Daniela ROȘCA

SPECIAL MATHEMATICS:

COUNTING PROBLEMS, DISCRETE PROBABILITIES, GRAPHS THEORY

U.T.PRESS
Cluj-Napoca, 2016
ISBN 978-606-737-188-8

Daniela ROŞCA

SPECIAL MATHEMATICS:

COUNTING PROBLEMS,

DISCRETE PROBABILITIES,

GRAPHS THEORY

U.T. PRESS
Cluj-Napoca, 2016
ISBN 978-606-737-188-8

Editura U.T.PRESS
Str.Observatorului nr. 34
C.P.42, O.P. 2, 400775 Cluj-Napoca
Tel.:0264-401.999 / Fax: 0264 - 430.408
e-mail: utpress@biblio.utcluj.ro
www.utcluj.ro/editura

Director: Ing. Călin D. Câmpean

Recenzia: Prof.dr. Dorian Popa
Prof.dr. Neculae Vornicescu

ISBN 978-606-737-188-8

Preface

The course **Special Mathematics**, including the chapters *Combinatorics (counting methods), Discrete Probability Theory* and *Graph Theory* is included in the **Computer Science** curricula of the first semester of the first year. The above mentioned chapters are parts of the so-called *discrete mathematics*, also called *finite mathematics,* which study the mathematical structures that are fundamentally discrete, in the sense of not requiring the notion of continuity. Most of the objects studied in finite mathematics are finite or countable sets, such as integers or finite graphs.

Discrete mathematics has become popular in recent decades due to its applications to computer science. Concepts from discrete mathematics are useful to study or describe objects or problems in computer algorithms and programming languages.

Besides the topics included in this book, discrete mathematics include: Markov chains, Algorithmics (a study of methods of calculation), Computability and Complexity Theories (dealing with theoretical and practical limitations of algorithms), Logic (a study of reasoning), Set Theory, Discrete and Computational Geometry, Coding Theory and Cryptology, Discrete optimization, Linear Algebra, and so on.

This book is accessible to any person with the usual mathematical knowledge obtained in highschool.

Cluj-Napoca, October 2016 Daniela Roşca

Contents

Chapter 1

Counting Methods

In many problems in probabilities or computer science we need to count the number of outcome events, of objects or of the operations needed in an algorithm. This chapter presents some basic methods of dealing with the commonest counting problems.

1.1 Counting as a bijection. The pigeonhole principle

How can we count a finite set? In practice, we usually point to the objects in turn and say the numbers one, two, three and so on, stopping when we reach the last one. This "point-and-say" method actually means to construct a *function* from the set we want to count to the set $\mathbb{N}_n = \{1, 2, \ldots, n\}$. What is important is that the function must be a bijection: *each* object is counted *once*. So the first definition will be:

Definition 1.1 *Let $m \in \mathbb{N}^*$. The set A has m members if there exists a bijection from A to $\mathbb{N}_m$.*

In order to prove that by counting we cannot obtain different answers, we need the so-called "*pigeonhole principle*".

Theorem 1.1 (The pigeonhole principle) *Let $m \in \mathbb{N}$. Then for every $n \in \mathbb{N}$ the following statement is true:*

$$\textit{If there is an injection from } \mathbb{N}_n \textit{ to } \mathbb{N}_m, \textit{ then } n \leq m.$$

An equivalent formulation is the following: If $n > m$, then there is no injection from $\mathbb{N}_n$ to $\mathbb{N}_m$. This means that, if $n > m$ and n letters are put into m pigeonholes, then (at least) one pigeonhole will receive more than one letter.

Using the pigeonhole principle, the following definition determines a unique value for m.

Definition 1.2 *If there is a bijection between A and $\mathbb{N}_m$, then we say that A has **size** or **cardinality** m and we write*

$$|A| = m.$$

In the sequel we refer to set A with n members as an n**-set** and to a subset Y with k members as a k**-subset** of A.

Some applications of the pigeonhole principle

Example 1.1 *In any set of 13 or more persons, there are at least two whose birthdays fall in the same month.*

The "pigeonholes" here are the 12 months and the "letters" are the people.

