao vértice  $v_e$  pela nova distância;

no conjunto  $R$ , o caminho de  $v_i$  para  $v_e$ , pelo novo caminho  $v_i, \dots, v_d, v_e$ .

Note que  $v_i, \dots, v_d$  representa o caminho mais curto de  $v_i$  para  $v_d$ .

5. Se no conjunto  $X$  estiver apenas o vértice  $v_f$ , então **TERMINAR**, senão, voltar a 2.

Vamos utilizar o Algoritmo de Dijkstra para o caso em que se pretende encontrar o caminho mais curto entre os vértices  $v_i = a$  e  $v_f = z$  para o grafo apresentado na Figura 1.20. Vamos recorrer a uma tabela para registar as diferentes iterações deste algoritmo, sendo que em cada iteração:

- $Mc$  é o caminho mínimo de  $a$  a  $v_d$ ;
- $M$  é conjunto dos vértices já explorados, ou seja, é o conjunto de todos os  $v_d$  obtidos até essa iteração;
- $v_d$  é tal que  $L(v_d)$  é o caminho  $v_i, \dots, v$ , com  $v \in X$  de custo mínimo;
- $A$  é o conjunto dos vértices adjacentes a  $v_d$ , não pertencentes a  $M$ ;
- $v_i, \dots, v_d, v_j$  é o caminho de  $v_i = a$  a cada  $v_j \in A$ ;
- $X$  é o conjunto de todos os vértices que não  $v_d$  para os quais se conhecem caminhos mínimos a partir de  $a$ ;
- $R$  é o conjunto dos caminhos mínimos de  $v_i = a$  a cada  $v \in X$

It.	$v_d$ ( $M$ )	$Mc$	$A$	$v_i, \dots, v_d, v_j$ e $L(v_j)$	$X$ e $Xd$	$R$
0		$a$	$\{b, d\}$	$a, b \mapsto L(b) = 4$ $a, d \mapsto L(d) = 2$	$\{d, b\}$ $\{2, 4\}$	$a, b$ $a, d$
1	$d$	$a, d$	$\{e\}$	$a, d, e \mapsto L(e) = 2 + 3 = 5$	$\{b, e\}$ $\{4, 5\}$	$a, b$ $a, d, e$
2	$b$	$a, b$	$\{c, e\}$	$a, b, c \mapsto L(c) = 4 + 3 = 7$ $a, b, e \mapsto L(e) = 4 + 3 = 7$	$\{e, c\}$ $\{5, 7\}$	$a, d, e$ $a, b, c$
3	$e$	$a, d, e$	$\{z\}$	$a, d, e, z \mapsto L(z) = 5 + 1 = 6$	$\{z, c\}$ $\{6, 7\}$	$a, b, c$ $a, d, e, z$

O caminho mínimo é  $a, d, e, z$  e tem um custo de 6.

A resolução do exemplo anterior é também apresentada na Figura 1.22 usando uma versão que em que os custos e os caminhos mínimos obtidos em cada iteração são representados no grafo.

#### Teorema 14:

**Algoritmo de Dijkstra** produz o custo de um caminho mais curto entre dois vértices num grafo simples, ponderado, conexo e não orientado.  $\square$

