Graphs, Networks and Algorithms

von
Dieter Jungnickel

Neuausgabe

Thematische Gliederung:

Informatik – Algorithmen & Datenstrukturen

Springer 2007

Verlag C.H. Beck im Internet:
www.beck.de
ISBN 978 3 540 72779 8

Inhaltsverzeichnis: Graphs, Networks and Algorithms – Jungnickel

1

Basic Graph Theory

It is time to get back to basics.
John Major

Graph theory began in 1736 when Leonhard Euler (1707–1783) solved the well-known *Königsberg bridge problem* [Eul36][1]. This problem asked for a circular walk through the town of Königsberg (now Kaliningrad) in such a way as to cross over each of the seven bridges spanning the river Pregel once, and only once; see Figure 1.1 for a rough sketch of the situation.

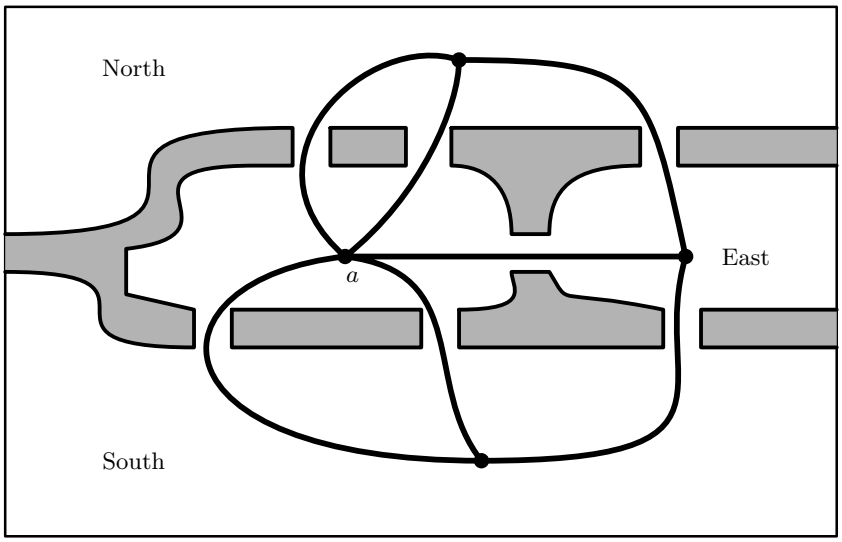

Fig. 1.1. The Königsberg bridge problem

When trying to solve this problem one soon gets the feeling that there is no solution. But how can this be proved? Euler realized that the precise shapes

[1] see [Wil86] and [BiLW76].

of the island and the other three territories involved are not important; the solvability depends only on their *connection properties*. Let us represent the four territories by points (called *vertices*), and the bridges by curves joining the respective points; then we get the *graph* also drawn in Figure 1.1. Trying to arrange a circular walk, we now begin a tour, say, at the vertex called a. When we return to a for the first time, we have used two of the five bridges connected with a. At our next return to a we have used four bridges. Now we can leave a again using the fifth bridge, but there is no possibility to return to a without using one of the five bridges a second time. This shows that the problem is indeed unsolvable. Using a similar argument, we see that it is also impossible to find *any* walk – not necessarily circular, so that the tour might end at a vertex different from where it began – which uses each bridge exactly once. Euler proved even more: he gave a necessary and sufficient condition for an arbitrary graph to admit a circular tour of the above kind. We will treat his theorem in Section 1.3. But first, we have to introduce some basic notions.

The present chapter contains a lot of definitions. We urge the reader to work on the exercises to get a better idea of what the terms really mean. Even though this chapter has an introductory nature, we will also prove a couple of nontrivial results and give two interesting applications. We warn the reader that the terminology in graph theory lacks universality, although this improved a little after the book by Harary [Har69] appeared.

1.1 Graphs, subgraphs and factors

A *graph* G is a pair $G = (V, E)$ consisting of a finite[2] set $V \neq \emptyset$ and a set E of two-element subsets of V. The elements of V are called *vertices*. An element $e = \{a, b\}$ of E is called an *edge* with *end vertices* a and b. We say that a and b are *incident* with e and that a and b are *adjacent* or *neighbors* of each other, and write $e = ab$ or $a \overset{e}{—} b$.

Let us mention two simple but important series of examples. The *complete graph* K_n has n vertices (that is, $|V| = n$) and all two-element subsets of V as edges. The *complete bipartite graph* $K_{m,n}$ has as vertex set the disjoint union of a set V_1 with m elements and a set V_2 with n elements; edges are all the sets $\{a, b\}$ with $a \in V_1$ and $b \in V_2$.

We will often illustrate graphs by pictures in the plane. The vertices of a graph $G = (V, E)$ are represented by (bold type) points and the edges by lines (preferably straight lines) connecting the end points. We give some examples in Figure 1.2. We emphasize that in these pictures the lines merely serve to indicate the vertices with which they are incident. In particular, the *inner points* of these lines as well as possible points of intersection of two edges (as in Figure 1.2 for the graphs K_5 and $K_{3,3}$) are not significant. In Section 1.5 we

[2] In graph theory, infinite graphs are studied as well. However, we restrict ourselves in this book – like [Har69] – to the finite case.

will study the question which graphs can be drawn without such additional points of intersection.

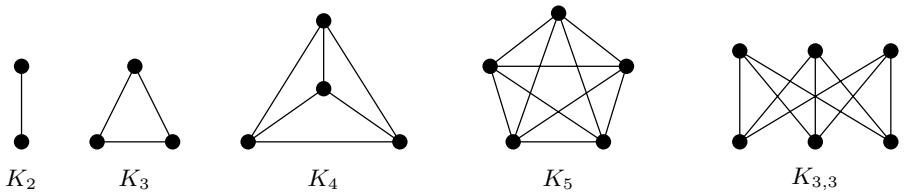

Fig. 1.2. Some graphs

Let $G = (V, E)$ be a graph and V' be a subset of V. By $E|V'$ we denote the set of all edges $e \in E$ which have both their vertices in V'. The graph $(V', E|V')$ is called the *induced subgraph* on V' and is denoted by $G|V'$. Each graph of the form (V', E') where $V' \subset V$ and $E' \subset E|V'$ is said to be a *subgraph* of G, and a subgraph with $V' = V$ is called a *spanning subgraph*. Some examples are given in Figure 1.3.

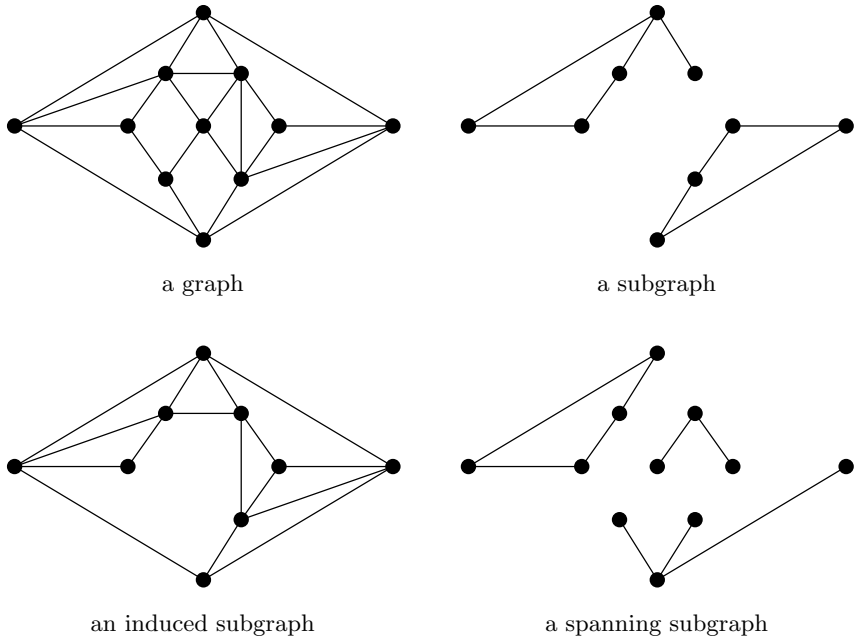

Fig. 1.3. Subgraphs

Given any vertex v of a graph, the *degree of* v, $\deg v$, is the number of edges incident with v. We can now state our first – albeit rather simple – result:

Lemma 1.1.1. *In any graph, the number of vertices of odd degree is even.*

Proof. Summing the degree over all vertices v, each edge is counted exactly twice, once for each of its vertices; thus $\sum_v \deg v = 2|E|$. As the right hand side is even, the number of odd terms $\deg v$ in the sum on the left hand side must also be even. □

If all vertices of a graph G have the same degree (say r), G is called a *regular* graph, more precisely an *r-regular* graph. The graph K_n is $(n-1)$-regular, the graph $K_{m,n}$ is regular only if $m = n$ (in which case it is n-regular). A *k-factor* is a k-regular spanning subgraph. If the edge set of a graph can be divided into k-factors, such a decomposition is called a *k-factorization* of the graph. A 1-factorization is also called a *factorization* or a *resolution*. Obviously, a 1-factor can exist only if G has an even number of vertices. Factorizations of K_{2n} may be interpreted as schedules for a tournament of $2n$ teams (in soccer, basketball etc.). The following exercise shows that such a factorization exists for all n. The problem of setting up schedules for tournaments will be studied in Section 1.7 as an application.

Exercise 1.1.2. We use $\{\infty, 1, \ldots, 2n-1\}$ as the vertex set of the complete graph K_{2n} and divide the edge set into subsets F_i for $i = 1, \ldots, 2n-1$, where $F_i = \{\infty i\} \cup \{jk : j + k \equiv 2i \pmod{2n-1}\}$. Show that the F_i form a factorization of K_{2n}. The case $n = 3$ is shown in Figure 1.4. Factorizations were first introduced by [Kir47]; interesting surveys are given by [MeRo85] and [Wal92].