Example 1.2 *At a party, whenever two persons are introduced, they shake hands with each other. Explain why there must be (at least) two persons at the party who shake the same number of hands.*

Solution. Let us denote by A the set of persons at the party, $m = |A|$, and denote

$$f(x) = \text{the number of persons in} A \text{ with which } x \text{ shakes hands.}$$

The possible values of $f(x)$ are $0, 1, \ldots, m-1$, so we can define the function $f : A \to B$, with $B = \{0, 1, \cdots, m-1\}$. We cannot apply the pigeonhole principle, because $|A| = |B|$. But, if there is a person x^* who shakes hands with $m-1$ people, then everyone shakes hands with x^*, and therefore there is nobody who doesn't shake hands. Therefore, the numbers 0 and $m-1$ cannot be both values of f, so $|B| \leq m-1$. From the pigeonhole principle we conclude that f is not injective, whence there exist two people x_1, x_2 such that $f(x_1) = f(x_2)$. So x_1 and x_2 shake the same number of hands.

This problem can also be formulated as follows: Show that in any set A of people there are two members of A who have the same number of friends in A. We suppose that $|A| \geq 2$ and that, if x is a friend of x', then x' is a friend of x.

Of course the "point-and-say" method is not sufficient to count more sophisticated sets, therefore we need to develop other methods.

1.2 First principles

The first principle is very simple and can be formulated mathematically as follows :

Principle 1 (the addition principle) If A and B are nonempty finite disjoint sets, then

$$|A \cup B| = |A| + |B|.$$

Obviously, the result can be extended to the union of any number of disjoint sets $A_1, A_2, \cdots, A_n$ in the obvious way, that is

$$|A_1 \cup A_2 \cup \cdots \cup A_n| = |A_1| + |A_2| + \cdots + |A_n|. \tag{1.1}$$

Remark 1.1 (counting interpretation) *The addition principle states that, when there are n cases such that the i-th case has k_i options, for $i = 1, 2, \ldots, n$, and no two of the cases have any options in common, the total number of options is*

$$k_1 + k_2 + \ldots + k_n.$$

A simple application of formula (1.1) is a more general form of the pigeonhole principle than the one given in Theorem 1.1. Suppose that some objects are distributed into n boxes and A_i represents the set of objects in box i, $1 \leq i \leq n$. Since the sets A_i are disjoint, the total number of objects is

$|A_1| + |A_2| + \cdots + |A_n|$. If we assume that no box contains more than k objects, the total number of objects is at most

$$k + k + \cdots + k = nk.$$

Expressing the reverse of the above statement, we get the **generalized pigeonhole principle**:

Theorem 1.2 *If m objects are distributed into n boxes and $m > nk$, then at least one box contains at least $k + 1$ objects.*

Example 1.3 *In a competition where the teams have five members, the rule is that the members of each team must have birthdays in the same month. How many persons are needed in order to guarantee that they can build a team.*

Solution. Applying the generalized pigeon principle, the number of persons is greater than $4 \cdot 12$, so the minimum number is 49.

■

In the case when the sets A_i are not necessarily disjoint, we have the following result.

Principle 2 (The *sieve* principle, or *inclusion-exclusion* principle) If $A_1, A_2, \ldots, A_n$ are finite sets, then

$$|A_1 \cup A_2 \cup \ldots \cup A_n| = \alpha_1 - \alpha_2 + \alpha_3 - \cdots (-1)^{n-1}\alpha_n, \tag{1.2}$$

where α_i is the sum of the cardinalities of the intersections of i sets:

$$\begin{aligned}
\alpha_1 &= \sum_{i=1}^{n} |A_i|, \\
\alpha_2 &= \sum_{1 \le i < j \le n} |A_i \cap A_j|, \\
&\cdots \\
\alpha_n &= \left| \bigcap_{i=1}^{n} A_i \right|.
\end{aligned}$$

The proof of the sieve principle can be made by induction on n.

An immediate consequence is the following result.

Corollary 1.3 *If $A_1, A_2, \ldots, A_n$ are subsets of a given set X of size N, then the number of members which are* not *in any of these subsets is*

$$\begin{aligned}
|X \setminus (A_1 \cup \ldots \cup A_n)| &= |X| - |A_1 \cup \cdots \cup A_n| \\
&= N - \alpha_1 + \alpha_2 - \ldots + (-1)^n \alpha_n.
\end{aligned} \tag{1.3}$$

An application of this principle is the *derangement problem* (see Section 1.9, p. 29).