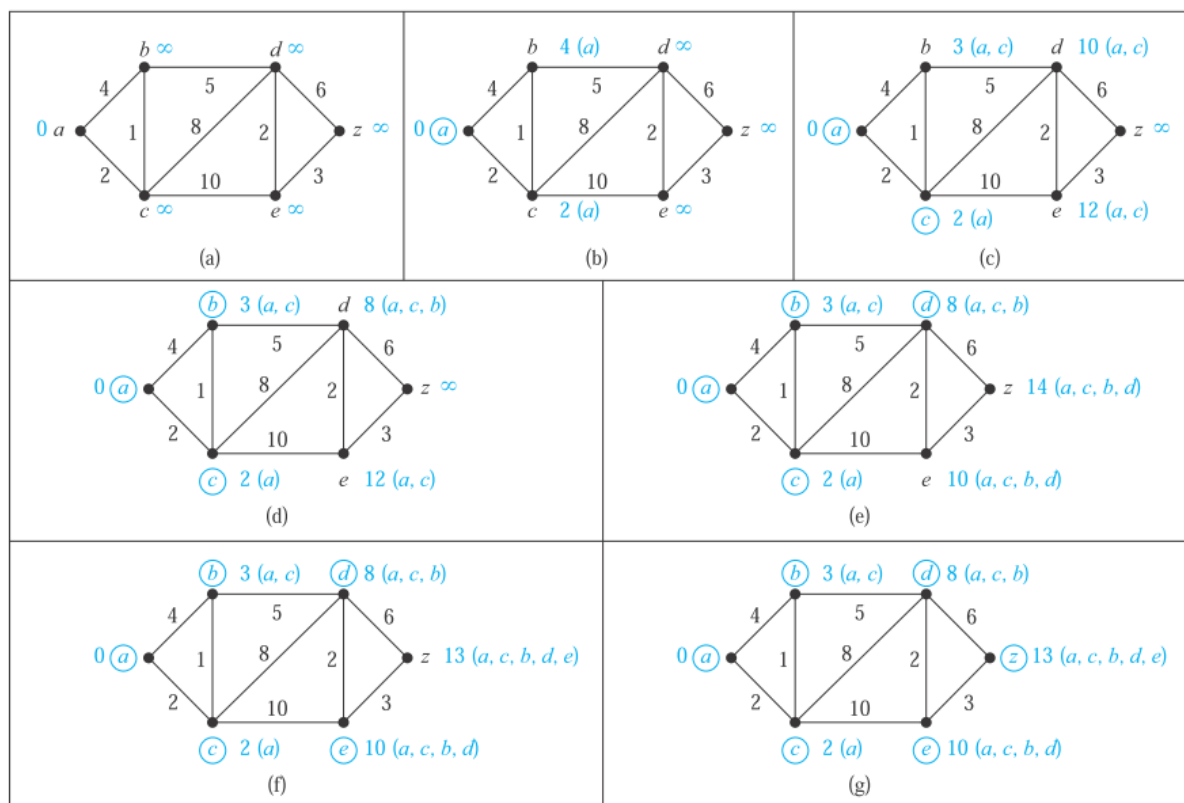


Figura 1.22: Retirada de [4].

**Teorema 15:**

O **Algoritmo de Dijkstra** usa  $\mathcal{O}(n^2)$  operações (adições e comparações) para encontrar o custo do caminho mais curto entre dois vértices num grafo simples, ponderado, conexo e não orientado, com  $n$  vértices.  $\square$

Demonstrações dos dois Teoremas anteriores podem ser encontrados por exemplo em [4].

Vamos de seguida resolver o problema de encontrar o caminho mais curto de  $a$  a  $z$  para o grafo orientado apresentado na Figura 1.23.

Na tabela abaixo são apresentados as diferentes iterações do algoritmo de Dijkstra de onde podemos concluir que o caminho mais curto é de  $a$  a  $z$  é  $a, d, g, z$  e tem custo 18.

It.	$v_d$ ( $M$ )	$Mc$	$A$	$v_i, \dots, v_d, v_j$ e $L(v_j)$	$X$ e $Xd$	$R$
0	—	$a$	$\{b, c, d\}$	$a, b \mapsto 6$ $a, c \mapsto 9$ $a, d \mapsto 5$	$\{d, b, c\}$ $\{5, 6, 9\}$	$a, b$ $a, c$ $a, d$



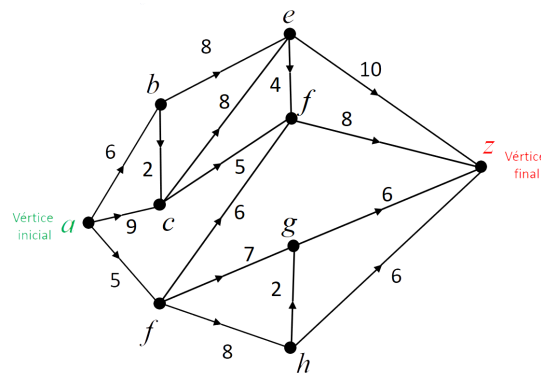


Figura 1.23: Grafo orientado.