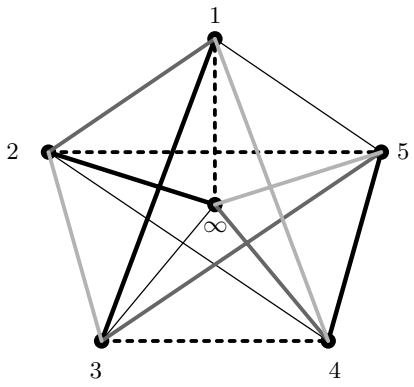

Fig. 1.4. A factorization of K_6

Let us conclude this section with two more exercises. First, we introduce a further family of graphs. The *triangular graph* T_n has as vertices the two-element subsets of a set with n elements. Two of these vertices are adjacent if and only if their intersection is not empty. Obviously, T_n is a $(2n-4)$-regular graph. But T_n has even stronger regularity properties: the number of vertices adjacent to two given vertices x, y depends only on whether x and y themselves are adjacent or not. Such a graph is called a *strongly regular graph*, abbreviated by *SRG*. These graphs are of great interest in finite geometry; see the books [CaLi91] and [BeJL99]. We will limit our look at SRG's in this book to a few exercises.

Exercise 1.1.3. Draw the graphs T_n for $n = 3, 4, 5$ and show that T_n has parameters $a = 2n-4$, $c = n-2$ and $d = 4$, where a is the degree of any vertex, c is the number of vertices adjacent to both x and y if x and y are adjacent, and d is the number of vertices adjacent to x and y if x and y are not adjacent.

For the next exercise, we need another definition. For a graph $G = (V, E)$, we will denote by $\binom{V}{2}$ the set of all pairs of its vertices. The graph $\overline{G} = (V, \binom{V}{2} \setminus E)$ is called the *complementary graph*. Two vertices of V are adjacent in $\overline{G}$ if and only if they are not adjacent in G.

Exercise 1.1.4. Let G be an SRG with parameters a, c, and d having n vertices. Show that $\overline{G}$ is also an SRG and determine its parameters. Moreover, prove the formula

$$a(a-c-1) = (n-a-1)d.$$

Hint: Count the number of edges yz for which y is adjacent to a given vertex x, whereas z is not adjacent to x.

1.2 Paths, cycles, connectedness, trees

Before we can go on to the theorem of Euler mentioned in Section 1.1, we have to formalize the idea of a circular tour. Let $(e_1, \ldots, e_n)$ be a sequence of edges in a graph G. If there are vertices $v_0, \ldots, v_n$ such that $e_i = v_{i-1}v_i$ for $i = 1, \ldots, n$, the sequence is called a *walk*; if $v_0 = v_n$, one speaks of a *closed walk*. A walk for which the e_i are distinct is called a *trail*, and a closed walk with distinct edges is a *closed trail*. If, in addition, the v_j are distinct, the trail is a *path*. A closed trail with $n \geq 3$, for which the v_j are distinct (except, of course, $v_0 = v_n$), is called a *cycle*. In any of these cases we use the notation

$$W: \quad v_0 \overset{e_1}{\text{——}} v_1 \overset{e_2}{\text{——}} v_2 \text{——} \ldots \text{——} v_{n-1} \overset{e_n}{\text{——}} v_n$$

and call n the *length* of W. The vertices v_0 and v_n are called the *start vertex* and the *end vertex* of W, respectively. We will sometimes specify a walk by

its sequence of vertices $(v_0, \ldots, v_n)$, provided that $v_{i-1}v_i$ is an edge for $i = 1, \ldots, n$. In the graph of Figure 1.5, (a, b, c, v, b, c) is a walk, but not a trail; and (a, b, c, v, b, u) is a trail, but not a path. Also, (a, b, c, v, b, u, a) is a closed trail, but not a cycle, whereas (a, b, c, w, v, u, a) is a cycle. The reader might want to consider some more examples.

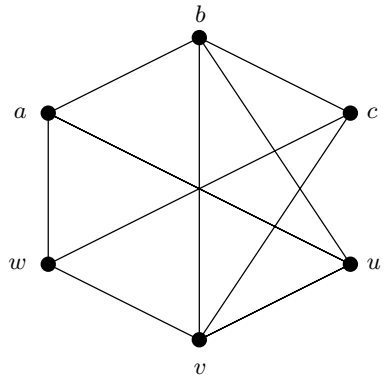

Fig. 1.5. An example for walks

Exercise 1.2.1. Show that any walk with start vertex a and end vertex b, where $a \neq b$, contains a path from a to b. Also prove that any closed walk of odd length contains a cycle. What do closed walks not containing a cycle look like?

Two vertices a and b of a graph G are called *connected* if there exists a walk with start vertex a and end vertex b. If all pairs of vertices of G are connected, G itself is called *connected*. For any vertex a, we consider (a) as a trivial walk of length 0, so that any vertex is connected with itself. Thus connectedness is an equivalence relation on the vertex set of G. The equivalence classes of this relation are called the *connected components* of G. Thus G is connected if and only if its vertex set V is its unique connected component. Components which contain only one vertex are also called *isolated vertices*. Let us give some exercises concerning these definitions.

Exercise 1.2.2. Let G be a graph with n vertices and assume that each vertex of G has degree at least $(n-1)/2$. Show that G must be connected.

Exercise 1.2.3. A graph G is connected if and only if there exists an edge $e = vw$ with $v \in V_1$ and $w \in V_2$ whenever $V = V_1 \dot{\cup} V_2$ (that is, $V_1 \cap V_2 = \emptyset$) is a decomposition of the vertex set of G.

Exercise 1.2.4. If G is not connected, the complementary graph $\overline{G}$ is connected.

If a and b are two vertices in the same connected component of a graph G, there has to exist a path of shortest length (say d) between a and b. (Why?) Then a and b are said to have *distance* $d = d(a, b)$. The notion of distances in a graph is fundamental; we will study it (and a generalization) thoroughly in Chapter 3.

In the remainder of this section, we will investigate the minimal connected graphs. First, some more definitions and an exercise. A graph is called *acyclic* if it does not contain a cycle. For a subset T of the vertex set V of a graph G we denote by $G \setminus T$ the induced subgraph on $V \setminus T$. This graph arises from G by omitting all vertices in T and all edges incident with these vertices. For a one-element set $T = \{v\}$ we write $G \setminus v$ instead of $G \setminus \{v\}$.

Exercise 1.2.5. Let G be a graph having n vertices, none of which are isolated, and $n-1$ edges, where $n \geq 2$. Show that G contains at least two vertices of degree 1.

Lemma 1.2.6. *A connected graph on n vertices has at least $n - 1$ edges.*

Proof. We use induction on n; the case $n = 1$ is trivial. Thus let G be a connected graph on $n \geq 2$ vertices. Choose an arbitrary vertex v of G and consider the graph $H = G \setminus v$. Note that H is not necessarily connected. Suppose H has connected components Z_i having n_i vertices ($i = 1, \ldots, k$), that is, $n_1 + \ldots + n_k = n - 1$. By induction hypothesis, the subgraph of H induced on Z_i has at least $n_i - 1$ edges. Moreover, v must be connected in G with each of the components Z_i by at least one edge. Thus G contains at least $(n_1 - 1) + \ldots + (n_k - 1) + k = n - 1$ edges. □

Lemma 1.2.7. *An acyclic graph on n vertices has at most $n - 1$ edges.*

Proof. If $n = 1$ or $E = \emptyset$, the statement is obvious. For the general case, choose any edge $e = ab$ in G. Then the graph $H = G \setminus e$ has exactly one more connected component than G. (Note that there cannot be a path in H from a to b, because such a path together with the edge e would give rise to a cycle in G.) Thus, H can be decomposed into connected, acyclic graphs $H_1, \ldots, H_k$ (where $k \geq 2$). By induction, we may assume that each graph H_i contains at most $n_i - 1$ edges, where n_i denotes the number of vertices of H_i. But then G has at most

$$(n_1 - 1) + \ldots + (n_k - 1) + 1 \;=\; (n_1 + \ldots + n_k) - (k - 1) \;\leq\; n - 1$$

edges. □

Theorem 1.2.8. *Let G be a graph with n vertices. Then any two of the following conditions imply the third:*

(a) *G is connected.*
(b) *G is acyclic.*
(c) *G has $n - 1$ edges.*

Proof. First let G be acyclic and connected. Then Lemmas 1.2.6 and 1.2.7 imply that G has exactly $n-1$ edges.

Next let G be a connected graph with $n-1$ edges. Suppose G contains a cycle C and consider the graph $H = G \setminus e$, where e is some edge of C. Then H is a connected with n vertices and $n-2$ edges, contradicting Lemma 1.2.6.

Finally, let G be an acyclic graph with $n-1$ edges. Then Lemma 1.2.7 implies that G cannot contain an isolated vertex, as omitting such a vertex would give an acyclic graph with $n-1$ vertices and $n-1$ edges. Now Exercise 1.2.5 shows that G has a vertex of degree 1, so that $G \setminus v$ is an acyclic graph with $n-1$ vertices and $n-2$ edges. By induction it follows that $G \setminus v$ and hence G are connected. □

Exercise 1.2.9. Give a different proof for Lemma 1.2.6 using the technique of omitting an edge e from G.

A graph T for which the conditions of Theorem 1.2.8 hold is called a *tree*. A vertex of T with degree 1 is called a *leaf*. A *forest* is a graph whose connected components are trees. We will have a closer look at trees in Chapter 4.

In Section 4.2 we will use rather sophisticated techniques from linear algebra to prove a formula for the number of trees on n vertices; this result is usually attributed to Cayley [Cay89], even though it is essentially due to Borchardt [Bor60]. Here we will use a more elementary method to prove a stronger result – which is indeed due to Cayley. By $f(n,s)$ we denote the number of forests G having n vertices and exactly s connected components, for which s fixed vertices are in distinct components; in particular, the number of trees on n vertices is $f(n,1)$. Cayley's theorem gives a formula for the numbers $f(n,s)$; we use a simple proof taken from [Tak90a].

Theorem 1.2.10. *One has* $f(n,s) = sn^{n-s-1}$.

Proof. We begin by proving the following recursion formula:

$$f(n,s) \;=\; \sum_{j=0}^{n-s} \binom{n-s}{j} f(n-1, s+j-1), \tag{1.1}$$

where we put $f(1,1) = 1$ and $f(n,0) = 0$ for $n \geq 1$. How can an arbitrary forest G with vertex set $V = \{1,\ldots,n\}$ having precisely s connected components be constructed? Let us assume that the vertices $1,\ldots,s$ are the specified vertices which belong to distinct components. The degree of vertex 1 can take the values $j = 0,\ldots,n-s$, as the neighbors of 1 may form an arbitrary subset $\Gamma(1)$ of $\{s+1,\ldots,n\}$. Then we have – after choosing the degree j of 1 – exactly $\binom{n-s}{j}$ possibilities to choose $\Gamma(1)$. Note that the graph $G \setminus 1$ is a forest with vertex set $V \setminus \{1\} = \{2,\ldots,n\}$ and exactly $s+j-1$ connected components, where the vertices $2,\ldots s$ and the j elements of $\Gamma(1)$ are in different connected components. After having chosen j and $\Gamma(1)$, we still have

$f(n-1, s+j-1)$ possibilities to construct the forest $G \setminus 1$. This proves the recursion formula (1.1).

