1

Introduction

Graph theory may be said to have its beginning in 1736 when EULER considered the (general case of the) **Königsberg bridge problem**: Does there exist a walk crossing each of the seven bridges of Königsberg exactly once? (Solutio Problematis ad geometriam situs pertinentis, *Commentarii Academiae Scientiarum Imperialis Petropolitanae* 8 (1736), pp. 128-140.)

It took 200 years before the first book on graph theory was written. This was "Theorie der endlichen und unendlichen Graphen" (Teubner, Leipzig, 1936) by KÖNIG in 1936. Since then graph theory has developed into an extensive and popular branch of mathematics, which has been applied to many problems in mathematics, computer science, and other scientific and not-so-scientific areas. For the history of early graph theory, see

N.L. BIGGS, R.J. LLOYD AND R.J. WILSON, "Graph Theory 1736 – 1936", Clarendon Press, 1986.

There are no standard notations for graph theoretical objects. This is natural, because the names one uses for the objects reflect the applications. Thus, for instance, if we consider a communications network (say, for email) as a graph, then the computers taking part in this network, are called nodes rather than vertices or points. On the other hand, other names are used for molecular structures in chemistry, flow charts in programming, human relations in social sciences, and so on.

These lectures study *finite graphs* and majority of the topics is included in

J.A. BONDY, U.S.R. MURTY, "Graph Theory with Applications", Macmillan, 1978.

R. DIESTEL, "Graph Theory", Springer-Verlag, 1997.

F. HARARY, "Graph Theory", Addison-Wesley, 1969.

D.B. WEST, "Introduction to Graph Theory", Prentice Hall, 1996.

R.J. WILSON, "Introduction to Graph Theory", Longman, (3rd ed.) 1985.

In these lectures we study *combinatorial aspects* of graphs. For more *algebraic* topics and methods, see

N. BIGGS, "Algebraic Graph Theory", Cambridge University Press, (2nd ed.) 1993.

C. GODSIL, G.F. ROYLE, "Algebraic Graph Theory", Springer, 2001.
and for *computational aspects*, see

S. EVEN, "Graph Algorithms", Computer Science Press, 1979.

In these lecture notes we mention several open problems that have gained respect among the researchers. Indeed, graph theory has the advantage that it contains easily formulated open problems that can be stated early in the theory. Finding a solution to any one of these problems is another matter.

Sections with a star ($*$) in their heading are optional.

Notations and notions

• For a finite set X, $|X|$ denotes its size (cardinality, the number of its elements).
• Let

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

and in general,

$$[i,n] = \{i,i+1,\ldots,n\}$$

for integers $i \leq n$.
• For a real number x, the **floor** and the **ceiling** of x are the integers

$$\lfloor x \rfloor = \max\{k \in \mathbb{Z} \mid k \leq x\} \text{ and } \lceil x \rceil = \min\{k \in \mathbb{Z} \mid x \leq k\}.$$

• A family $\{X_1, X_2, \ldots, X_k\}$ of subsets $X_i \subseteq X$ of a set X is a **partition** of X, if

$$X = \bigcup_{i \in [1,k]} X_i \quad \text{and} \quad X_i \cap X_j = \emptyset \text{ for all different } i \text{ and } j\,.$$

• For two sets X and Y,

$$X \times Y = \{(x,y) \mid x \in X,\ y \in Y\}$$

is their **Cartesian product**, and

$$X \triangle Y = (X \setminus Y) \cup (Y \setminus X)$$

is their **symmetric difference**. Here $X \setminus Y = \{x \mid x \in X, x \notin Y\}$.
• Two integers $n, k \in \mathbb{N}$ (often $n = |X|$ and $k = |Y|$ for sets X and Y) have the **same parity**, if both are even, or both are odd, that is, if $n \equiv k \pmod 2$. Otherwise, they have opposite parity.

Graph theory has abundant examples of **NP-complete problems**. Intuitively, a problem is in P [1] if there is an efficient (practical) algorithm to find a solution to it. On the other hand, a problem is in NP [2], if it is first efficient to guess a solution and then efficient to check that this solution is correct. It is conjectured (and not known) that P $\neq$ NP. This is one of the great problems in modern mathematics and theoretical computer science. If the guessing in NP-problems can be replaced by an efficient systematic search for a solution, then P=NP. For any one NP-complete problem, if it is in P, then necessarily P=NP.

[1] Solvable – by an algorithm – in polynomially many steps on the size of the problem instances.
[2] Solvable *nondeterministically* in polynomially many steps on the size of the problem instances.