It.	$v_d$ ( $M$ )	$Mc$	$A$	$v_i, \dots, v_d, v_j \in L(v_j)$	$X$ e $Xd$	$R$
1	$d$	$a, d$	$\{f, g, h\}$	$a, d, f \mapsto 5 + 6 = 11$ $a, d, g \mapsto 5 + 7 = 12$ $a, d, h \mapsto 5 + 8 = 13$	$\{b, c, f, g, h\}$ $\{6, 9, 11, 12, 13\}$	$a, b$ $a, c$ $a, d, f$ $a, d, g$ $a, d, h$
2	$b$	$a, b$	$\{c, e\}$	$a, b, c \mapsto 6 + 2 = 8$ $a, b, e \mapsto 6 + 8 = 14$	$\{c, f, g, h, e\}$ $\{8, 11, 12, 13, 14\}$	$a, b, c$ $a, b, e$ $a, d, f$ $a, d, g$ $a, d, h$
3	$c$	$a, c$	$\{e, f\}$	$a, c, e \mapsto 8 + 8 = 16$ $a, c, f \mapsto 8 + 5 = 13$	$\{f, g, h, e\}$ $\{11, 12, 13, 14\}$	$a, b, e$ $a, d, f$ $a, d, g$ $a, d, h$
4	$f$	$a, d, f$	$\{z\}$	$a, d, f, z \mapsto 11 + 8 = 19$	$\{g, h, e, z\}$ $\{12, 13, 14, 19\}$	$a, b, e$ $a, d, f, z$
5	$g$	$a, d, g$	$\{z\}$	$a, d, g, z \mapsto 12 + 6 = 18$	$\{h, e, z\}$ $\{13, 14, 18\}$	$a, b, e$ $a, d, g, z$ $a, d, h$
6	$h$	$a, d, h$	$\{z\}$	$a, d, h, z \mapsto 13 + 6 = 19$	$\{e, z\}$ $\{14, 18\}$	$a, b, e$ $a, d, g, z$ $a, d, g$
7	$e$	$a, b, e$	$\{z\}$	$a, b, e, z \mapsto 14 + 10 = 24$	$z$ $\{18\}$	$a, d, g, z$

### 3 Árvores e suas aplicações

Todos estamos familiarizados com a ideia árvore genealógica. Estas são grafos cujos vértices representam membros de uma família e as arestas representam o grau de parentesco. A este tipo de grafos dá-nos o nome de **árvores**.

#### Definição 17:

**Árvores** (*Trees*) são grafos simples, não orientados, conexos que não contêm circuitos simples.

**Florestas** (*Forests*) são grafos não conexos que não contêm circuitos simples, i.e., são coleções não conexas de árvores ou dito de outro modo, são grafos não conexos cujas componentes conexas são árvores.

Na Figura 1.24 temos que:

- (a) e (b) são **árvores** pois são grafos conexos que não contêm circuitos simples;
- (c) **não é árvore** pois contém, por exemplo, o circuito simples  $e, b, a, d, e$ ;
- (d) **não é árvore** pois não é um grafo conexo, por exemplo, não existe nenhum caminho de  $a$  para  $e$ , é uma **floresta**.

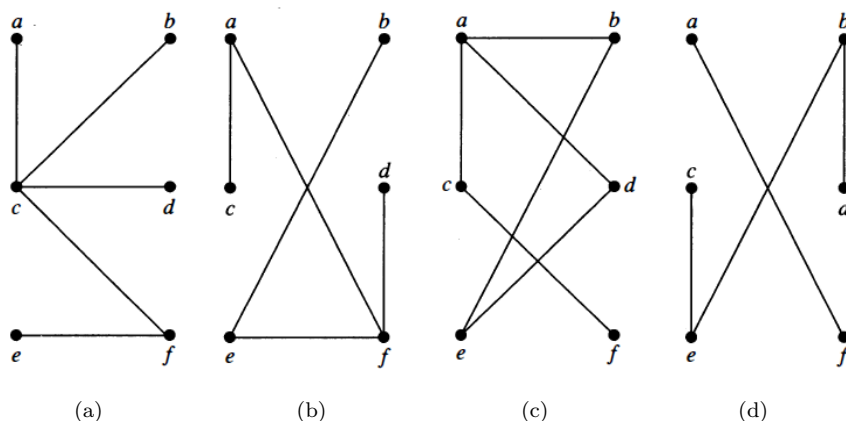


Figura 1.24: Árvores e Florestas

De seguida são apresentadas algumas condições necessárias e suficientes para um grafo ser uma árvore.

#### Teorema 16:

Seja  $G = (V, E)$  um grafo simples com  $v$  vértices.

Então,  $G$  é uma árvore se e só se  $G$  não contém circuitos simples e tem  $v - 1$  arestas.  $\square$

#### Definição 18:

Designa-se por *vértice de corte* (ou ponto de articulação), um vértices que ao ser removido de um grafo (assim como as arestas nele incidem) produz um subgrafo com um maior número de componentes conexas.

Uma aresta que ao ser removida de um grafo produz um subgrafo com um maior número de componentes conexas designa-se por *aresta de corte* ou *ponte*.

**Teorema 17:**

Seja  $G = (V, E)$  um grafo simples com  $v$  vértices, então são equivalentes as seguintes afirmações:

- $G$  é uma árvore;
- $G$  não contém circuitos e tem  $v - 1$  arestas;
- $G$  é conexo e tem  $v - 1$  arestas;
- $G$  é conexo e cada aresta é uma ponte;
- Existe um único caminho simples entre qualquer par de vértices de  $G$ ;
- $G$  não contém circuitos, mas acrescentando uma aresta obtém-se um circuito.

□

**Exercício:**

Use o Teorema anterior para verificar quais dos grafos apresentados na Figura 1.24 são árvores.