We now prove the desired formula for the $f(n,s)$ by using induction on n. The case $n = 1$ is trivial. Thus we let $n \geq 2$ and assume that

$$f(n-1,i) \;=\; i(n-1)^{n-i-2} \qquad \text{holds for} \quad i = 1, \ldots n-1. \tag{1.2}$$

Using this in equation (1.1) gives

$$\begin{aligned}
f(n,s) \;&=\; \sum_{j=0}^{n-s} \binom{n-s}{j} (s+j-1)(n-1)^{n-s-j-1} \\
&=\; \sum_{j=1}^{n-s} j \binom{n-s}{j} (n-1)^{n-s-j-1} \\
&\qquad\qquad + (s-1) \sum_{j=0}^{n-s} \binom{n-s}{j} (n-1)^{n-s-j-1} \\
&=\; (n-s) \sum_{j=1}^{n-s} \binom{n-s-1}{j-1} (n-1)^{n-s-j-1} \\
&\qquad\qquad + (s-1) \sum_{j=0}^{n-s} \binom{n-s}{j} (n-1)^{n-s-j-1} \\
&=\; \frac{n-s}{n-1} \sum_{k=0}^{n-s-1} \binom{n-s-1}{k} (n-1)^{(n-s-1)-k} \times 1^k \\
&\qquad\qquad + \frac{s-1}{n-1} \sum_{j=0}^{n-s} \binom{n-s}{j} (n-1)^{n-s-j} \times 1^j \\
&=\; \frac{(n-s)n^{n-s-1} + (s-1)n^{n-s}}{n-1} \;=\; sn^{n-s-1}.
\end{aligned}$$

This proves the theorem. □

Note that the rather tedious calculations in the induction step may be replaced by the following – not shorter, but more elegant – combinatorial argument. We have to split up the sum we got from using equation (1.2) in (1.1) in a different way:

$$\begin{aligned}
f(n,s) \;&=\; \sum_{j=0}^{n-s} \binom{n-s}{j} (s+j-1)(n-1)^{n-s-j-1} \\
&=\; \sum_{j=0}^{n-s} \binom{n-s}{j} (n-1)^{n-s-j} \\
&\qquad\qquad - \sum_{j=0}^{n-s-1} \binom{n-s}{j} (n-s-j)(n-1)^{n-s-j-1}.
\end{aligned}$$

Now the first sum counts the number of words of length $n-s$ over the alphabet $V = \{1, \dots, n\}$, as the binomial coefficient counts the number of possibilities for distributing j entries 1 (where j has to be between 0 and $n-s$), and the factor $(n-1)^{n-s-j}$ gives the number of possibilities for filling the remaining $n-s-j$ positions with entries $\neq 1$. Similarly, the second sum counts the number of words of length $n-s$ over the alphabet $V = \{0, 1, \dots, n\}$ which contain exactly one entry 0. As there are obvious formulas for these numbers, we directly get

$$f(n,s) \;=\; n^{n-s} - (n-s)n^{n-s-1} \;=\; sn^{n-s-1}.$$

Borchardt's result is now an immediate consequence of Theorem 1.2.10:

Corollary 1.2.11. *The number of trees on n vertices is n^{n-2}.* □

It is interesting to note that n^{n-2} is also the cardinality of the set $\mathbf{W}$ of words of length $n-2$ over an alphabet V with n elements, which suggests that we might prove Corollary 1.2.11 by constructing a bijection between $\mathbf{W}$ and the set $\mathbf{T}$ of trees with vertex set V. This is indeed possible as shown by Prüfer [Pru18]; we will follow the account in [Lue89] and construct the *Prüfer code* $\pi_V \colon \mathbf{T} \to \mathbf{W}$ recursively. As we will need an ordering of the elements of V, we assume in what follows, without loss of generality, that V is a subset of $\mathbb{N}$.

Thus let $G = (V, E)$ be a tree. For $n = 2$ the only tree on V is mapped to the empty word; that is, we put $\pi_V(G) = ()$. For $n \geq 3$ we use the smallest leaf of G to construct a tree on $n-1$ vertices. We write

$$v \;=\; v(G) \;=\; \min\{u \in V \colon \deg_G(u) = 1\} \tag{1.3}$$

and denote by $e = e(G)$ the unique edge incident with v, and by $w = w(G)$ the other end vertex of e. Now let $G' = G \setminus v$. Then G' has $n-1$ vertices, and we may assume by induction that we know the word corresponding to G' under the Prüfer code on $V' = V \setminus \{v\}$. Hence we can define recursively

$$\pi_V(G) \;=\; (w, \pi_{V'}(G')). \tag{1.4}$$

It remains to show that we have indeed constructed the desired bijection. We need the following lemma which allows us to determine the minimal leaf of a tree G on V from its Prüfer code.

Lemma 1.2.12. *Let G be a tree on V. Then the leaves of G are precisely those elements of V which do not occur in $\pi_V(G)$. In particular,*

$$v(G) \;=\; \min\{u \in V \colon u \textit{ does not occur in } \pi_V(G)\}. \tag{1.5}$$

Proof. First suppose that an element u of V occurs in $\pi_V(G)$. Then u was added to $\pi_V(G)$ at some stage of our construction; that is, some subtree H of G was considered, and u was adjacent to the minimal leaf $v(H)$ of H. Now

if u were also a leaf of G (and thus of H), then H would have to consist only of u and $v(G)$, so that H would have the empty word as Prüfer code, and u would not occur in $\pi_V(G)$, contradicting our assumption.

Now suppose that u is not a leaf. Then there is at least one edge incident with u which is discarded during the construction of the Prüfer code of G, since the construction only ends when a tree on two vertices remains of G. Let e be the edge incident with u which is omitted first. At that point of the construction, u is not a leaf, so that the other end vertex of e has to be the minimal leaf of the respective subtree. But then, by our construction, u is used as the next coordinate in $\pi_V(G)$. □

Theorem 1.2.13. *The Prüfer code $\pi_V : \mathbf{T} \to \mathbf{W}$ defined by equations (1.3) and (1.4) is a bijection.*

Proof. For $n = 2$, the statement is clear, so let $n \geq 3$. First we show that π_V is surjective. Let $\mathbf{w} = (w_1, \ldots, w_{n-2})$ be an arbitrary word over V, and denote by v the smallest element of V which does not occur as a coordinate in $\mathbf{w}$. By induction, we may assume that there is a tree G' on the vertex set $V' = V \setminus \{v\}$ with $\pi_{V'}(G') = (w_2, \ldots, w_{n-2})$. Now we add the edge $e = vw_1$ to G' (as Lemma 1.2.12 suggests) and get a tree G on V. It is easy to verify that $v = v(G)$ and thus $\pi_V(G) = \mathbf{w}$. To prove injectivity, let G and H be two trees on $\{1, \ldots, n\}$ and suppose $\pi_V(G) = \pi_V(H)$. Now let v be the smallest element of V which does not occur in $\pi_V(G)$. Then Lemma 1.2.12 implies that $v = v(G) = v(H)$. Thus G and H both contain the edge $e = vw$, where w is the first entry of $\pi_V(G)$. Then G' and H' are both trees on $V' = V \setminus \{v\}$, and we have $\pi_{V'}(G') = \pi_{V'}(H')$. Using induction, we conclude $G' = H'$ and hence $G = H$. □

Note that the proof of Theorem 1.2.13 together with Lemma 1.2.12 gives a constructive method for decoding the Prüfer code.

Example 1.2.14. Figure 1.6 shows some trees and their Prüfer codes for $n = 6$ (one for each isomorphism class, see Exercise 4.1.6).

Exercise 1.2.15. Determine the trees with vertex set $\{1, \ldots, n\}$ corresponding to the following Prüfer codes: $(1, 1, \ldots, 1)$; $(2, 3, \ldots, n-2, n-1)$; $(2, 3, \ldots, n-3, n-2, n-2)$; $(3, 3, 4, \ldots, n-3, n-2, n-2)$.

Exercise 1.2.16. How can we determine the degree of an arbitrary vertex u of a tree G from its Prüfer code $\pi_V(G)$? Give a condition for $\pi_V(G)$ to correspond to a path or a star (where a *star* is a tree having one exceptional vertex z which is adjacent to all other vertices).

Exercise 1.2.17. Let $(d_1, \ldots, d_n)$ be a sequence of positive integers. Show that there is a tree on n vertices having degrees $d_1, \ldots, d_n$ if and only if

$$d_1 + \ldots + d_n \;=\; 2(n-1), \tag{1.6}$$

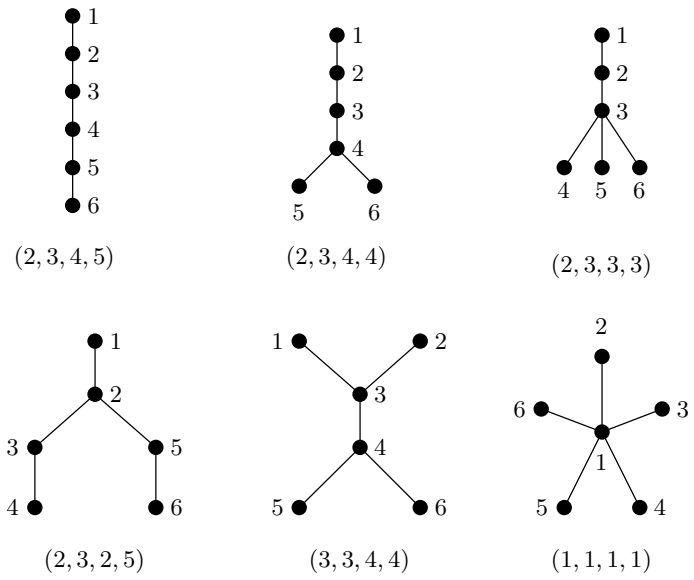

Fig. 1.6. Some trees and their Prüfer codes

and construct a tree with degree sequence $(1, 1, 1, 1, 2, 3, 3)$. Hint: Use the Prüfer code.