1.3 Counting sets of pairs

In practice we need sometimes to count things which can be described naturally as pairs of objects. For example, we want to calculate the teaching load for the last year of the first cycle. In order to do this we make a table,

where each row corresponds to a student and each column corresponds to a subject. If a student x takes the course y then we mark the corresponding (x, y) position in the table. The total number of marks is the teaching load.

student \ course	C_1	C_2	C_3	$\cdots$	C_n	row total
S_1	✓	✓				·
S_2		✓			✓	·
S_3	✓	✓	✓			$r_x(S)$
$\vdots$						$\vdots$
S_m	✓				✓	·
column total	·	·	$c_y(S)$	$\cdots$	·	$\lvert S\rvert$

Our problem will be to count a subset S of the cartesian product $X \times Y$, where X is the set of students and Y is the set of courses.

The first method is to count the marks on the row x and find the row total $r_x(S)$, for each $x \in X$. The size of S can be obtained by adding all row totals,

$$|S| = \sum_{x \in X} r_x(S). \tag{1.4}$$

The second method is to count the marks in the column y and find the column total $c_y(S)$, for each $y \in Y$. In this case $|S|$ can be obtained by adding the column totals,

$$|S| = \sum_{y \in Y} c_y(S). \tag{1.5}$$

Obviously the two sums must coincide and these two expressions for $|S|$ are often used in practice to verify results or in theory to derive some properties of the cardinalities of X and Y.

Example 1.4 *In a class, 24 of the students are girls. Each girl knows 7 of the boys in the class and each boy knows 8 of the girls in the class. How many boys are in the class?*

Solution. Let us denote by n the number of boys. We make a table where we place the boys on columns and the girls on rows. Each time the boy x knows the girl y, we put a mark on the entry (x, y). By equating the total row and total column we get $8n = 7 \cdot 24$, so there are $n = 21$ boys.

Example 1.5 *Is it possible to find a collection of subsets of $\mathbb{N}_7$ with the property that each one has 5 elements and each element of $\mathbb{N}_7$ belongs to exactly 4 of the subsets?*

Solution. We make a table with entries $1, 2, \ldots, 7$ on rows and the required subsets $A_1, \ldots, A_k$ of $\mathbb{N}_7$ (k unknown at this stage) on columns. Whenever $i \in A_k$, we mark the position (k, i). The total row is $4 \cdot 7 = 28$, while the total column is $5 \cdot k$. Since, 5 is not a divisor of 28, the problem has a negative answer.

A simple consequence of the equalities (1.4) and (1.5) is the second basic principle:

Principle 3 (The multiplication principle) Given the sets X and Y, the size of $X \times Y$ is given by

$$|X \times Y| = |X| \cdot |Y|.$$

Obviously, the result can be extended to the product of any number of sets:

$$|X_1 \times X_2 \times \ldots \times X_n| = |X_1| \cdot |X_2| \cdot \ldots \cdot |X_n|.$$

Remark 1.2 (counting interpretation) *The multiplication principle states that when a procedure can be broken down into n steps, such that there are k_1 options for step 1, and such that after the completion of step $i-1$ $(i = 1, 2, \ldots, n)$ there are k_i options for step i, the number of ways performing the procedure is*

$$k_1 \cdot k_2 \cdot \ldots \cdot k_n.$$

1.4 Functions, words and selections

In most of the counting problems we have a number, say n, of *objects* or *things* which are *distributed* into k *classes* or *groups*. The number of ways in which this distribution can be made depends on whether

1. the objects can be distinguished or not;
2. the classes can be distinguished or not;
3. the order of objects in a class is relevant or not;
4. the order of classes is relevant or not;
5. the objects can be used more than once or not at all;
6. empty classes are allowed or not.