1.1 Graphs and their plane figures

Let V be a *finite* set, and denote by

$$E(V) = \{\{u, v\} \mid u, v \in V,\ u \neq v\}.$$

the **2-sets** of V, i.e., subsets of two distinct elements.

DEFINITION. A pair $G = (V, E)$ with $E \subseteq E(V)$ is called a **graph** (**on** V). The elements of V are the **vertices** of G, and those of E the **edges** of G. The vertex set of a graph G is denoted by V_G and its edge set by E_G. Therefore $G = (V_G, E_G)$.

In literature, graphs are also called *simple graphs*; vertices are called *nodes* or *points*; edges are called *lines* or *links*. The list of alternatives is long (but still finite).

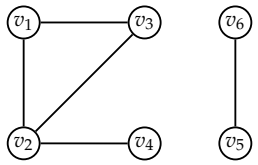
A pair $\{u, v\}$ is usually written simply as uv. Notice that then $uv = vu$. In order to simplify notations, we also write $v \in G$ and $e \in G$ instead of $v \in V_G$ and $e \in E_G$.

DEFINITION. For a graph G, we denote

$$\nu_G = |V_G| \text{ and } \varepsilon_G = |E_G|.$$

The number ν_G of the vertices is called the **order** of G, and ε_G is the **size** of G. For an edge $e = uv \in G$, the vertices u and v are its **ends**. Vertices u and v are **adjacent** or **neighbours**, if $uv \in G$. Two edges $e_1 = uv$ and $e_2 = uw$ having a common end, are **adjacent** with each other.

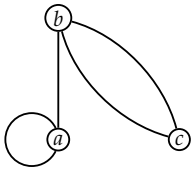
A graph G can be represented as a plane figure by drawing a line (or a curve) between the points u and v (representing vertices) if $e = uv$ is an edge of G. The figure on the right is a geometric representation of the graph G with $V_G = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ and $E_G = \{v_1v_2, v_1v_3, v_2v_3, v_2v_4, v_5v_6\}$.



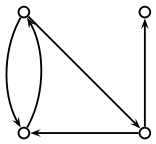
Often we shall omit the identities (names v) of the vertices in our figures, in which case the vertices are drawn as anonymous circles.

Graphs can be generalized by allowing **loops** vv and **parallel** (or **multiple**) **edges** between vertices to obtain a **multigraph** $G = (V, E, \psi)$, where $E = \{e_1, e_2, \ldots, e_m\}$ is a set (of symbols), and $\psi\colon E \to E(V) \cup \{vv \mid v \in V\}$ is a function that attaches an unordered pair of vertices to each $e \in E$: $\psi(e) = uv$.

Note that we can have $\psi(e_1) = \psi(e_2)$. This is drawn in the figure of G by placing two (parallel) edges that connect the common ends. On the right there is (a drawing of) a multigraph G with vertices $V = \{a, b, c\}$ and edges $\psi(e_1) = aa$, $\psi(e_2) = ab$, $\psi(e_3) = bc$, and $\psi(e_4) = bc$.



Later we concentrate on (simple) graphs.



DEFINITION. We also study **directed graphs** or **digraphs** $D = (V, E)$, where the edges have a direction, that is, the edges are ordered: $E \subseteq V \times V$. In this case, $uv \neq vu$.

The directed graphs have representations, where the edges are drawn as arrows. A digraph can contain edges uv and vu of *opposite directions*.

Graphs and digraphs can also be coloured, labelled, and weighted:

DEFINITION. A function $\alpha\colon V_G \to K$ is a **vertex colouring** of G by a set K of colours. A function $\alpha\colon E_G \to K$ is an **edge colouring** of G. Usually, $K = [1, k]$ for some $k \geq 1$.

If $K \subseteq \mathbb{R}$ (often $K \subseteq \mathbb{N}$), then α is a **weight function** or a **distance function**.

Isomorphism of graphs

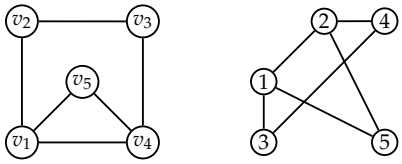
DEFINITION. Two graphs G and H are **isomorphic**, denoted by $G \cong H$, if there exists a bijection $\alpha\colon V_G \to V_H$ such that

$$uv \in E_G \iff \alpha(u)\alpha(v) \in E_H$$

for all $u, v \in G$.

Hence G and H are isomorphic if the vertices of H are renamings of those of G. Two isomorphic graphs enjoy the same graph theoretical properties, and *they are often identified*. In particular, all isomorphic graphs have the same plane figures (excepting the identities of the vertices). This shows in the figures, where we tend to replace the vertices by small circles, and talk of 'the graph' although there are, in fact, infinitely many such graphs.