We remark that the determination of the possible degree sequences for arbitrary graphs on n vertices is a considerably more difficult problem; see, for instance, [SiHo91] and [BaSa95].

We have now seen two quite different proofs for Corollary 1.2.11 which illustrate two important techniques for solving enumeration problems, namely using recursion formulas on the one hand and using bijections on the other. In Section 4.2 we will see yet another proof which will be based on the application of algebraic tools (like matrices and determinants). In this text, we cannot treat the most important tool of enumeration theory, namely generating functions. The interested reader can find the basics of enumeration theory in any good book on combinatorics; for a more thorough study we recommend the books by Stanley [Sta86, Sta99] or the extensive monograph [GoJa83], all of which are standard references.

Let us also note that the number $f(n)$ of forests on n vertices has been studied several times; see [Tak90b] and the references given there. Takács proves the following simple formula which is, however, not at all easy to derive:

$$f(n) \;=\; \frac{n!}{n+1} \sum_{j=0}^{\lfloor n/2 \rfloor} (-1)^j \frac{(2j+1)(n+1)^{n-2j}}{2^j j!(n-2j)!}.$$

Finally, me mention an interesting asymptotic result due to Rényi [Ren59] which compares the number of all forests with the number of all trees:

$$\lim_{n\to\infty} \frac{f(n)}{n^{n-2}} = \sqrt{e} \approx 1.6487.$$

1.3 Euler tours

In this section we will solve the Königsberg bridge problem for arbitrary graphs. The reader should note that Figure 1.1 does not really depict a graph according to the definitions given in Section 1.1, because there are pairs of vertices which are connected by more than one edge. Thus we generalize our definition as follows. Intuitively, for a *multigraph* on a vertex set V, we want to replace the edge set of an ordinary graph by a family E of two-element subsets of V. To be able to distinguish different edges connecting the same pair of vertices, we formally define a *multigraph* as a triple (V, E, J), where V and E are disjoint sets, and J is a mapping from E to the set of two-element subsets of V, the *incidence map*. The image $J(e)$ of an edge e is the set $\{a, b\}$ of end vertices of e. Edges e and e' with $J(e) = J(e')$ are called *parallel*. Then all the notions introduced so far carry over to multigraphs. However, in this book we will – with just a few exceptions – restrict ourselves to graphs.[3]

The circular tours occurring in the Königsberg bridge problem can be described abstractly as follows. An *Eulerian trail* of a multigraph G is a trail which contains each edge of G (exactly once, of course); if the trail is closed, then it is called an *Euler tour*.[4] A multigraph is called *Eulerian* if it contains an Euler tour. The following theorem of [Eul36] characterizes the Eulerian multigraphs.

Theorem 1.3.1 (Euler's theorem). *Let G be a connected multigraph. Then the following statements are equivalent:*

(a) *G is Eulerian.*
(b) *Each vertex of G has even degree.*
(c) *The edge set of G can be partitioned into cycles.*

Proof: We first assume that G is Eulerian and pick an Euler tour, say C. Each occurrence of a vertex v in C adds 2 to its degree. As each edge of G occurs exactly once in C, all vertices must have even degree. The reader should work out this argument in detail.

[3] Some authors denote the structure we call a *multigraph* by *graph*; graphs according to our definition are then called *simple graphs*. Moreover, sometimes even edges e for which $J(e)$ is a set $\{a\}$ having only one element are admitted; such edges are then called *loops*. The corresponding generalization of multigraphs is often called a *pseudograph*.

[4] Sometimes one also uses the term *Eulerian cycle*, even though an Euler tour usually contains vertices more than once.

Next suppose that (b) holds and that G has n vertices. As G is connected, it has at least $n-1$ edges by Lemma 1.2.6. Since G does not contain vertices of degree 1, it actually has at least n edges, by Exercise 1.2.5. Then Lemma 1.2.7 shows that there is a cycle K in G. Removing K from G we get a graph H in which all vertices again have even degree. Considering the connected components of H separately, we may – using induction – partition the edge set of H into cycles. Hence, the edge set of G can be partitioned into cycles.

Finally, assume the validity of (c) and let C be one of the cycles in the partition of the edge set E into cycles. If C is an Euler tour, we are finished. Otherwise there exists another cycle C' having a vertex v in common with C. We can w.l.o.g. use v as start and end vertex of both cycles, so that CC' (that is, C followed by C') is a closed trail. Continuing in the same manner, we finally reach an Euler tour. □

Corollary 1.3.2. *Let G be a connected multigraph with exactly $2k$ vertices of odd degree. Then G contains an Eulerian trail if and only if $k = 0$ or $k = 1$.*

Proof: The case $k = 0$ is clear by Theorem 1.3.1. So suppose $k \neq 0$. Similar to the proof of Theorem 1.3.1 it can be shown that an Eulerian trail can exist only if $k = 1$; in this case the Eulerian trail has the two vertices of odd degree as start and end vertices. Let $k = 1$ and name the two vertices of odd degree a and b. By adding an (additional) edge ab to G, we get a connected multigraph H whose vertices all have even degree. Hence H contains an Euler tour C by Theorem 1.3.1. Omitting the edge ab from C then gives the desired Eulerian trail in G. □

Exercise 1.3.3. Let G be a connected multigraph having exactly $2k$ vertices of odd degree ($k \neq 0$). Then the edge set of G can be partitioned into k trails.

The *line graph* $L(G)$ of a graph G has as vertices the edges of G; two edges of G are adjacent in $L(G)$ if and only if they have a common vertex in G. For example, the line graph of the complete graph K_n is the triangular graph T_n.

Exercise 1.3.4. Give a formula for the degree of a vertex of $L(G)$ (using the degrees in G). In which cases is $L(K_{m,n})$ an SRG?

Exercise 1.3.5. Let G be a connected graph. Find a necessary and sufficient condition for $L(G)$ to be Eulerian. Conclude that the line graph of an Eulerian graph is likewise Eulerian, and show that the converse is false in general.

Finally we recommend the very nice survey [Fle83] which treats Eulerian graphs and a lot of related questions in detail; for another survey, see [LeOe86]. A much more extensive treatment of these subjects can be found in two monographs by Fleischner [Fle90, Fle91]. For a survey of line graphs, see [Pri96].

1.4 Hamiltonian cycles

In 1857 Sir William Rowan Hamilton (1805–1865, known to every mathematician for the quaternions and the theorem of Cayley–Hamilton) invented the following *Icosian game* which he then sold to a London game dealer in 1859 for 25 pounds; it was realized physically as a pegboard with holes. The corners of a regular dodecahedron are labelled with the names of cities; the task is to find a circular tour along the edges of the dodecahedron visiting each city exactly once, where sometimes the first steps of the tour might also be prescribed. More about this game can be found in [BaCo87]. We may model the Icosian game by looking for a cycle in the corresponding *dodecahedral graph* which contains each vertex exactly once. Such a cycle is therefore called a *Hamiltonian cycle*. In Figure 1.7 we give a solution for Hamilton's original problem.

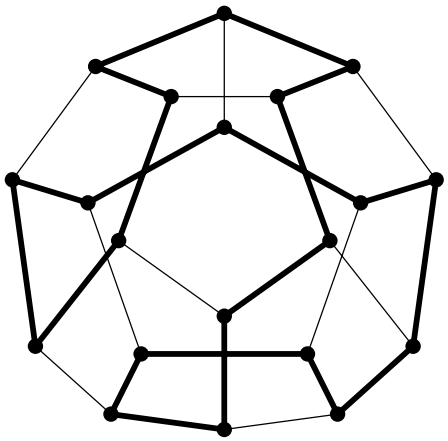

Fig. 1.7. The Icosian game

Although Euler tours and Hamiltonian cycles have similar definitions, they are quite different. For example, there is no nice characterization of *Hamiltonian graphs*; that is, of those graphs containing a Hamiltonian cycle. As we will see in the next chapter, there are good reasons to believe that such a good characterization cannot exist. However, we know many sufficient conditions for the existence of a Hamiltonian cycle; most of these conditions are statements about the degrees of the vertices. Obviously, the complete graph K_n is Hamiltonian.

We first prove a theorem from which we can derive several sufficient conditions on the sequence of degrees in a graph. Let G be a graph on n vertices. If G contains non-adjacent vertices u and v such that $\deg u + \deg v \geq n$, we add the edge uv to G. We continue this procedure until we get a graph

$[G]$, in which, for any two non-adjacent vertices x and y, we always have $\deg x + \deg y < n$. The graph $[G]$ is called the *closure* of G. (We leave it to the reader to show that $[G]$ is uniquely determined.) Then we have the following theorem due to Bondy and Chvátal [BoCh76].

Theorem 1.4.1. *A graph G is Hamiltonian if and only if its closure $[G]$ is Hamiltonian.*

Proof. If G is Hamiltonian, $[G]$ is obviously Hamiltonian. As $[G]$ is derived from G by adding edges sequentially, it will suffice to show that adding just one edge – as described above – does not change the fact whether a graph is Hamiltonian or not. Thus let u and v be two non-adjacent vertices with $\deg u + \deg v \geq n$, and let H be the graph which results from adding the edge uv to G. Suppose that H is Hamiltonian, but G is not. Then there exists a Hamiltonian cycle in H containing the edge uv, so that there is a path $(x_1, x_2, \ldots, x_n)$ in G with $x_1 = u$ and $x_n = v$ containing each vertex of G exactly once. Consider the sets

$$X = \{x_i : vx_{i-1} \in E \text{ and } 3 \leq i \leq n-1\}$$

and

$$Y = \{x_i : ux_i \in E \text{ and } 3 \leq i \leq n-1\}.$$

As u and v are not adjacent in G, we have $|X| + |Y| = \deg u + \deg v - 2 \geq n-2$. Hence there exists an index i with $3 \leq i \leq n-1$ such that vx_{i-1} as well as ux_i are edges in G. But then $(x_1, x_2, \ldots, x_{i-1}, x_n, x_{n-1}, \ldots, x_i, x_1)$ is a Hamiltonian cycle in G (see Figure 1.8), a contradiction. □

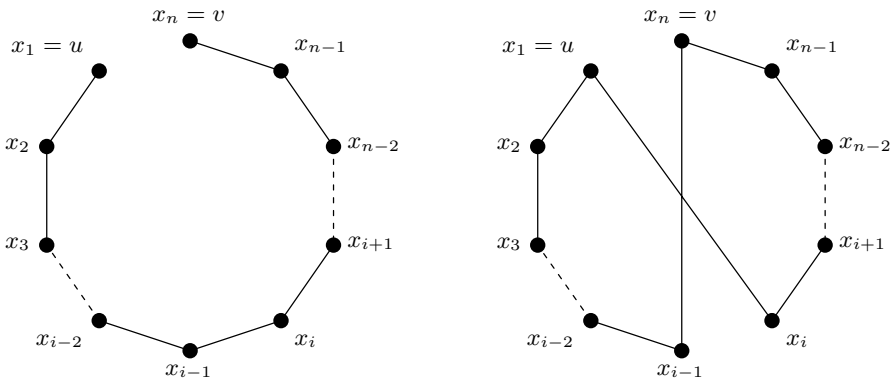

Fig. 1.8. Proof of Theorem 1.4.1

In general, it will not be much easier to decide whether $[G]$ is Hamiltonian. But if, for example, $[G]$ is a complete graph, G has to be Hamiltonian by

Theorem 1.4.1. Using this observation, we obtain the following two sufficient conditions for the existence of a Hamiltonian cycle due to Ore and Dirac [Ore60, Dir52], respectively.