Exercise 1.4 *Find a practical example for each of the cases enumerated above.*

1.4.1 Ordered selections with repetitions (words)

For a given set $Y \neq \emptyset$, consider the set $\{f,\ f : \mathbb{N}_m \to Y\}$. The values taken by the function f determine the m-tuple

$$(f(1), f(2), \ldots, f(m)) \in Y \times \ldots \times Y := Y^m.$$

Conversely, each element of Y^m is an m-tuple $(y_1, y_2, \ldots, y_m)$ and corresponds to a function $f : \mathbb{N}_m \to Y$ defined by

$$f(1) = y_1,\ f(2) = y_2, \ldots, f(m) = y_m.$$

In conclusion, defining a function $f : \mathbb{N}_m \to Y$ is equivalent to choosing an element of Y^m. If we think of the members of Y as the letters of an alphabet, then the sequence $f(1), f(2), \ldots f(m)$ can be regarded as the m letters of a *word*. For example, if $Y = \{a, b, c, d\}$, the words *"bad"* and *"cab"* correspond, respectively to the functions $f, g : \mathbb{N}_3 \to Y$ given by

$$\begin{aligned} f(1) = b,\ f(2) = a,\ f(3) = d, \\ g(1) = c,\ g(2) = a,\ g(3) = b. \end{aligned}$$

Definition 1.3 *A **word** of length m in the* alphabet Y *is a function $f : \mathbb{N}_m \to Y$.*

Theorem 1.5 *Let X, Y be finite nonempty sets and let*

$$\mathcal{F} = \{f,\ f : X \to Y\}.$$

If $|X| = m$ and $|Y| = n$, then

$$|\mathcal{F}| = n^m.$$

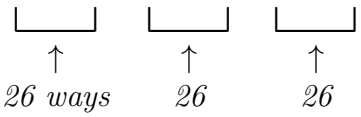
Proof. Any $f \in \mathcal{F}$ is determined by the m-tuple of its values

$$(f(x_1), f(x_2), \ldots, f(x_m)) \in Y^m.$$

Therefore $|\mathcal{F}| = |Y^m| = n^m$ ■

An equivalent formulation of this theorem is: *The number of words of length m in an alphabet Y of n symbols is n^m.*

Example 1.6 *In an alphabet with 26 letters, there are 26^3 words with 3 letters, assuming that there are no restrictions on spelling. An intuitive illustration of this number is the following:*



Here the three boxes represent the three "positions" of the letters of the word, each of which being "filled" independently in 26 ways.

So, there is another way of interpreting a function $f : \mathbb{N}_m \to Y$ (or, equivalently, a word of length m in the alphabet Y): To setup the word *"bad"*, we first select the letter b from the stock, then a, then d. We suppose we have an unlimited stock of letters. Thus, each word represent an **ordered selection** of the letters of the alphabet $Y = \{a, b, \ldots, z\}$, **with repetitions** allowed. In conclusion, we can state the following theorem.

Theorem 1.6 *The number of ordered selections with repetitions of m objects from the n-set Y is n^m.*

Example 1.7 *If X is an m-set, then the total number of subsets of X is 2^m.*

Solution. Suppose $X = \{x_1, x_2, \ldots, x_m\}$ and let Y be the alphabet $\{0, 1\}$. Any subset S of X corresponds to a word of length n in Y, defined by the function

$$f(i) = \begin{cases} 1, & \text{if } x_i \in S, \\ 0, & \text{if } x_i \notin S. \end{cases}$$

For example, if $m = 7$ and $S = \{x_2, x_4, x_5\}$, the word is 0101100. In conclusion, the number of distinct subsets of X is the same as the number of words of length m in the alphabet $\{0, 1\}$, which is 2^m.

1.4.2 Injections as ordered selections without repetitions

In some situations, we are allowed to have only one object of each kind. For instance, if we select a team, no player can be selected more than once.

To perform an *ordered selection* means to define a function $f : \mathbb{N}_m \to Y$, with $f(i)$ being the i-th selected object. To allow repetitions means that it is possible to have $f(j) = f(k)$ for $j \neq k$. Not allowing repetitions means to impose that f is *injective.* We are interested to determine the number of injections. This number is given in the following theorem.