Em muitas aplicações, há um vértice que designamos por **raiz**,  $r$ . Após especificarmos uma raiz podemos atribuir uma direção a cada aresta. Como o caminho entre a raiz e cada vértice no grafo é único, direcionamos cada aresta usando esses caminhos. Dizemos assim que temos uma **árvore com raiz  $r$** . Escolhendo uma raiz diferente obtemos outra árvore com raiz (ver Figura 1.25).

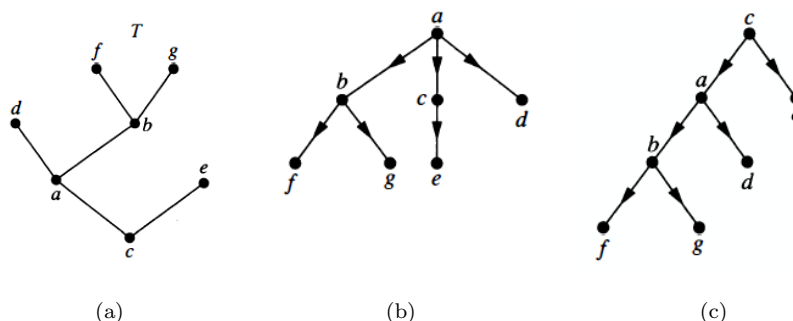


Figura 1.25: (a) Árvore  $T$ . (b) Árvore com raiz em  $a$ . (c) Árvore com raiz em  $c$ .

De seguida é apresentada a terminologia para árvores (que se inspira na botânica e na genealogia).

Seja  $G$  uma árvore com raiz  $r$ .

- Se  $v$  é um vértice diferente de  $r$ , o **pai** de  $v$  é o único vértice  $u$  para o qual existe uma aresta orientada de  $u$  para  $v$ . Nesse caso,  $v$  diz-se um **filho** de  $u$ .
- Vértices com o mesmo pai são chamados de **irmãos**.
- Os **ascendentes** de um vértice  $w$ , que não a raiz, são os vértices no caminho da raiz a esse vértice, i.e., o pai de  $w$ , o pai do seu pai e assim sucessivamente até atingir a raiz.
- Os **descendentes** de um vértice  $v$  são todos os vértices que têm como ascendente  $v$ .

- Um vértice de  $G$  é uma **folha** se não tiver filhos.
- Os vértices que têm filhos dizem-se **vértices internos**.
- Se  $a$  é um vértice de uma árvore, a **sub-árvore** com raiz  $a$  é o subgrafo da árvore que constituído por  $a$  e pelos seus descendentes, e todas as arestas incidentes nesses descendentes (ver, por exemplo Figura 1.27).
- A **profundidade do vértice** numa árvore é o comprimento do caminho da raiz até ao vértice. Em particular, a raiz tem profundidade 0.
- A **altura da árvore** é a maior profundidade de todos os seus vértices, é o comprimento do maior caminho entre a raiz e um vértice.

### Definição 19:

Uma árvore com raiz designa-se por **árvore  $m$ -ária** se todo o vértice interno não tiver mais de  $m$  filhos.

No caso  $m = 2$ , a árvore chama-se **binária** (ver por exemplo, Figura 1.26).

A árvore **árvore  $m$ -ária** diz-se uma **árvore  $m$ -ária plena** se todo o vértice interno tiver exatamente  $m$  filhos.

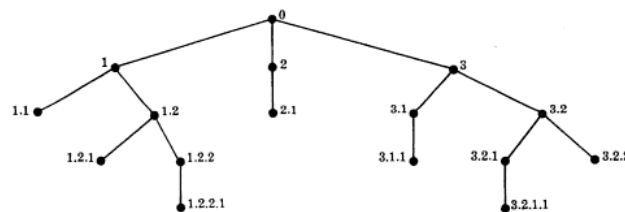


Figura 1.26: Árvore com raiz em 0.

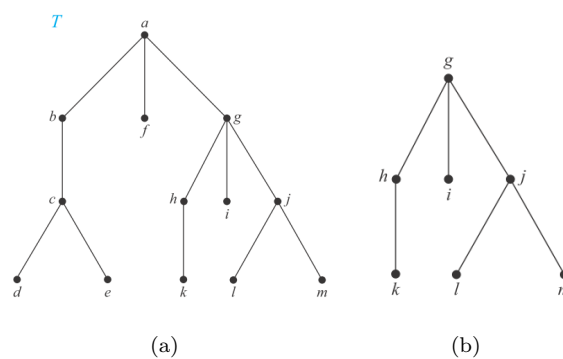


Figura 1.27: (a) Árvore  $T$  com raiz  $a$ . (b) Sub-árvore com raiz  $g$ .