Corollary 1.4.2. *Let G be a graph with $n \geq 3$ vertices. If $\deg u + \deg v \geq n$ holds for any two non-adjacent vertices u and v, then G is Hamiltonian.* □

Corollary 1.4.3. *Let G be a graph with $n \geq 3$ vertices. If each vertex of G has degree at least $n/2$, then G is Hamiltonian.* □

Bondy and Chvátal used their Theorem 1.4.1 to derive further sufficient conditions for the existence of a Hamiltonian cycle; in particular, they obtained the earlier result of Las Vergnas [Las72] in this way. We also refer the reader to [Har69, Ber73, Ber78, GoMi84, Chv85] for more results about Hamiltonian graphs.

Exercise 1.4.4. Let G be a graph with n vertices and m edges, and assume $m \geq \frac{1}{2}(n-1)(n-2)+2$. Use Corollary 1.4.2 to show that G is Hamiltonian.

Exercise 1.4.5. Determine the minimal number of edges a graph G with six vertices must have if $[G]$ is the complete graph K_6.

Exercise 1.4.6. If G is Eulerian, then $L(G)$ is Hamiltonian. Does the converse hold?

We now digress a little and look at one of the oldest problems in recreational mathematics, the *knight's problem*. This problem consists of moving a knight on a chessboard – beginning, say, in the upper left corner – such that it reaches each square of the board exactly once and returns with its last move to the square where it started.[5] As mathematicians tend to generalize everything, they want to solve this problem for chess boards of arbitrary size, not even necessarily square. Thus we look at boards having $m \times n$ squares. If we represent the squares of the chessboard by vertices of a graph G and connect two squares if the knight can move directly from one of them to the other, a solution of the knight's problem corresponds to a Hamiltonian cycle in G. Formally, we may define G as follows. The vertices of G are the pairs (i,j) with $1 \leq i \leq m$ and $1 \leq j \leq n$; as edges we have all sets $\{(i,j),(i',j')\}$ with $|i-i'|=1$ and $|j-j'|=2$ or $|i-i'|=2$ and $|j-j'|=1$. Most of the vertices of G have degree 8, except the ones which are too close to the border of the chess-board. For example, the vertices at the corners have degree 2. In our context of Hamiltonian graphs, this interpretation of the knight's

[5] It seems that the first known knight's tours go back more than a thousand years to the Islamic and Indian world around 840–900. The first examples in the modern European literature occur in 1725 in Ozanam's book [Oza25], and the first mathematical analysis of knight's tours appears in a paper presented by Euler to the Academy of Sciences at Berlin in 1759 [Eul66]. See the excellent website by Jelliss [Jel03]; and [Wil89], an interesting account of the history of Hamiltonian graphs.

problem is of obvious interest. However, solving the problem is just as well possible without looking at it as a graph theory problem. Figure 1.9 gives a solution for the ordinary chess-board of $8 \times 8 = 64$ squares; the knight moves from square to square according to the numbers with which the squares are labelled. Figure 1.9 also shows the Hamiltonian cycle in the corresponding graph.

1	52	25	38	3	54	15	40
24	37	2	53	26	39	4	55
51	64	27	12	29	14	41	16
36	23	62	45	60	43	56	5
63	50	11	28	13	30	17	42
22	35	46	61	44	59	6	57
49	10	33	20	47	8	31	18
34	21	48	9	32	19	58	7

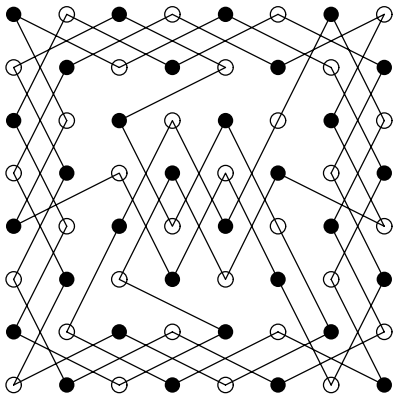

Fig. 1.9. A knight's cycle

The following theorem of Schwenk [Schw91] solves the knight's problem for arbitrary rectangular chessboards.

Result 1.4.7. *Every chessboard of size $m \times n$ (where $m \leq n$) admits a knight's cycle, with the following three exceptions:*

(a) *m and n are both odd;*

(b) *$m = 1, 2$ or 4;*

(c) *$m = 3$ and $n = 4, 6$ or 8.* □

The proof (which is elementary) is a nice example of how such problems can be solved recursively, combining the solutions for some small sized chessboards. Solutions for boards of sizes 3×10, 3×12, 5×6, 5×8, 6×6, 6×8, 7×6, 7×8 and 8×8 are needed, and these can easily be found by computer. The version of the knight's problem where no last move closing the cycle is required has also been studied; see [CoHMW92, CoHMW94].

Exercise 1.4.8. Show that knight's cycles are impossible for the cases (a) and (b) in Theorem 1.4.7. (Case (c) is more difficult.) Hint: For case (a) use the ordinary coloring of a chessboard with black and white squares; for (b) use the same coloring as well as another appropriate coloring (say, in red and green squares) and look at a hypothetical knight's cycle.

We close this section with a first look at one of the most fundamental problems in combinatorial optimization, the *travelling salesman problem* (for short, the *TSP*). This problem will later serve as our standard example of a *hard* problem, whereas most of the other problems we will consider are *easy*.[6]

Imagine a travelling salesman who has to take a circular journey visiting n cities and wants to be back in his home city at the end of the journey. Which route is – knowing the distances between the cities – the best one? To translate this problem into the language of graph theory, we consider the cities as the vertices of the complete graph K_n; any circular tour then corresponds to a Hamiltonian cycle in K_n. To have a measure for the expense of a route, we give each edge e a *weight* $w(e)$. (This weight might be the distance between the cities, but also the time the journey takes, or the cost, depending on the criterion subject to which we want to optimize the route.) The expense of a route then is the sum of the weights of all edges in the corresponding Hamiltonian cycle. Thus our problem may be stated formally as follows.

Problem 1.4.9 (travelling salesman problem, TSP). Consider the complete graph K_n together with a weight function $w\colon E \to \mathbb{R}^+$. Find a cyclic permutation $(1, \pi(1), \dots, \pi^{n-1}(1))$ of the vertex set $\{1, \dots, n\}$ such that

$$w(\pi) := \sum_{i=1}^{n} w(\{i, \pi(i)\})$$

is minimal. We call any cyclic permutation π of $\{1, \dots, n\}$ as well as the corresponding Hamiltonian cycle

$$1 \quad — \quad \pi(1) \quad — \quad \dots \quad — \quad \pi^{n-1}(1) \quad — \quad 1$$

in K_n a *tour*. An *optimal tour* is a tour π such that $w(\pi)$ is minimal among all tours.

Note that looking at all the possibilities for tours would be a lot of work: even for only nine cities we have $8!/2 = 20160$ possibilities. (We can always take the tour to begin at vertex 1, and fix the direction of the tour.) Of course it would be feasible to examine all these tours – at least by computer. But for 20 cities, we already get about 10^{17} possible tours, making this brute force approach more or less impossible.

It is convenient to view Problem 1.4.9 as a problem concerning matrices, by writing the weights as a matrix $W = (w_{ij})$. Of course, we have $w_{ij} = w_{ji}$ and $w_{ii} = 0$ for $i = 1, \dots, n$. The instances of a TSP on n vertices thus correspond to the symmetric matrices in $(\mathbb{R}^+)^{(n,n)}$ with entries 0 on the main diagonal. In the following example we have rounded the distances between the nine cities Aachen, Basel, Berlin, Dusseldorf, Frankfurt, Hamburg, Munich, Nuremberg and Stuttgart to units of 10 kilometers; we write $10w_{ij}$ for the rounded distance.

[6] The distinction between *easy* and *hard* problems can be made quite precise; we will explain this in Chapter 2.

Example 1.4.10. Determine an optimal tour for

$$\begin{array}{c} \\ Aa \\ Ba \\ Be \\ Du \\ Fr \\ Ha \\ Mu \\ Nu \\ St \end{array} \begin{array}{c} \begin{array}{ccccccccc} Aa & Ba & Be & Du & Fr & Ha & Mu & Nu & St \end{array} \\ \left(\begin{array}{ccccccccc} 0 & 57 & 64 & 8 & 26 & 49 & 64 & 47 & 46 \\ 57 & 0 & 88 & 54 & 34 & 83 & 37 & 43 & 27 \\ 64 & 88 & 0 & 57 & 56 & 29 & 60 & 44 & 63 \\ 8 & 54 & 57 & 0 & 23 & 43 & 63 & 44 & 41 \\ 26 & 34 & 56 & 23 & 0 & 50 & 40 & 22 & 20 \\ 49 & 83 & 29 & 43 & 50 & 0 & 80 & 63 & 70 \\ 64 & 37 & 60 & 63 & 40 & 80 & 0 & 17 & 22 \\ 47 & 43 & 44 & 44 & 22 & 63 & 17 & 0 & 19 \\ 46 & 27 & 63 & 41 & 20 & 70 & 22 & 19 & 0 \end{array} \right) \end{array}$$

An optimal tour and a tour which is slightly worse (obtained by replacing the edges *MuSt* and *BaFr* by the edges *MuBa* and *StFr*) are shown in Figure 1.10. We will study the TSP in Chapter 15 in detail, always illustrating the various techniques which we encounter using the present example.