Example 1.1. The following graphs are isomorphic. Indeed, the required isomorphism is given by $v_1 \mapsto 1$, $v_2 \mapsto 3$, $v_3 \mapsto 4$, $v_4 \mapsto 2$, $v_5 \mapsto 5$.



Isomorphism Problem. *Does there exist an efficient algorithm to check whether any two given graphs are isomorphic or not?*

The following table lists the number $2^{\binom{n}{2}}$ of all graphs on a given set of n vertices, and the number of all nonisomorphic graphs on n vertices. It tells that at least for computational purposes an efficient algorithm for checking whether two graphs are isomorphic or not would be greatly appreciated.

n	1	2	3	4	5	6	7	8	9
graphs	1	2	8	64	1024	32 768	2 097 152	268 435 456	$2^{36} > 6 \cdot 10^{10}$
nonisomorphic	1	2	4	11	34	156	1044	12 346	274 668

Other representations

Plane figures catch graphs for our eyes, but if a problem on graphs is to be *programmed*, then these figures are, to say the least, unsuitable. Integer matrices are ideal for computers, since every respectable programming language has array structures for these, and computers are good in crunching numbers.

Let $V_G = \{v_1, \ldots, v_n\}$ be ordered. The **adjacency matrix** of G is the $n \times n$-matrix M with entries $M_{ij} = 1$ or $M_{ij} = 0$ according to whether $v_i v_j \in G$ or $v_i v_j \notin G$. For instance, the graph in Example 1.1 has an adjacency matrix on the right. Notice that the adjacency matrix is always symmetric (with respect to its diagonal consisting of zeros).

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

A graph has usually many different adjacency matrices, one for each ordering of its set V_G of vertices. The following result is obvious from the definitions.

Theorem 1.1. *Two graphs G and H are isomorphic if and only if they have a common adjacency matrix. Moreover, two isomorphic graphs have exactly the same set of adjacency matrices.*

Graphs can also be represented by sets. For this, let $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ be a family of subsets of a set X, and define the **intersection graph** $G_{\mathcal{X}}$ as the graph with vertices $X_1, \ldots, X_n$, and edges $X_i X_j$ for all i and j ($i \neq j$) with $X_i \cap X_j \neq \emptyset$.

Theorem 1.2. *Every graph is an intersection graph of some family of subsets.*

Proof. Let G be a graph, and define, for all $v \in G$, a set

$$X_v = \{\{v, u\} \mid vu \in G\}.$$

Then $X_u \cap X_v \neq \emptyset$ if and only if $uv \in G$. □

Let $s(G)$ be the smallest size of a base set X such that G can be represented as an intersection graph of a family of subsets of X, that is,

$$s(G) = \min\{|X| \;\mid\; G \cong G_{\mathcal{X}} \text{ for some } \mathcal{X} \subseteq 2^X\}.$$

How small can $s(G)$ be compared to the order ν_G (or the size ε_G) of the graph? It was shown by KOU, STOCKMEYER AND WONG (1976) that it is algorithmically difficult to determine the number $s(G)$ – the problem is NP-complete.

Example 1.2. As yet another example, let $A \subseteq \mathbb{N}$ be a finite set of natural numbers, and let $G_A = (A, E)$ be the graph with $rs \in E$ if and only if r and s (for $r \neq s$) have a common divisor > 1. As an exercise, we state: *All graphs can be represented in the form G_A for some set A of natural numbers.*

1.2 Subgraphs

Ideally, given a nice problem the local properties of a graph determine a solution. In these situations we deal with (small) parts of the graph (subgraphs), and a solution can be found to the problem by combining the information determined by the parts. For instance, as we shall later see, the existence of an Euler tour is very local, it depends only on the number of the neighbours of the vertices.

Degrees of vertices

DEFINITION. Let $v \in G$ be a vertex a graph G. The **neighbourhood** of v is the set

$$N_G(v) = \{u \in G \mid vu \in G\}\,.$$

The **degree** of v is the number of its neighbours:

$$d_G(v) = |N_G(v)|\,.$$

If $d_G(v) = 0$, then v is said to be **isolated** in G, and if $d_G(v) = 1$, then v is a **leaf** of the graph. The **minimum degree** and the **maximum degree** of G are defined as

$$\delta(G) = \min\{d_G(v) \mid v \in G\} \quad \text{and} \quad \Delta(G) = \max\{d_G(v) \mid v \in G\}\,.$$

The following lemma, due to EULER (1736), tells that if several people shake hands, then the number of hands shaken is even.