Theorem 1.7 *The number of ordered selections without repetitions, of m objects from a set Y of size n is the same as the number of injections $f : \mathbb{N}_m \to Y$ and is given by*

$$n(n-1)(n-2)\dots(n-m+1) = \frac{n!}{(n-m)!}. \tag{1.6}$$

Proof. The value $f(1)$ can be chosen in n ways. If no repetitions is allowed, the value $f(2)$ can be chosen in $n-1$ ways, and finally, the value $f(m)$ can be chosen in $n-m+1$ ways. The product rule yields formula (1.6). ■

Remark 1.3 *The numbers $n(n-1)\dots(n-m+1)$ appear so frequently in counting problems, that they have acquired their own names:*

$$n^{\underline{m}} := n(n-1)(n-2)\dots(n-m+1)$$

*are called **falling factorials** (of n of length m). Analogously, we set*

$$n^{\overline{m}} := n(n+1)(n-2)\dots(n+m-1)$$

*and call $n^{\overline{m}}$ **rising factorials**.*

Remark 1.4 *The number $n^{\underline{m}}$ is in fact the number of words of length m all of whose entries, elements of the n- set Y, are different. For this reason, these words are also called m- **permutations** of Y. For example,* 1234 *and* 6512 *are two* 4- *permutations of $Y = \{1, 2, \dots, 6\}$.*

Example 1.8 *How many four-letter words can be made from an alphabet of 9 symbols if there are no restrictions on spelling, except that no letter can be used more than once?*

Answer. $9^{\underline{4}} = 9 \cdot 8 \cdot 7 \cdot 6 = 3024$.

Example 1.9 *For a class of 30 students, in how many ways can one give prizes at the end of the year, knowing that there is one "first prize", one "second prize" and one "third prize"?*

Answer. $30^{\underline{3}} = 30 \cdot 29 \cdot 28 = 24360$.

1.4.3 Permutations

Suppose a number of objects are placed randomly in a row. Mathematically this means defining a function from the set of "positions" occupied by the objects, to the set of objects.

Definition 1.4 *A **permutation** of a finite set $X \neq \emptyset$ is a bijection $\sigma : X \to X$.*

To simplify the writing, one frequently takes instead of an arbitrary n-set X, the set $\mathbb{N}_n$. The set of all permutations of $\mathbb{N}_n$ will be denoted by Π_n. As an immediate consequence of Theorem 1.7, we have

Theorem 1.8 *The number of permutation of* $\mathbb{N}_n$ *is*

$$|\Pi_n| = n(n-1)\cdot\ldots\cdot 2\cdot 1 = n\,! \tag{1.7}$$

A permutation $\pi \in \Pi_n$ will be denoted by

$$(\pi(1), \pi(2), \ldots, \pi(n))\,.$$

Example 1.10 *Four persons can form a queue in* $4! = 24$ *ways.*

Definition 1.5 *A permutation* $\pi \in \Pi_n$ *is called a* ***derangement*** *if* $\pi(i) \neq i$ *for all* $i \in \mathbb{N}_n$*.*

Thus $(3, 2, 1)$ is not a derangement of $\mathbb{N}_3$, but $(2, 3, 1)$ is a derangement. The number of derangements is calculated in Section 1.9, p. 29.

1.4.4 Unordered selections without repetitions

Suppose we have a set X with n objects and we select k of them. The result of the selection is a subset Y of size k. In this model, it is the result of the selection which is important, rather the process of the selection. Also, there is no possibility of repetitions (each member of X is either in Y or not, and no member can be selected twice). For example, from a class of 30 students, we want to select 3 students for a free excursion around the world. Of course nobody will be interested in the order of selection, but only in the final selection.

The number of unordered selections without repetitions of k objects from a set X, of size n, is the number of k- subsets of an n- set X. In general, this number is denoted by $\binom{n}{k}$ (to be read "n choose k") and is called **binomial number**[1].

Theorem 1.9 *The binomial number* $\binom{n}{k}$ *has the expression*

$$\binom{n}{k} = \frac{n(n-1)\ldots(n-k+1)}{k!} = \frac{n^{\underline{k}}}{k!} = \frac{n!}{k!(n-k)!}. \tag{1.8}$$

Proof. The k elements of the k-subset can be selected in $n(n-1)\cdots(n-k+1)$ ways, as we did for the ordered selections without repetitions. These k object can be permuted between themselves in $k!$ ways. Since the order does not matter, we choose only one of the $k!$ ways, so finally the number of k-subsets of an n-set will be

$$\frac{n(n-1)\ldots(n-k+1)}{k!}.$$

■

An immediate property is $\binom{n}{k} = \binom{n}{n-k}$.