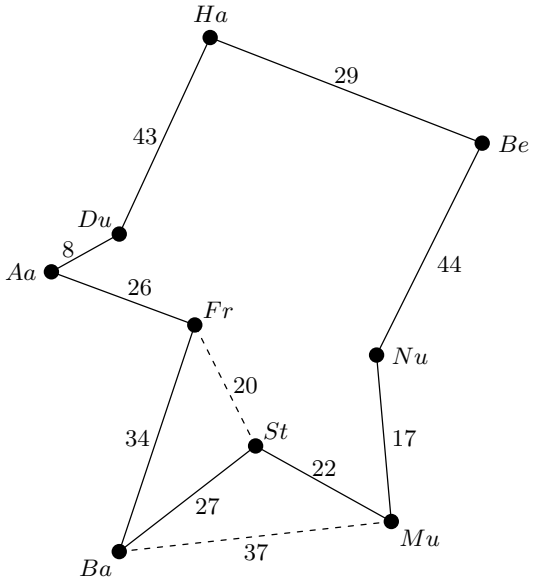

Fig. 1.10. Two tours for the TSP on 9 cities

Even though the number of possible tours grows exponentially with n, there still might be an easy method to solve the TSP. For example, the number of closed trails in a graph may also grow very fast as the number of edges

increases; but, as we will see in Chapter 2, it is still easy to find an Euler tour or to decide that no such tour exists. On the other hand, it is difficult to find Hamiltonian cycles. We will return to these examples in the next chapter to think about the complexity (that is, the degree of difficulty) of a problem.

1.5 Planar graphs

This section is devoted to the problem of drawing graphs in the plane. First, we need the notion of isomorphism. Two graphs $G = (V, E)$ and $G' = (V', E')$ are called *isomorphic* if there is a bijection $\alpha : V \to V'$ such that we have $\{a, b\} \in E$ if and only if $\{\alpha(a), \alpha(b)\} \in E'$ for all a, b in V. Let E be a set of line segments in three-dimensional Euclidean space and V the set of end points of the line segments in E. Identifying each line segment with the two-element set of its end points, we can consider (V, E) as a graph. Such a graph is called *geometric* if any two line segments in E are disjoint or have one of their end points in common.

Lemma 1.5.1. *Every graph is isomorphic to a geometric graph.*

Proof. Let $G = (V, E)$ be a graph on n vertices. Choose a set V' of n points in $\mathbb{R}^3$ such that no four points lie in a common plane (Why is that possible?) and map V bijectively to V'. Let E' contain, for each edge e in E, the line segment connecting the images of the vertices on e. It is easy to see that (V', E') is a geometric graph isomorphic to G. □

As we have only a plane piece of paper to draw graphs, Lemma 1.5.1 does not help us a lot. We call a geometric graph *plane* if its line segments all lie in one plane. Any graph isomorphic to a plane graph is called *planar*.[7] Thus, the planar graphs are exactly those graphs which can be drawn in the plane without additional points of intersection between the edges; see the comments after Figure 1.2. We will see that most graphs are not planar; more precisely, we will show that planar graphs can only contain comparatively few edges (compared to the number of vertices).

Let $G = (V, E)$ be a planar graph. If we omit the line segments of G from the plane surface on which G is drawn, the remainder splits into a number of connected open regions; the closure of such a region is called a *face*. The following theorem gives another famous result due to Euler [Eul52/53].

Theorem 1.5.2 (Euler's formula). *Let G be a connected planar graph with n vertices, m edges and f faces. Then $n - m + f = 2$.*

[7] In the definition of planar graphs, one often allows not only line segments, but curves as well. However, this does not change the definition of planarity as given above, see [Wag36]. For multigraphs, it is necessary to allow curves.

Proof. We use induction on m. For $m = 0$ we have $n = 1$ and $f = 1$, so that the statement holds. Now let $m \neq 0$. If G contains a cycle, we discard one of the edges contained in this cycle and get a graph G' with $n' = n$, $m' = m - 1$ and $f' = f - 1$. By induction hypothesis, $n' - m' + f' = 2$ and hence $n - m + f = 2$. If G is acyclic, then G is a tree so that $m = n - 1$, by Theorem 1.2.8; as $f = 1$, we again obtain $n - m + f = 2$. □

Originally, Euler's formula was applied to the vertices, edges and faces of a convex polyhedron; it is used, for example, to determine the five regular polyhedra (or *Platonic solids*, namely the tetrahedron, octahedron, cube, icosahedron and dodecahedron); see, for instance, [Cox73]. We will now use Theorem 1.5.2 to derive bounds on the number of edges of planar graphs. We need two more definitions. An edge e of a connected graph G is called a *bridge* if $G \setminus e$ is not connected. The *girth* of a graph containing cycles is the length of a shortest cycle.

Theorem 1.5.3. *Let G be a connected planar graph on n vertices. If G is acyclic, then G has precisely $n-1$ edges. If G has girth at least g, then G can have at most $\frac{g(n-2)}{g-2}$ edges.*

Proof. The first claim holds by Theorem 1.2.8. Thus let G be a connected planar graph having n vertices, m edges and girth at least g. Then $n \geq 3$. We use induction on n; the case $n = 3$ is trivial. Suppose first that G contains a bridge e. Discard e so that G is divided into two connected induced subgraphs G_1 and G_2 on disjoint vertex sets. Let n_i and m_i be the numbers of vertices and edges of G_i, respectively, for $i = 1, 2$. Then $n = n_1 + n_2$ and $m = m_1 + m_2 + 1$. As e is a bridge, at least one of G_1 and G_2 contains a cycle. If both G_1 and G_2 contain cycles, they both have girth at least g, so that by induction

$$m \;=\; m_1 + m_2 + 1 \;\leq\; \frac{g((n_1 - 2) + (n_2 - 2))}{g - 2} + 1 \;<\; \frac{g(n-2)}{g-2}.$$

If, say, G_2 is acyclic, we have $m_2 = n_2 - 1$ and

$$m = m_1 + m_2 + 1 \;\leq\; \frac{g(n_1 - 2)}{g - 2} + n_2 \;<\; \frac{g(n-2)}{g-2}.$$

Finally suppose that G does not contain a bridge. Then each edge of G is contained in exactly two faces. If we denote the number of faces whose border is a cycle consisting of i edges by f_i, we get

$$2m \;=\; \sum_i i f_i \;\geq\; \sum_i g f_i \;=\; gf,$$

as each cycle contains at least g edges. By Theorem 1.5.2, this implies

$$m + 2 = n + f \leq n + \frac{2m}{g} \quad \text{and hence} \quad m \leq \frac{g(n-2)}{g-2}. \qquad \square$$

In particular, we obtain the following immediate consequence of Theorem 1.5.3, since G is either acyclic or has girth at least 3.

Corollary 1.5.4. *Let G be a connected planar graph with n vertices, where $n \geq 3$. Then G contains at most $3n - 6$ edges.* □

Example 1.5.5. By Corollary 1.5.4, the complete graph K_5 is not planar, as a planar graph on five vertices can have at most nine edges. The complete bipartite graph $K_{3,3}$ has girth 4; this graph is not planar by Theorem 1.5.3, as it has more than eight edges.

Exercise 1.5.6. Show that the graphs which arise by omitting one edge e from either K_5 or $K_{3,3}$ are planar. Give plane realizations for $K_5 \setminus e$ and $K_{3,3} \setminus e$ which use straight line segments only.

For the sake of completeness, we will state one of the most famous results in graph theory, namely the characterization of planar graphs due to Kuratowski [Kur30]. We refer the reader to [Har69], [Aig84] or [Tho81] for the elementary but rather lengthy proof. Again we need some definitions. A *subdivision* of a graph G is a graph H which can be derived from G by applying the following operation any number of times: replace an edge $e = ab$ by a path $(a, x_1, \ldots, x_k, b)$, where $x_1, \ldots, x_k$ are an arbitrary number of *new* vertices; that is, vertices which were not in a previous subdivision. For convenience, G is also considered to be a subdivision of itself. Two graphs H and H' are called *homeomorphic* if they are isomorphic to subdivisions of the same graph G. Figure 1.11 shows a subdivision of $K_{3,3}$.

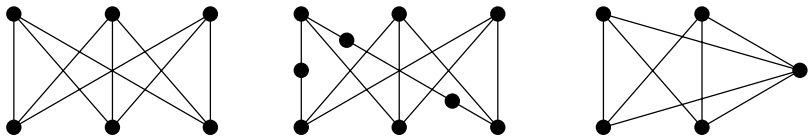

Fig. 1.11. $K_{3,3}$, a subdivision and a contraction

Exercise 1.5.7. Let (V, E) and (V', E') be homeomorphic graphs. Show that $|E| - |V| = |E'| - |V'|$.

Result 1.5.8 (Kuratowski's theorem). *A graph G is planar if and only if it does not contain a subgraph which is homeomorphic to K_5 or $K_{3,3}$.* □

In view of Example 1.5.5, a graph having a subgraph homeomorphic to K_5 or $K_{3,3}$ cannot be planar. For the converse we refer to the sources given above. There is yet another interesting characterization of planarity. If we identify two adjacent vertices u and v in a graph G, we get an *elementary contraction* of G; more precisely, we omit u and v and replace them by a new vertex w which is adjacent to all vertices which were adjacent to u or v before;[8] the

[8] Note that we introduce only one edge wx, even if x was adjacent to *both* u and v, which is the appropriate operation in our context. However, there are occasions

resulting graph is usually denoted by G/e, where $e = uv$. Figure 1.11 also shows a contraction of $K_{3,3}$. A graph G is called *contractible* to a graph H if H arises from G by a sequence of elementary contractions. For the proof of the following theorem see [Wag37], [Aig84], or [HaTu65].

Result 1.5.9 (Wagner's theorem). *A graph G is planar if and only if it does not contain a subgraph which is contractible to K_5 or $K_{3,3}$.*

Exercise 1.5.10. Show that the *Petersen graph* (see Figure 1.12, cf. [Pet98]) is not planar. Give three different proofs using 1.5.3, 1.5.8, and 1.5.9.

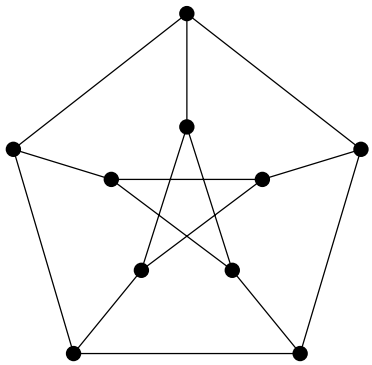

Fig. 1.12. The Petersen graph

Exercise 1.5.11. Show that the Petersen graph is isomorphic to the complement of the triangular graph T_5.