Lemma 1.1 (Handshaking lemma). *For each graph G,*

$$\sum_{v \in G} d_G(v) = 2 \cdot \varepsilon_G\,.$$

Moreover, the number of vertices of odd degree is even.

Proof. Every edge $e \in E_G$ has two ends. The second claim follows immediately from the first one. □

Lemma 1.1 holds equally well for multigraphs, when $d_G(v)$ is defined as the number of edges that have v as an end, and when *each loop vv is counted twice.*

Note that the degrees of a graph G do not determine G. Indeed, there are graphs $G = (V, E_G)$ and $H = (V, E_H)$ on the same set of vertices that are *not* isomorphic, but for which $d_G(v) = d_H(v)$ for all $v \in V$.

Subgraphs

DEFINITION. A graph H is a **subgraph** of a graph G, denoted by $H \subseteq G$, if $V_H \subseteq V_G$ and $E_H \subseteq E_G$. A subgraph $H \subseteq G$ **spans** G (and H is a **spanning subgraph** of G), if every vertex of G is in H, i.e., $V_H = V_G$.

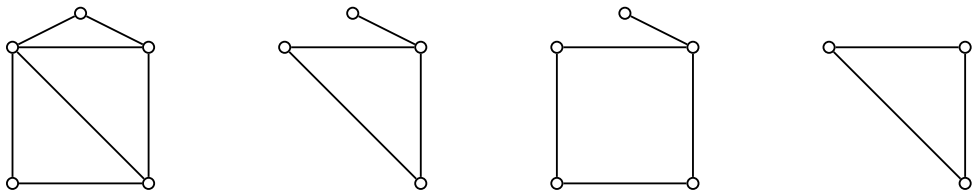
Also, a subgraph $H \subseteq G$ is an **induced subgraph**, if $E_H = E_G \cap E(V_H)$. In this case, H is **induced** by its set V_H of vertices.

In an induced subgraph $H \subseteq G$, the set E_H of edges consists of all $e \in E_G$ such that $e \in E(V_H)$. To each nonempty subset $A \subseteq V_G$, there corresponds a unique induced subgraph

$$G[A] = (A, E_G \cap E(A)) .$$

To each subset $F \subseteq E_G$ of edges there corresponds a unique spanning subgraph of G,

$$G[F] = (V_G, F) .$$



G subgraph spanning induced

For a set $F \subseteq E_G$ of edges, let

$$G-F = G[E_G \setminus F]$$

be the subgraph of G obtained by removing (only) the edges $e \in F$ from G. In particular, $G-e$ is obtained from G by removing $e \in G$.

Similarly, we write $G + F$, if each $e \in F$ (for $F \subseteq E(V_G)$) is added to G.

For a subset $A \subseteq V_G$ of vertices, we let $G-A \subseteq G$ be the subgraph induced by $V_G \setminus A$, that is,

$$G-A = G[V_G \setminus A] ,$$

and, *e.g.*, $G-v$ is obtained from G by removing the vertex v together with the edges that have v as their end.

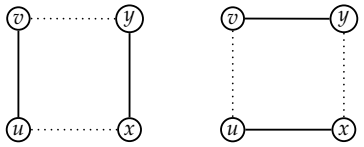
Reconstruction Problem. The famous open problem, **Kelly-Ulam problem** or the **Reconstruction Conjecture**, states that *a graph of order at least* 3 *is determined up to isomorphism by its vertex deleted subgraphs* $G-v$ *(*$v \in G$*)*: if there exists a bijection $\alpha\colon V_G \to V_H$ such that $G-v \cong H-\alpha(v)$ for all v, then $G \cong H$.

2-switches

DEFINITION. For a graph G, a 2-**switch** with respect to the edges $uv, xy \in G$ with $ux, vy \notin G$ replaces the edges uv and xy by ux and vy. Denote

$$G \xrightarrow{2s} H$$

if there exists a finite sequence of 2-switches that carries G to H.

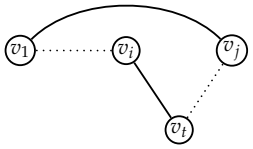


Note that if $G \xrightarrow{2s} H$ then also $H \xrightarrow{2s} G$ since we can apply the sequence of 2-switches in reverse order.

Before proving Berge's switching theorem we need the following tool.