[1]Most of the Romanian books use the notation C_n^k for the binomial number $\binom{n}{k}$.

Remark 1.5 *The binomial number can be defined in a more general case as*

$$\binom{x}{k} = \frac{x(x-1)\dots(x-k+1)}{k!}, \tag{1.9}$$

for $x \in \mathbb{R},\ k \in \mathbb{N} \cup \{0\}$.

Example 1.11 *One can we select a team of* 7 *students, out of a group of 30 students, to make a trip, in* $\binom{30}{7} = 2035800$ *ways.*

Example 1.12 ***(Counting bit strings with exactly*** k ***zeros)*** *What is the number of bit strings of length* n *with exactly* k *zeros?*

Solution. Each bit string is determined by choosing a k-subset of the n-set of "positions". Thus, 0 s are placed in these k positions and 1 s in the remaining $n-k$ positions, as we did in Example 1.7, p. 12. In conclusion, the required number is $\binom{n}{k}$.

1.4.5 Unordered selections with repetitions

We are interested now in the case of unordered selections with repetitions.

Example 1.13 *List the unordered selections of four objects of the set* $\{a, b, c\}$, *with repetitions allowed.*

Solution. There are 15 such selections, as follows:

aaaa	aaab	aaac	aabb	aabc
aacc	abbb	abbc	abcc	accc
bbbb	bbbc	bbcc	bccc	cccc

How can we count them in general? The idea is to represent such selections as words in the alphabet $\{0, 1\}$. For example, the word associated to the selection *abcc* will be 101011. The zeroes are *markers* which separate the kinds of objects and the ones tell us how many of each object there are. If there is no object of a kind, we do not put anything. Thus, in our example, this association will be

a		b		c	c
1	0	1	0	1	1

The problem is reduced to the problem of counting the number of bit strings of length six, which contain exactly two zeros. This number is $\binom{6}{2} = 15$ (see Example 1.12, p. 15).

In general, we can state the following theorem.

Theorem 1.10 *The number of unordered selections, with repetitions, of* k *objects from a set of* n *objects is* $\binom{n+k-1}{k}$.

Proof. Since the selections are unordered, we may arrange objects so that, within each selection, all the objects of a given kind come first, followed by the objects of another kind, and so on. When this has been done, we can assign to each selection a word of length $n + (k-1)$ in the alphabet $\{0, 1\}$, as we explained in the example above. Suppose we have n_i objects of the i-th kind. The first n_1 letters of the word will be 1, followed by a single 0, the next n_2 letters will be again 1, followed by a single 0, and so on. If

for a certain i we have $n_i = 0$, then we will have two consecutive markers. For instance, for the selections *accc* and *cccc* the words will be 100111 and 000111, respectively. The association defined by this rule is a bijection from the set of selections to the set of words of length $n + k - 1$ which contain exactly $n - 1$ zeroes. In conclusion, the number of words is

$$\binom{n+k-1}{n-1} = \binom{n+k-1}{k}. \tag{1.10}$$

■

Remark 1.6 *One can conclude that defining an unordered selection with repetitions of k objects from a set of n objects is equivalent to placing k identical balls in n distinct boxes. Indeed, the construction of the bit-string can be done by using a 1 for each ball and 0 as a marker which separates the boxes.*

Example 1.14 *We have six kinds of cookies. How many different plates containing ten cookies can we make? (supposing we have at least 10 cookies of each kind).*

Answer. $\binom{15}{5} = 3003$.

Example 1.15 *Show that, if three indistinguishable dice are rolled, there are 56 possible outcomes. How many outcomes do we have when n such dice are rolled?*

Answer. This number is $\binom{n+5}{5}$, in particular $\binom{8}{5} = 56$.

Example 1.16 *Suppose that the expression $(x_1 + x_2 + x_3)^n$ is expanded and the terms are collected, according to the usual rule of algebra. What is the number of terms in the resulting formula?*

Answer. We have n ones (objects, corresponding to the monomials x_i), and 2 markers, separating x_1 from x_2 and respectively x_2 from x_3. Therefore, the number of bit strings of length $n + 2$ containing 2 zeroes and n ones will be $\binom{n+2}{n}$.