The isomorphisms of a graph G to itself are called *automorphisms*; clearly, they form a group, the *automorphism group* of G. In this book we will not study automorphisms of graphs, except for some comments on Cayley graphs in Chapter 9; we refer the reader to [Yap86], [Har69], or [CaLi91]. However, we give an exercise concerning this topic.

Exercise 1.5.12. Show that the automorphism group of the Petersen graph contains a subgroup isomorphic to the symmetric group S_5. Hint: Use Exercise 1.5.11.

Exercise 1.5.13. What is the minimal number of edges which have to be removed from K_n to get a planar graph? For each n, construct a planar graph having as many edges as possible.

---

where it is actually necessary to introduce two parallel edges wx instead, so that a contracted graph will in general become a multigraph.

The final exercise in this section shows that planar graphs have to contain many vertices of small degree.

Exercise 1.5.14. Let G be a planar graph on n vertices and denote the number of vertices of degree at most d by n_d. Prove

$$n_d \geq \frac{n(d-5)+12}{d+1}$$

and apply this formula to the cases $d = 5$ and $d = 6$. (Hint: Use Corollary 1.5.4.) Can this formula be strengthened?

Much more on planarity (including algorithms) can be found in the monograph by [NiCh88].

1.6 Digraphs

For many applications – especially for problems concerning traffic and transportation – it is useful to give a direction to the edges of a graph, for example to signify a one-way street in a city map. Formally, a *directed graph* or, for short, a *digraph* is a pair $G = (V, E)$ consisting of a finite set V and a set E of ordered pairs (a, b), where $a \neq b$ are elements of V. The elements of V are again called *vertices*, those of E *edges*; the term *arc* is also used instead of *edge* to distinguish between the directed and the undirected case. Instead of $e = (a, b)$, we again write $e = ab$; a is called the *start vertex* or *tail*, and b the *end vertex* or *head* of e. We say that a and b are *incident* with e, and call two edges of the form ab and ba *antiparallel*. To draw a directed graph, we proceed as in the undirected case, but indicate the direction of an edge by an arrow. *Directed multigraphs* can be defined analogously to multigraphs; we leave the precise formulation of the definition to the reader.

There are some operations connecting graphs and digraphs. Let $G = (V, E)$ be a directed multigraph. Replacing each edge of the form (a, b) by an undirected edge $\{a, b\}$, we obtain the *underlying multigraph* $|G|$. Replacing parallel edges in $|G|$ by a single edge, we get the *underlying graph* (G). Conversely, let $G = (V, E)$ be a multigraph. Any directed multigraph H with $|H| = G$ is called an *orientation* of G. Replacing each edge ab in E by two arcs (a, b) and (b, a), we get the *associated directed multigraph* $\overrightarrow{G}$; we also call $\overrightarrow{G}$ the *complete orientation* of G. The complete orientation of K_n is called the *complete digraph* on n vertices. Figure 1.13 illustrates these definitions.

We can now transfer the notions introduced for graphs to digraphs. There are some cases where two possibilities arise; we only look at these cases explicitly and leave the rest to the reader. We first consider trails. Thus let $G = (V, E)$ be a digraph. A sequence of edges $(e_1, \ldots, e_n)$ is called a *trail* if the corresponding sequence of edges in $|G|$ is a trail. We define walks, paths, closed trails and cycles accordingly. Thus, if $(v_0, \ldots, v_n)$ is the corresponding

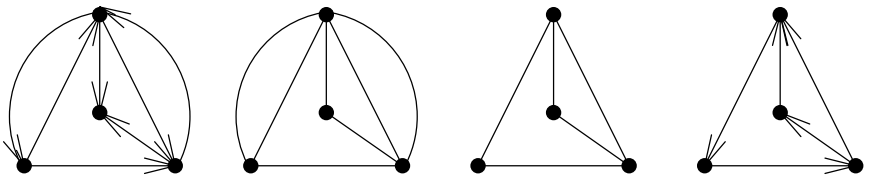

Fig. 1.13. (Directed) multigraphs

sequence of vertices, $v_{i-1}v_i$ or v_iv_{i-1} must be an edge of G. In the first case, we have a *forward edge*, in the second a *backward edge*. If a trail consists of forward edges only, it is called a *directed trail*; analogous definitions can be given for walks, closed trails, etc. In contrast to the undirected case, there may exist directed cycles of length 2, namely cycles of the form (ab, ba).

A *directed Euler tour* in a directed multigraph is a directed closed trail containing each edge exactly once. We want to transfer Euler's theorem to the directed case; this requires some more definitions. The *indegree* $d_{\text{in}}(v)$ of a vertex v is the number of edges with head v, and the *outdegree* $d_{\text{out}}(v)$ of v is the number of edges with tail v. A directed multigraph is called *pseudosymmetric* if $d_{\text{in}}(v) = d_{\text{out}}(v)$ holds for every vertex v. Finally, a directed multigraph G is called *connected* if $|G|$ is connected. We can now state the directed analogue of Euler's theorem. As the proof is quite similar to that of Theorem 1.3.1, we shall leave it to the reader and merely give one hint: the part *(b) implies (c)* needs a somewhat different argument.

Theorem 1.6.1. *Let G be a connected directed multigraph. Then the following statements are equivalent:*

(a) *G has a directed Euler tour.*
(b) *G is pseudosymmetric.*
(c) *The edge set of G can be partitioned into directed cycles.* □

For digraphs there is another obvious notion of connectivity besides simply requiring that the underlying graph be connected. We say that a vertex b of a digraph G is *accessible* from a vertex a if there is a directed walk with start vertex a and end vertex b. As before, we allow walks to have length 0 so that each vertex is accessible from itself. A digraph G is called *strongly connected* if each vertex is accessible from every other vertex. A vertex a from which every other vertex is accessible is called a *root* of G. Thus a digraph is strongly connected if and only if each vertex is a root.

Note that a connected digraph is not necessarily strongly connected. For example, a tree can never be strongly connected; here, of course, a digraph G is called a *tree* if $|G|$ is a tree. If G has a root r, we call G a *directed tree*,

an *arborescence* or a *branching* with root r. Clearly, given any vertex r, an undirected tree has exactly one orientation as a directed tree with root r.

We now consider the question which connected multigraphs can be oriented in such a way that the resulting graph is strongly connected. Such multigraphs are called *orientable*. Thus we ask which connected systems of streets can be made into a system of one-way streets such that people can still move from each point to every other point. The answer is given by the following theorem [Rob39].

Theorem 1.6.2 (Robbins' theorem). *A connected multigraph is orientable if and only if it does not contain any bridge.* □

We will obtain Theorem 1.6.2 by proving a stronger result which allows us to orient the edges one by one, in an arbitrary order. We need some more terminology. A *mixed multigraph* has edges which are either directed or undirected. (We leave the formal definition to the reader.) A *directed trail* in a mixed multigraph is a trail in which each oriented edge is a forward edge, but the trail might also contain undirected edges. A mixed multigraph is called *strongly connected* if each vertex is accessible from every other vertex by a directed trail. The theorem of Robbins is an immediate consequence of the following result due to Boesch and Tindell [BoTi80].

Theorem 1.6.3. *Let G be a mixed multigraph and e an undirected edge of G. Suppose that G is strongly connected. Then e can be oriented in such a way that the resulting mixed multigraph is still strongly connected if and only if e is not a bridge.*

Proof. Obviously, the condition that e is not a bridge is necessary. Thus suppose that e is an undirected edge of G for which neither of the two possible orientations of e gives a strongly connected mixed multigraph. We have to show that e is a bridge of $|G|$. Let u and w be the vertices incident with e, and denote the mixed multigraph we get by omitting e from G by H. Then there is no directed trail in H from u to w: otherwise, we could orient e from w to u and get a strongly connected mixed multigraph. Similarly, there is no directed trail in H from w to u.

Let S be the set of vertices which are accessible from u in H by a directed trail. Then u is, for any vertex $v \in S$, accessible from v in H for the following reason: u is accessible in G from v by a directed trail W; suppose W contains the edge e, then w would be accessible in H from u, which contradicts our observations above. Now put $T = V \setminus S$; as w is in T, this set is not empty. Then every vertex $t \in T$ is accessible from w in H, because t is accessible from w in G, and again: if the trail from w to t in G needed the edge e, then t would be accessible from u in H, and thus t would not be in T.

We now prove that e is the only edge of $|G|$ having a vertex in S and a vertex in T, which shows that e is a bridge. By definition of S, there cannot be an edge (s,t) or an edge $\{s,t\}$ with $s \in S$ and $t \in T$ in G. Finally, if

G contained an edge (t, s), then u would be accessible in H from w, as t is accessible from w and u is accessible from s. □

Mixed multigraphs are an obvious model for systems of streets. However, we will restrict ourselves to multigraphs or directed multigraphs for the rest of this book. One-way streets can be modelled by just using directed multigraphs, and ordinary two-way streets may then be represented by pairs of antiparallel edges.

We conclude this section with a couple of exercises.

Exercise 1.6.4. Let G be a multigraph. Prove that G does not contain a bridge if and only if each edge of G is contained in at least one cycle. (We will see another characterization of these multigraphs in Chapter 7: any two vertices are connected by two edge-disjoint paths.)

Exercise 1.6.5. Let G be a connected graph all of whose vertices have even degree. Show that G has a strongly connected, pseudosymmetric orientation.

Some relevant papers concerning (strongly connected) orientations of graphs are [ChTh78], [ChGT85], and [RoXu88].

Exercise 1.6.6. Prove that any directed closed walk contains a directed cycle. Also show that any directed walk W with start vertex a and end vertex b, where $a \neq b$, contains a directed path from a to b.

Hint: The desired path may be constructed from W by removing directed cycles.

1.7 An application: Tournaments and leagues

We conclude this chapter with an application of the factorizations mentioned before, namely setting up schedules for tournaments[9]. If we want to design a schedule for a tournament, say in soccer or basketball, where each of the $2n$ participating teams should play against each of the other teams exactly once, we can use a factorization $\mathbf{F} = \{F_1, \ldots, F_{2n-1}\}$ of K_{2n}. Then each edge $\{i, j\}$ represents the match between the teams i and j; if $\{i, j\}$ is contained in the factor F_k, this match will be played on the k-th day; thus we have to specify an ordering of the factors. If there are no additional conditions on the schedule, we can use any factorization. At the end of this section we will make a few comments on how to set up *balanced* schedules.