Lemma 1.2. *Let G be a graph of order n with a degree sequence $d_1 \geq d_2 \geq \cdots \geq d_n$, where $d_G(v_i) = d_i$. Then there is a graph G' such that $G \xrightarrow{2s} G'$ with $N_{G'}(v_1) = \{v_2, \ldots, v_{d_1+1}\}$.*

Proof. Let $d = \Delta(G)$ $(= d_1)$. Suppose that there is a vertex v_i with $2 \leq i \leq d+1$ such that $v_1v_i \notin G$. Since $d_G(v_1) = d$, there exists a v_j with $j \geq d+2$ such that $v_1v_j \in G$. Here $d_i \geq d_j$, since $j > i$. Since $v_1v_j \in G$, there exists a v_t $(2 \leq t \leq n)$ such that $v_iv_t \in G$, but $v_jv_t \notin G$. We can now perform a 2-switch with respect to the vertices v_1, v_j, v_i, v_t. This gives a new graph H, where $v_1v_i \in H$ and $v_1v_j \notin H$, and the other neighbours of v_1 remain to be its neighbours.



When we repeat this process for all indices i with $v_1v_i \notin G$ for $2 \leq i \leq d+1$, we obtain a graph G' as required. □

Theorem 1.3 (BERGE **(1973)).** *Two graphs G and H on a common vertex set V satisfy $d_G(v) = d_H(v)$ for all $v \in V$ if and only if H can be obtained from G by a sequence of 2-switches.*

Proof. If $G \xrightarrow{2s} H$, then clearly H has the same degrees as G.

In converse, we use induction on the order ν_G. Let G and H have the same degrees. By Lemma 1.2, we have a vertex v and graphs G' and H' such that $G \xrightarrow{2s} G'$ and $H \xrightarrow{2s} H'$ with $N_{G'}(v) = N_{H'}(v)$. Now the graphs $G'-v$ and $H'-v$ have the same degrees. By the induction hypothesis, $G'-v \xrightarrow{2s} H'-v$, and thus also $G' \xrightarrow{2s} H'$. Finally, we observe that $H' \xrightarrow{2s} H$ by the 'reverse 2-switches', and this proves the claim. □

DEFINITION. Let $d_1, d_2, \ldots, d_n$ be a descending sequence of nonnegative integers, that is, $d_1 \geq d_2 \geq \cdots \geq d_n$. Such a sequence is said to be **graphical**, if there exists a graph $G = (V, E)$ with $V = \{v_1, v_2, \ldots, v_n\}$ such that $d_i = d_G(v_i)$ for all i.

Using the next result recursively one can decide whether a sequence of integers is graphical or not.

Theorem 1.4 (HAVEL **(1955),** HAKIMI **(1962)).** *A sequence $d_1, d_2, \ldots, d_n$ (with $d_1 \geq 1$ and $n \geq 2$) is graphical if and only if*

$$d_2 - 1,\ d_3 - 1, \ldots, d_{d_1+1} - 1,\ d_{d_1+2},\ d_{d_1+3}, \ldots, d_n \tag{1.1}$$

is graphical (when put into nonincreasing order).

Proof. ($\Leftarrow$) Consider G of order $n-1$ with vertices (and degrees)

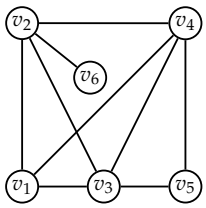
$$d_G(v_2) = d_2 - 1, \ldots, d_G(v_{d_1+1}) = d_{d_1+1} - 1,$$
$$d_G(v_{d_1+2}) = d_{d_1+2}, \ldots, d_G(v_n) = d_n$$

as in (1.1). Add a new vertex v_1 and the edges v_1v_i for all $i \in [2, d_{d_1+1}]$. Then in the new graph H, $d_H(v_1) = d_1$, and $d_H(v_i) = d_i$ for all i.

($\Rightarrow$) Assume $d_G(v_i) = d_i$. By Lemma 1.2 and Theorem 1.3, we can suppose that $N_G(v_1) = \{v_2, \ldots, v_{d_1+1}\}$. But now the degree sequence of $G - v_1$ is in (1.1). □

Example 1.3. Consider the sequence $s = 4, 4, 4, 3, 2, 1$. By Theorem 1.4,

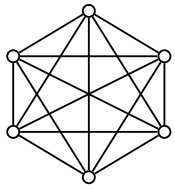
$$\begin{aligned} s \text{ is graphical} &\iff 3, 3, 2, 1, 1 \text{ is graphical} \\ &\quad\ \ 2, 1, 1, 0 \text{ is graphical} \\ &\quad\ \ 0, 0, 0 \text{ is graphical.} \end{aligned}$$



The last sequence corresponds to a graph with no edges, and hence also our original sequence s is graphical. Indeed, the graph G on the right has this degree sequence.