**Exemplo 13:**

Na árvore  $T$  da Figura 1.27:

- O pai de  $c$  é  $b$ .
- Os irmãos de  $h$  são  $i$  e  $j$ .
- Os ascendentes de  $e$  são  $c, b$  e  $a$ .
- Os descendentes de  $b$  são  $c, d$  e  $e$ .
- Os vértices internos são  $a, b, c, g, h$  e  $j$ .
- As folhas são  $d, e, f, i, k, l$  e  $m$ .

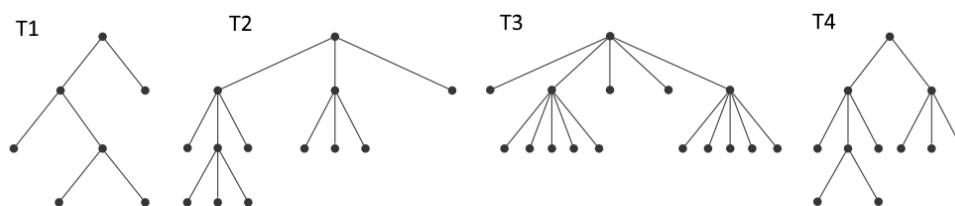


Figura 1.28: Árvores com raiz.

**Exemplo 14:**

Na Figura 1.28:

- $T_1$  é uma árvore binária plena porque cada um dos seus vértices internos tem 2 filhos.
- $T_2$  é uma árvore 3-ária plena porque cada um dos seus vértices internos tem 3 filhos.
- $T_3$  é uma árvore 5-ária plena porque cada um dos seus vértices internos tem 5 filhos.
- $T_4$  não é uma árvore  $m$ -ária plena porque alguns dos vértices internos têm 2 filhos enquanto que outros têm 3 filhos.

**Árvores geradoras de custo mínimo**

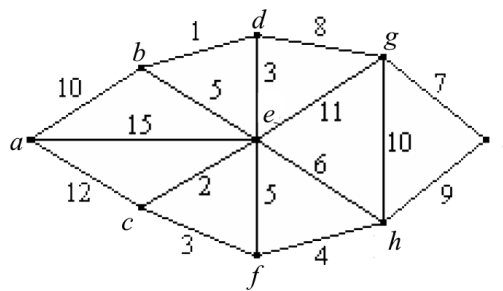
Pretende-se estabelecer uma ligação elétrica entre as várias tomadas de um dos andares de uma habitação. Que ligação deve ser estabelecida de forma a usar o menor comprimento de cabo possível?

A resolução deste problema pode ser feita usando um grafo pesado no qual:

- os vértices representam as tomadas;
- as arestas são as ligações entre as tomadas;
- os pesos das arestas são os comprimentos dessas ligações.

Suponhamos que existem 9 tomadas  $a, b, c, d, e, f, g, h$ , sem a tomada  $a$  a mais próxima do quadro elétrico. Consideremos que as ligações entre tomadas que podem ser consideradas vizinhas.

Medindo as distâncias entre tomadas será possível construir um grafo como o do exemplo seguinte:



Por exemplo, para “levar corrente” de  $a$  para  $g$ , há várias possibilidades tais como:

- $a, b, d, g$
- $a, e, g$ ;
- $a, e, d, g$ ;
- $a, c, e, g$ ;
- entre outras...

Destes apenas um caminho pode ser escolhido

Do mesmo modo para “levar corrente” de  $a$  para cada um dos oito vértices (i.e.  $b, c, d, e, f, g, h, i$  terão de ser escolhidos oito caminhos (um por cada vértice). Interessa que a escolha desses caminhos corresponda à utilização do menor comprimento de cabo possível!!! Esses 8 caminhos formarão a **árvore geradora de custo mínimo do grafo**.

### Definição 20:

Seja  $G$  um grafo simples. Uma **árvore geradora** de  $G$  é um subgrafo de  $G$  que consiste numa árvore que contém todos os vértices de  $G$  <sup>a</sup>.

Uma **árvore geradora de custo mínimo de um grafo ponderado** é uma árvore constituída por todos os vértices desse grafo e pelas arestas para as quais se obtém um peso total (soma dos pesos dessas arestas) menor.

---

<sup>a</sup>Para obter árvores geradoras de grafos pode-se usar algoritmos como por exemplo o depth-first search (DFS) ou o breadth-first search (BFS), que podem ser encontrados em [4]

A árvore geradora de custo mínimo de um grafo pode ser obtida usando o **Algoritmo de Kruskal** que se descreve de seguida. Sejam  $v_1, v_2, v_n$  os vértices do grafo  $G$ . A árvore  $T$  será um conjunto de arestas que, no final, constituirão a árvore geradora de custo mínimo e  $S$  será uma partição do conjunto dos vértices de  $G$  que irá variar de iteração para iteração.