Of course, the above method can also be used to set up a schedule for a league (like, for example, the German soccer league), if we consider the two rounds as two separate tournaments. But then there is the additional problem of planning the home and away games. Look at the first round first. Replace

[9] This section will not be used in the remainder of the book and may be skipped during the first reading.

each 1-factor $F_k \in \mathbf{F}$ by an arbitrary orientation D_k of F_k, so that we get a factorization $\mathbf{D}$ of an orientation of K_{2n} – that is, a tournament as defined in Exercise 7.5.5 below. Then the home and away games of the first round are fixed as follows: if D_k contains the edge ij, the match between the teams i and j will be played on the k-th day of the season as a home match for team i. Of course, when choosing the orientation of the round of return matches, we have to take into account how the first round was oriented; we look at that problem later.

Now one wants home and away games to alternate for each team as far as possible. Hence we cannot just use an arbitrary orientation $\mathbf{D}$ of an arbitrary factorization $\mathbf{F}$ to set up the first round. This problem was solved by de Werra [deW81] who obtained the following results. Define a $(2n \times (2n-1))$-matrix $P = (p_{ik})$ with entries A and H as follows: $p_{ik} = H$ if and only if team i has a home match on the k-th day of the season; that is, if D_k contains an edge of the form ij. De Werra calls this matrix the *home-away pattern* of $\mathbf{D}$. A pair of consecutive entries p_{ik} and $p_{i,k+1}$ is called a *break* if the entries are the same; that is, if there are two consecutive home or away games; thus we want to avoid breaks as far as possible. Before determining the minimal number of breaks, an example might be useful.

Example 1.7.1. Look at the case $n = 3$ and use the factorization of K_6 shown in Figure 1.4; see Exercise 1.1.2. We choose the orientation of the five factors as follows: $D_1 = \{1\infty, 25, 43\}$, $D_2 = \{\infty 2, 31, 54\}$, $D_3 = \{3\infty, 42, 15\}$, $D_4 = \{\infty 4, 53, 21\}$ and $D_5 = \{5\infty, 14, 32\}$. Then we obtain the following matrix P:

$$P = \begin{pmatrix} A & H & A & H & A \\ H & A & H & A & H \\ H & A & A & H & A \\ A & H & H & A & H \\ H & A & H & A & A \\ A & H & A & H & H \end{pmatrix},$$

where the lines and columns are ordered $\infty, 1, \ldots, 5$ and $1, \ldots, 5$, respectively. Note that this matrix contains four breaks, which is best possible for $n = 3$ according to the following lemma.

Lemma 1.7.2. *Every oriented factorization of K_{2n} has at least $2n-2$ breaks.*

Proof. Suppose $\mathbf{D}$ has at most $2n-3$ breaks. Then there are at least three vertices for which the corresponding lines of the matrix P do not contain any breaks. At least two of these lines (the lines i and j, say) have to have the same entry (H, say) in the first column. As both lines do not contain any breaks, they have the same entries, and thus both have the form

$$H \quad A \quad H \quad A \quad H \ldots$$

Then, none of the factors D_k contains one of the edges ij or ji, a contradiction. (In intuitive terms: if the teams i and j both have a home match or both have an away match, they cannot play against each other.) □

The main result of de Werra shows that the bound of Lemma 1.7.2 can always be achieved.

Theorem 1.7.3. *The 1-factorization of K_{2n} given in Exercise 1.1.2 can always be oriented in such a way that the corresponding matrix P contains exactly $2n-2$ breaks.*

Sketch of proof. We give an edge $\{\infty, k\}$ of the 1-factor F_k of Exercise 1.1.2 the orientation $k\infty$ if k is odd, and the orientation ∞k if k is even. Moreover, the edge $\{k+i, k-i\}$ of the 1-factor F_k is oriented as $(k+i, k-i)$ if i is odd, and as $(k-i, k+i)$ if i is even. (Note that the orientation in Example 1.1.3 was obtained using this method.) Then it can be shown that the orientated factorization $\mathbf{D}$ of K_{2n} defined in this way has indeed exactly $2n-2$ breaks. The lines corresponding to the vertices ∞ and 1 do not contain any breaks, whereas exactly one break occurs in all the other lines. The comparatively long, but not really difficult proof of this statement is left to the reader. Alternatively, the reader may consult [deW81] or [deW88]. □

Sometimes there are other properties an optimal schedule should have. For instance, if there are two teams from the same city or region, we might want one of them to have a home game whenever the other has an away game. Using the optimal schedule from Theorem 1.7.3, this can always be achieved.

Corollary 1.7.4. *Let $\mathbf{D}$ be the oriented factorization of K_{2n} with exactly $2n-2$ breaks which was described in Theorem 1.7.3. Then, for each vertex i, there exists a vertex j such that $p_{ik} \neq p_{jk}$ for all $k = 1, \ldots, 2n-1$.*

Proof. The vertex *complementary* to vertex ∞ is vertex 1: team ∞ has a home game on the k-th day of the season (that is, ∞k is contained in D_k) if k is even. Then 1 has the form $1 = k - i$ for some odd i, so that 1 has an away game on that day. Similarly it can be shown that the vertex complementary to $2i$ (for $i = 1, \ldots, n-1$) is the vertex $2i+1$. □

Now we still have the problem of finding a schedule for the return round of the league. Choose oriented factorizations $\mathbf{D}_H$ and $\mathbf{D}_R$ for the first and second round. Of course, we want $\mathbf{D} = \mathbf{D}_H \cup \mathbf{D}_R$ to be a complete orientation of K_{2n}; hence ji should occur as an edge in $\mathbf{D}_R$ if ij occurs in $\mathbf{D}_H$. If this is the case, $\mathbf{D}$ is called a *league schedule* for $2n$ teams. For $\mathbf{D}_H$ and $\mathbf{D}_R$, there are home-away patterns P_H and P_R, respectively; we call $P = (P_H P_R)$ the home-away pattern of $\mathbf{D}$. As before, we want a league schedule to have as few breaks as possible. We have the following result.

Theorem 1.7.5. *Every league schedule $\mathbf{D}$ for $2n$ teams has at least $4n-4$ breaks; this bound can be achieved for all n.*

Proof. As P_H and P_R both have at least $2n-2$ breaks by Lemma 1.7.2, P obviously contains at least $4n-4$ breaks. A league schedule having exactly $4n-4$ breaks can be obtained as follows. By Theorem 1.7.3, there exists an oriented factorization $\mathbf{D}_H = \{D_1, \ldots, D_{2n-1}\}$ of K_{2n} with exactly $2n-2$ breaks. Put $\mathbf{D}_R = \{E_1, \ldots, E_{2n-1}\}$, where E_i is the 1-factor having the opposite orientation as D_{2n-i}; that is, $ji \in E_i$ if and only if $ij \in D_{2n-i}$. Then P_H and P_R each contain exactly $2n-2$ breaks; moreover, the first column of P_R corresponds to the factor E_1, and the last column of P_H corresponds to the factor D_{2n-1} which is the factor with the opposite orientation of E_1. Thus, there are no breaks between these two columns of P, and the total number of breaks is indeed $4n-4$. □

In reality, the league schedules described above are unwelcome, because the return round begins with the same matches with which the first round ended, just with home and away games exchanged. Instead, $\mathbf{D}_R$ is usually defined as follows: $\mathbf{D}_R = \{E_1, \ldots, E_{2n-1}\}$, where E_i is the 1-factor oriented opposite to D_i. Such a league schedule is called *canonical*. The following result can be proved analogously to Theorem 1.7.5.

Theorem 1.7.6. *Every canonical league schedule* $\mathbf{D}$ *for* $2n$ *teams has at least* $6n-6$ *breaks; this bound can be achieved for all* n. □

For more results about league schedules and related problems we refer to [deW80, deW82, deW88] and [Schr80]. In practice, one often has many additional secondary restrictions – sometimes even conditions contradicting each other – so that the above theorems are not sufficient for finding a solution. In these cases, computers are used to look for an adequate solution satisfying the most important requirements. As an example, we refer to [Schr92] who discusses the selection of a schedule for the soccer league in the Netherlands for the season 1988/89. Another actual application with secondary restrictions is treated in [deWJM90], while [GrRo96] contains a survey of some European soccer leagues.

Back to tournaments again! Although any factorization of K_{2n} can be used, in most practical cases there are additional requirements which the schedule should satisfy. Perhaps the teams should play an equal number of times on each of the n playing fields, because these might vary in quality. The best one can ask for in a tournament with $2n-1$ games for each team is, of course, that each team plays twice on each of $n-1$ of the n fields and once on the remaining field. Such a schedule is called a *balanced tournament design*. Every schedule can be written as an $n \times (2n-1)$ matrix $M = (m_{ij})$, where the entry m_{ij} is given by the pair xy of teams playing in round j on field i. Sometimes it is required in addition that, for the first as well as for the last n columns of M, the entries in each row of M form a 1-factor of K_{2n}; this is then called a *partitioned balanced tournament design* (PBTD) on $2n$ vertices. Obviously, such a tournament schedule represents the best possible solution concerning a uniform distribution of the playing fields. We give an example for $n = 5$, and

cite an existence result for PBDT's (without proof) which is due to Lamken and Vanstone [LaVa87, Lam87].

Example 1.7.7. The following matrix describes a PBTD on 10 vertices:

$$\left(\begin{array}{cccc|c|cccc} 94 & 82 & 13 & 57 & 06 & 23 & 45 & 87 & 91 \\ 83 & 95 & 46 & 02 & 17 & 84 & 92 & 05 & 63 \\ 56 & 03 & 97 & 81 & 42 & 67 & 01 & 93 & 85 \\ 12 & 47 & 80 & 96 & 53 & 90 & 86 & 14 & 72 \\ 07 & 16 & 25 & 43 & 98 & 15 & 37 & 26 & 04 \end{array}\right)$$

Result 1.7.8. *Let $n \geq 5$ and $n \notin \{9, 11, 15, 26, 28, 33, 34\}$. Then there exists a PBTD on $2n$ vertices.* □

Finally, we recommend the interesting survey [LaVa89] about tournament designs, which are studied in detail in the books of Anderson [And90, And97].