Special graphs

DEFINITION. A graph $G = (V, E)$ is **trivial**, if it has only one vertex, *i.e.*, $v_G = 1$; otherwise G is **nontrivial**.



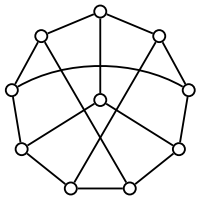
The graph $G = K_V$ is the **complete graph** on V, if every two vertices are adjacent: $E = E(V)$. All complete graphs of order n are isomorphic with each other, and they will be denoted by K_n.

The **complement** of G is the graph $\overline{G}$ on V_G, where $E_{\overline{G}} = \{e \in E(V) \mid e \notin E_G\}$. The complements $G = \overline{K}_V$ of the complete graphs are called **discrete graphs**. In a discrete graph $E_G = \emptyset$. Clearly, all discrete graphs of order n are isomorphic with each other.

A graph G is said to be **regular**, if every vertex of G has the same degree. If this degree is equal to r, then G is r-**regular** or **regular of degree** r.

A discrete graph is 0-regular, and a complete graph K_n is $(n-1)$-regular. In particular, $\varepsilon_{K_n} = n(n-1)/2$, and therefore $\varepsilon_G \leq n(n-1)/2$ for all graphs G that have order n.

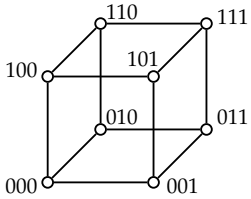
Many problems concerning (induced) subgraphs are algorithmically difficult. For instance, to find a maximal complete subgraph (a subgraph K_m of maximum order) of a graph is unlikely to be even in NP.



Example 1.4. The graph on the right is the **Petersen graph** that we will meet several times (drawn differently). It is a 3-regular graph of order 10.

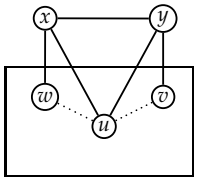
Example 1.5. Let $k \geq 1$ be an integer, and consider the set $\mathbb{B}^k$ of all binary strings of length k. For instance, $\mathbb{B}^3 = \{000, 001, 010, 100, 011, 101, 110, 111\}$. Let Q_k be the graph, called the k-**cube**, with $V_{Q_k} = \mathbb{B}^k$, where $uv \in Q_k$ if and only if the strings u and v differ in exactly one place.

The order of Q_k is $\nu_{Q_k} = 2^k$, the number of binary strings of length k. Also, Q_k is k-regular, and so, by the handshaking lemma, $\varepsilon_{Q_k} = k \cdot 2^{k-1}$. On the right we have the 3-cube, or simply the cube.



Example 1.6. Let $n \geq 4$ be any even number. We show by induction that there exists a 3-regular graph G with $\nu_G = n$. Notice that all 3-regular graphs have even order by the handshaking lemma.

If $n = 4$, then K_4 is 3-regular. Let G be a 3-regular graph of order $2m-2$, and suppose that $uv, uw \in E_G$. Let $V_H = V_G \cup \{x, y\}$, and $E_H = (E_G \setminus \{uv, uw\}) \cup \{ux, xv, uy, yw, xy\}$. Then H is 3-regular of order $2m$.



1.3 Paths and cycles

The most fundamental notions in graph theory are practically oriented. Indeed, many graph theoretical questions ask for optimal solutions to problems such as: find a shortest path (in a complex network) from a given point to another. This kind of problems can be difficult, or at least nontrivial, because there are usually choices what branch to choose when leaving an intermediate point.

Walks

DEFINITION. Let $e_i = u_i u_{i+1} \in G$ be edges of G for $i \in [1, k]$. The sequence $W = e_1 e_2 \dots e_k$ is a **walk of length** k **from** u_1 **to** u_{k+1}. Here e_i and e_{i+1} are compatible in the sense that e_i is adjacent to e_{i+1} for all $i \in [1, k-1]$.

We write, more informally,

$$W\colon u_1 \to u_2 \to \ldots \to u_k \to u_{k+1} \quad \text{or} \quad W\colon u_1 \overset{k}{\to} u_{k+1}\,.$$

Write $u \overset{\star}{\to} v$ to say that there is a walk of some length from u to v. Here we understand that $W\colon u \overset{\star}{\to} v$ *is always a specific walk*, $W = e_1e_2\ldots e_k$, although we sometimes do not care to mention the edges e_i on it. The length of a walk W is denoted by $|W|$.