*Input:* grafo não-orientado conexo com pesos e  $n$  vértices

1. **Construir** o vetor RAMO com as arestas por ordem crescente dos seus pesos.
2. Faça-se  $T = \emptyset$ ,  $S = \{\{v_1\}, \{v_2\}, \dots, \{v_n\}\}$ ,  $nr = 0$  e  $k = 0$
3. Enquanto  $nr < n - 1$

$$k = k + 1; (v_i, v_j) = RAMO[k];$$

**Procurar** em  $S$  o subconjunto de  $V$  que contém  $v_i$ . Seja  $S_i$  esse subconjunto;

**Procurar** em  $S$  o subconjunto que contém  $v_j$ . Seja  $S_j$  esse subconjunto;

Se  $S_i \neq S_j$

então  $T = T \cup \{(v_i, v_j)\}$ ;

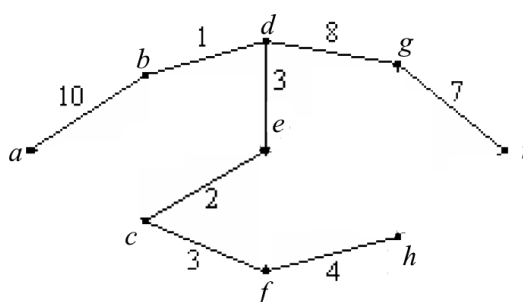
$S_i = S_i \cup S_j$ ;

$nr = nr + 1$ ;

Fim (Enquanto)

**Exercício:** Encontrar a árvore geradora de custo mínimo para o grafo anterior.

Resposta: Custo = 38 e  $T = \{(b, d), (c, e), (c, f), (d, e), (f, h), (g, i), (d, g), (a, b)\}$



## Aplicações

**Código de Huffman** O código de Huffman é um algoritmo utilizado na compressão de dados: bits string que representam textos, ficheiros de imagem e de áudio. Voltaremos a este assunto mais tarde nesta UC.

**Check collision:** Um exemplo de árvores e grafos diz respeito a um algoritmo de verificação de colisões de um sistema robótico. Na Figura 1.29 podemos ver que para os objetos existentes no *workspace* do robot é criada uma árvore em que cada vértice é constituído por uma representação mais ou menos grosseira do objeto ou por várias “peças”. Analogamente para o braço e mão do robot é criada uma árvore constituída por uma ou várias “partes” do braço e mão do robot.

**Organização de uma empresa** : A estrutura de uma grande empresa pode ser modelada como árvore com raiz. Cada vértice da árvore representa um cargo na organização. Por outro lado, cada aresta indica que é o chefe de cada vértice. Na Figura 1.30, podemos ver por exemplo que o *Director of Hardware Development* trabalha diretamente para o *VicePresident of R&D*.

Outros exemplos de aplicações são a representação de expressões algébricas (Figura 1.32) ou a tarefa de construção de um objeto por um robot (Figura 1.33).

Para exemplos de outras aplicações pode consulta, por exemplo, [4] ou [1].

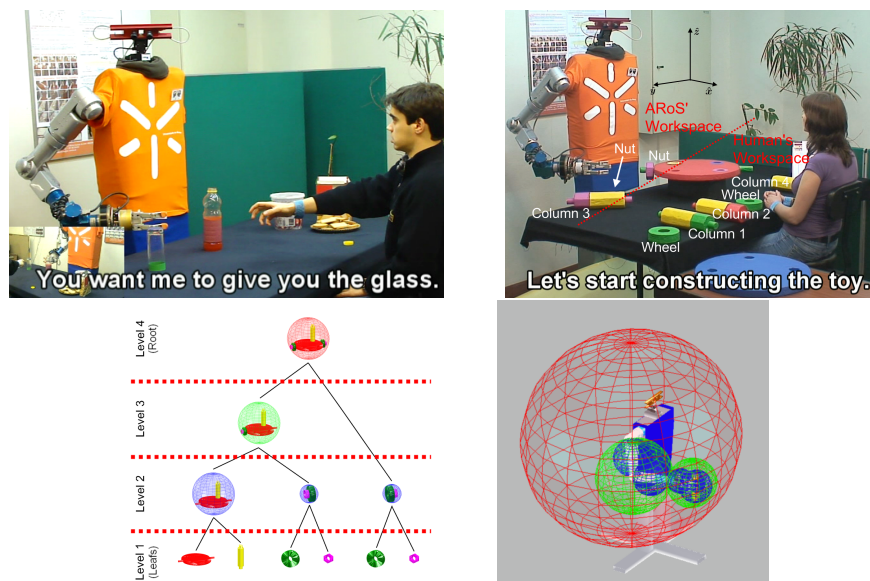
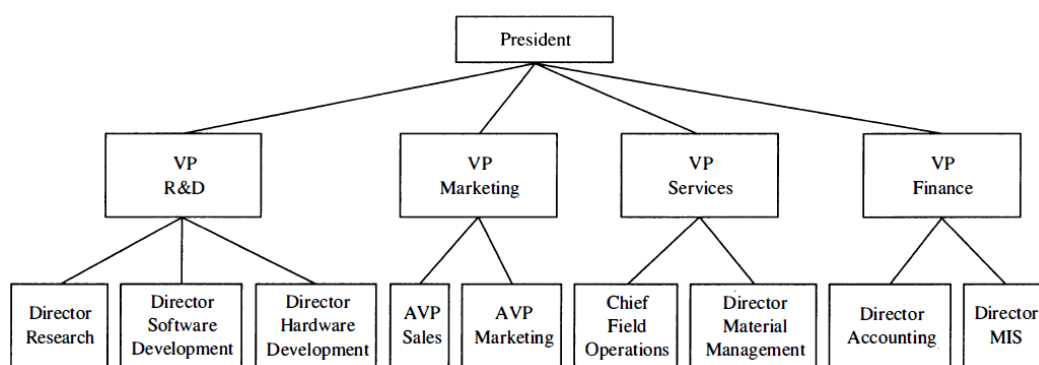


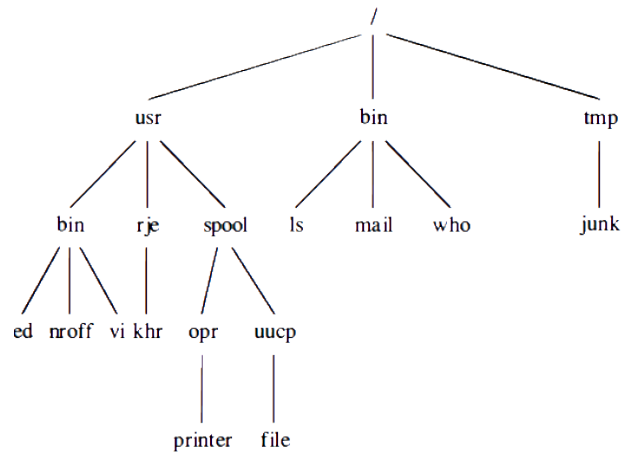
Figura 1.29: Detecção de colisões (in E. Costa e Silva, 2011 (PhD Thesis))



in (Rosen, 2012)

Figura 1.30: Árvore relativa à organização de uma empresa.





in (Rosen, 2012)

Figura 1.31: Pastas de um computador.