DEFINITION. Let $W = e_1e_2\ldots e_k$ ($e_i = u_iu_{i+1}$) be a walk.

W is **closed**, if $u_1 = u_{k+1}$.

W is a **path**, if $u_i \neq u_j$ for all $i \neq j$.

W is a **cycle**, if it is closed, and $u_i \neq u_j$ for $i \neq j$ except that $u_1 = u_{k+1}$.

W is a **trivial path**, if its length is 0. A trivial path has no edges.

For a walk $W\colon u = u_1 \to \ldots \to u_{k+1} = v$, also

$$W^{-1}\colon v = u_{k+1} \to \ldots \to u_1 = u$$

is a walk in G, called the **inverse walk** of W.

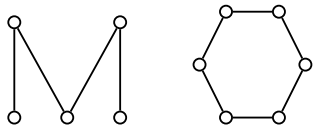
A vertex u is an **end** of a path P, if P starts or ends in u.

The **join** of two walks $W_1\colon u \overset{\star}{\to} v$ and $W_2\colon v \overset{\star}{\to} w$ is the walk $W_1W_2\colon u \overset{\star}{\to} w$. (Here the end v must be common to the walks.)

Paths P and Q are **disjoint**, if they have no vertices in common, and they are **independent**, if they can share only their ends.

Clearly, the inverse walk P^{-1} of a path P is a path (the **inverse path** of P). The join of two paths need not be a path.

A (sub)graph, which is a path (cycle) of length $k-1$ (k, resp.) having k vertices is denoted by P_k (C_k, resp.). If k is even (odd), we say that the path or cycle is **even** (**odd**). Clearly, all paths of length k are isomorphic. The same holds for cycles of fixed length.



Lemma 1.3. *Each walk $W\colon u \overset{\star}{\to} v$ with $u \neq v$ contains a path $P\colon u \overset{\star}{\to} v$, that is, there is a path $P\colon u \overset{\star}{\to} v$ that is obtained from W by removing edges and vertices.*

Proof. Let $W\colon u = u_1 \to \ldots \to u_{k+1} = v$. Let $i < j$ be indices such that $u_i = u_j$. If no such i and j exist, then W, itself, is a path. Otherwise, in $W = W_1W_2W_3\colon u \overset{\star}{\to} u_i \overset{\star}{\to} u_j \overset{\star}{\to} v$ the portion $U_1 = W_1W_3\colon u \overset{\star}{\to} u_i = u_j \overset{\star}{\to} v$ is a shorter walk. By repeating this argument, we obtain a sequence $U_1, U_2, \ldots, U_m$ of walks $u \overset{\star}{\to} v$ with $|W| > |U_1| > \cdots > |U_m|$. When the procedure stops, we have a path as required. (Notice that in the above it may very well be that W_1 or W_3 is a trivial walk.) □

DEFINITION. If there exists a walk (and hence a path) from u to v in G, let

$$d_G(u,v) = \min\{k \mid u \xrightarrow{k} v\}$$

be the **distance** between u and v. If there are no walks $u \xrightarrow{\star} v$, let $d_G(u,v) = \infty$ by convention. A graph G is **connected**, if $d_G(u,v) < \infty$ for all $u, v \in G$; otherwise, it is **disconnected**. The maximal connected subgraphs of G are its **connected components**. Denote

$$c(G) = \text{ the number of connected components of } G\,.$$

If $c(G) = 1$, then G is, of course, connected.

The maximality condition means that a subgraph $H \subseteq G$ is a connected component if and only if H is connected and there are no edges leaving H, i.e., for every vertex $v \notin H$, the subgraph $G[V_H \cup \{v\}]$ is disconnected. Apparently, every connected component is an induced subgraph, and

$$N^*_G(v) = \{u \mid d_G(v,u) < \infty\}$$

is *the* connected component of G that contains $v \in G$. In particular, the connected components form a partition of G.

Shortest paths

DEFINITION. Let G^α be an edge weighted graph, that is, G^α is a graph G together with a weight function $\alpha\colon E_G \to \mathbb{R}$ on its edges. For $H \subseteq G$, let

$$\alpha(H) = \sum_{e \in H} \alpha(e)$$

be the (total) **weight** of H. In particular, if $P = e_1 e_2 \dots e_k$ is a path, then its weight is $\alpha(P) = \sum_{i=1}^{k} \alpha(e_i)$. The **minimum weighted distance** between two vertices is

$$d^\alpha_G(u,v) = \min\{\alpha(P) \mid P\colon u \xrightarrow{\star} v\}\,.$$

In extremal problems we seek for optimal subgraphs $H \subseteq G$ satisfying specific conditions. In practice we encounter situations where G might represent

- a distribution or transportation network (say, for mail), where the weights on edges are *distances*, travel *expenses*, or *rates of flow* in the network;
- a system of channels in (tele)communication or computer architecture, where the weights present the rate of *unreliability* or *frequency of action* of the connections;
- a model of chemical bonds, where the weights measure molecular *attraction*.