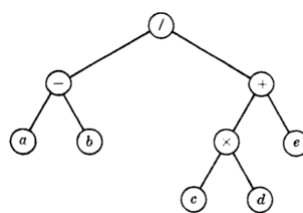


Figura 1.32: Árvore binária com raiz em / que representa a expressão

$$(a - b)/(cd + e)$$

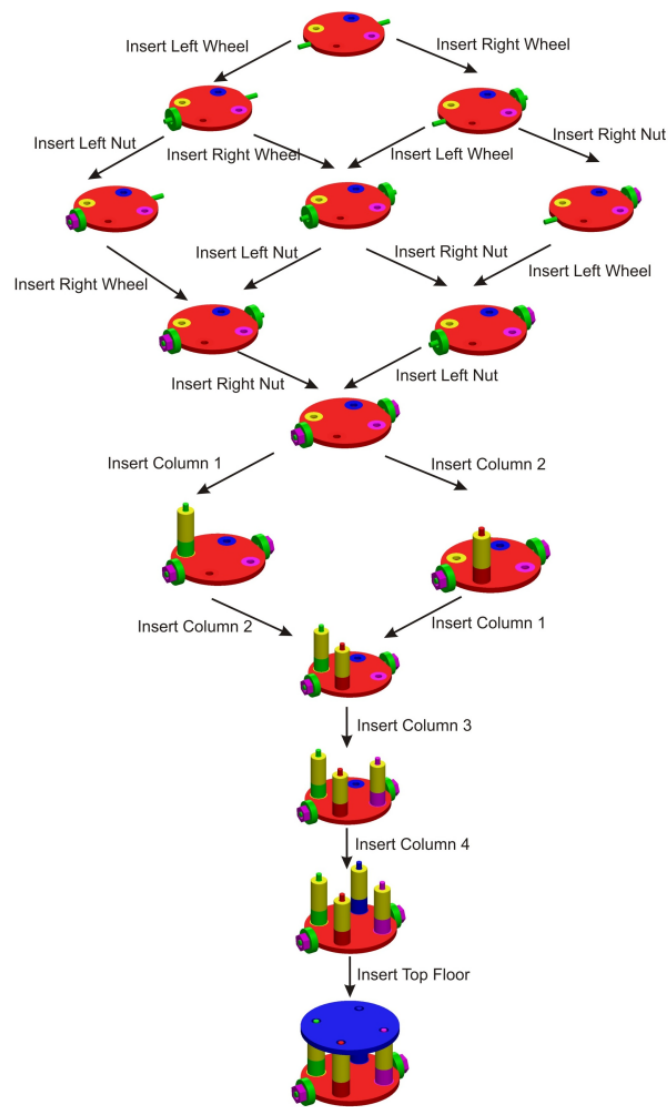


Figura 1.33: Construção de um objeto por um robot.

## Outras Terminologias

Em alguns livros é usada a terminologia que se apresenta de seguida.

**Passeios** Dado um grafo  $G = (V(G), E(G))$ , designamos por **passeio** em  $G$  toda a sequência não vazia

$$P = (v_0, e_1, v_1, e_2, v_2, \dots, e_k, v_k)$$

com  $v_0, v_1, \dots, v_k \in V(G)$ ,  $e_1, e_2, \dots, e_k \in E(G)$  e os vértices  $v_{i-1}$  e  $v_i$  são os extremos da aresta  $e_i$ , para todo  $i = 1, \dots, k$ .

Na Figura 1.34(a) é apresentado um exemplo de um passeio.

O passeio  $P$  designa-se por **trajeto** se todas as arestas forem distintas (Ver exemplo da Figura 1.34(b)).

Se, adicionalmente, todos os vértices forem distintos, o passeio  $P$  designa-se por **caminho simples** (ou, simplesmente, caminho). Na Figura 1.34(c) é apresentado um exemplo de um caminho simples.

No caso do vértice inicial ser igual ao vértice final, então designamos os conceitos anteriores, respetivamente, por **passeio fechado** (Figura 1.35(a)), **trajeto fechado** (Figura 1.35(b)) e **ciclo** (Figura 1.9), em que, neste último conceito, não existe repetição dos vértices exceto o inicial e o final.

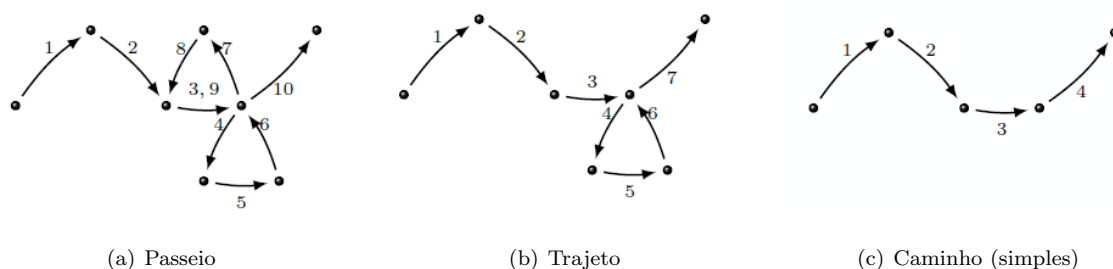


Figura 1.34: Passeio, trajeto e caminho.

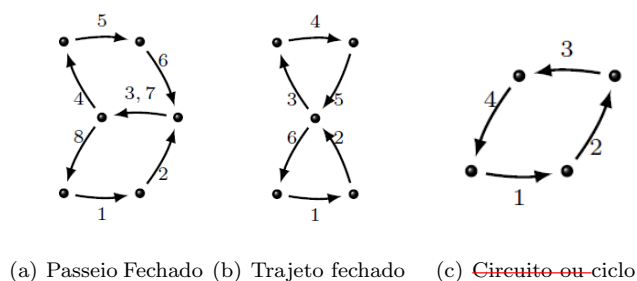


Figura 1.35: Passeio fechado, Trajeto fechado e ciclo.



# Bibliografia

- [1] J. L. Gersting. *Mathematical Structures for Computer Science: A Modern Approach to Discrete Mathematics*. W.H. Freeman & Company, 6th edition edition, 2007.
- [2] H. Lieberman. *Introduction to Operations Research*. McGraw-Hill, 8th edition edition, 2007.
- [3] S. Lipschutz and M. Lipson. *Matemática Discreta*. Bookman, 3.<sup>a</sup> edição edition, 2013.
- [4] K.H. Rosen. *Discrete Mathematics and Its Applications*. McGraw-Hill, 7th edition edition, 2012.