In these examples we look for a subgraph with the smallest weight, and which connects two given vertices, or all vertices (if we want to travel around). On the other hand, if the graph represents a network of pipelines, the weights are volumes or capacities, and then one wants to find a subgraph with the maximum weight.

We consider the minimum problem. For this, let G be a graph with an integer weight function $\alpha\colon E_G \to \mathbb{N}$. In this case, call $\alpha(uv)$ the **length** of uv.

The **shortest path problem**: *Given a connected graph G with a weight function $\alpha\colon E_G \to \mathbb{N}$, find $d^{\alpha}_G(u,v)$ for given $u, v \in G$.*

Assume that G is a connected graph. Dijkstra's algorithm solves the problem for every pair u, v, where u is a fixed starting point and $v \in G$. Let us make the convention that $\alpha(uv) = \infty$, if $uv \notin G$.

Dijkstra's algorithm:

(i) Set $u_0 = u$, $t(u_0) = 0$ and $t(v) = \infty$ for all $v \neq u_0$.

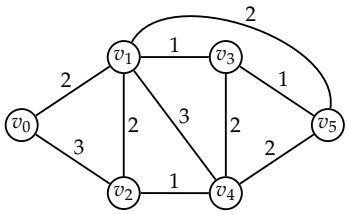
(ii) For $i \in [0, \nu_G - 1]$: for each $v \notin \{u_1, \dots, u_i\}$,

$$\text{replace } t(v) \text{ by } \min\{t(v), t(u_i) + \alpha(u_i v)\}\,.$$

Let $u_{i+1} \notin \{u_1, \dots, u_i\}$ be *any* vertex with the least value $t(u_{i+1})$.

(iii) Conclusion: $d^{\alpha}_G(u,v) = t(v)$.

Example 1.7. Consider the following weighted graph G. Apply Dijkstra's algorithm to the vertex v_0.



• $u_0 = v_0$, $t(u_0) = 0$, others are ∞.

• $t(v_1) = \min\{\infty, 2\} = 2$, $t(v_2) = \min\{\infty, 3\} = 3$, others are ∞. Thus $u_1 = v_1$.

• $t(v_2) = \min\{3, t(u_1) + \alpha(u_1 v_2)\} = \min\{3, 4\} = 3$, $t(v_3) = 2 + 1 = 3$, $t(v_4) = 2 + 3 = 5$, $t(v_5) = 2 + 2 = 4$. Thus choose $u_2 = v_3$.

• $t(v_2) = \min\{3, \infty\} = 3$, $t(v_4) = \min\{5, 3 + 2\} = 5$, $t(v_5) = \min\{4, 3 + 1\} = 4$. Thus set $u_3 = v_2$.

• $t(v_4) = \min\{5, 3 + 1\} = 4$, $t(v_5) = \min\{4, \infty\} = 4$. Thus choose $u_4 = v_4$.

• $t(v_5) = \min\{4, 4 + 1\} = 4$. The algorithm stops.

We have obtained:

$$t(v_1) = 2,\ t(v_2) = 3,\ t(v_3) = 3,\ t(v_4) = 4,\ t(v_5) = 4\,.$$

These are the minimal weights from v_0 to each v_i.

The steps of the algorithm can also be rewritten as a table:

v_1	**2**	-	-	-	-
v_2	3	3	**3**	-	-
v_3	∞	**3**	-	-	-
v_4	∞	5	5	**4**	-
v_5	∞	4	4	4	**4**

The correctness of Dijkstra's algorithm can verified be as follows.

Let $v \in V$ be any vertex, and let $P\colon u_0 \xrightarrow{\star} u \xrightarrow{\star} v$ be a shortest path from u_0 to v, where u is any vertex $u \neq v$ on such a path, possibly $u = u_0$. Then, clearly, the first part of the path, $u_0 \xrightarrow{\star} u$, is a shortest path from u_0 to u, and the latter part $u \xrightarrow{\star} v$ is a shortest path from u to v. Therefore, the length of the path P equals the sum of the weights of $u_0 \xrightarrow{\star} u$ and $u \xrightarrow{\star} v$. Dijkstra's algorithm makes use of this observation iteratively.