1.5 Designs

A manufacturer has developed a new product and wishes to evaluate n varieties of it by asking some consumer to test them. It may be impracticable for each consumer to test all the varieties, so it would be reasonable to impose the following conditions:

(i) Each consumer should test the same number k of varieties,

(ii) Each variety should be tested by the same number r of consumers.

For example, if $n = 8,\ k = 4,\ r = 3$, a possible scheme would be to ask 6 people to test the varieties, as follows:

$$1234 \quad 5678 \quad 1357 \quad 2468 \quad 1247 \quad 3658,$$

where the numbers $1, 2, \ldots, 8$ represent the varieties.

Definition 1.6 *Let X be an n-set. A set $\mathcal{B}$ of k-subsets of X is a **design** with parameters (n, k, r) if each member of X belongs to exactly r of the subsets of $\mathcal{B}$.*

*An element $B \in \mathcal{B}$ is called a **block** of the design.*

As we have already seen in Example 1.5, p. 10, there is no design with parameters $(7, 5, 4)$, so it is necessary to find conditions on the parameters (n, k, r) for the existence of a design with these parameters.

So, let us consider an n-set X and let $\mathcal{C}$ be a set of k-subsets of X, not necessary a design. As in Section 1.3, we make a table (for the particular case $n = 6, k = 3$. A mark at the position (x, C_i) means $x \in C_i$. For each $x \in X$, let $r(x)$ be the number of times x occurs as a member of a subset C_i.

x \ C	C_1	C_2	C_3	C_4	$r(x)$
1	✓	✓		✓	3
2	✓		✓		2
3		✓			1
4	✓		✓	✓	3
5			✓		1
6		✓		✓	2
	3	3	3	3	

If we suppose that each C_i is a k-subset of X, then on each column the total will be k. Equating the row-total and the column-total we get

$$\sum_{x \in X} r(X) = |\,\mathcal{C}| \cdot k. \tag{1.11}$$

Suppose now we have a design B with parameters (n, k, r). In this case in the equality (1.11) we replace $r(x) = r$, obtaining

$$n \cdot r = k \cdot b, \tag{1.12}$$

where $b = |\mathcal{B}|$ is the number of blocks.

Furthermore, the total number of k-subsets of X is $\binom{n}{k}$, so the number b of blocks should be smaller that this number. In conclusion

$$b = \frac{nr}{k} \leq \binom{n}{k}. \tag{1.13}$$

It can be proven that conditions (1.12) and (1.13) are also sufficient for the existence of a design. More precisely, the following theorem is true. For the proof of the reverse implication and other properties of designs, see [2].

Theorem 1.11 *There exists a design with parameters (n, k, r) if and only if*

$$k \mid nr \quad \textit{and} \quad r \leq \binom{n-1}{k-1}.$$

1.6 Partitions and distributions

In this section we will deal with partitions of a set into subsets (set partition), integer partitions, and distributions of a set of objects into a set of boxes.

1.6.1 Partitions of a set into subsets (set partitions)

Definition 1.7 *Let $I \neq \emptyset$ be a set of indices, finite or infinite. A **partition** of a set X is a family $\mathcal{H} = \{X_i,\ i \in I\}$ of* nonempty *subsets of X, such that*

1. $X = \bigcup_{i \in I} X_i$,
2. *X_i are pairwise disjoint.*

*This means that every element $x \in X$ must belong to one and only one subset X_i. The subsets X_i are also called the **classes** or **parts** of the partition $\mathcal{H}$.*

Remark 1.7 *Defining a partition of an $n-$set X into k parts is equivalent to placing n distinct balls in k identical boxes such that no box remains empty (surjective placement).*

Example 1.17 *The family $\mathcal{H} = \{X_1, X_2, X_3, X_4\}$, where*

$$X_1 = \{1, 2, 9\},\ X_2 = \{3, 5, 7, 10\},\ X_3 = \{4\},\ X_4 = \{6, 8\}$$

is a partition of $\mathbb{N}_{10}$.

Other example of partition is given in Appendix A , p. 134, where we also showed how an equivalence relation determines a partition.

We are interested to calculate the number of partitions. A first result regarding this number is the following.

Theorem 1.12 *Let $S(n, k)$ denote the number of partitions of an n-set X into k parts (also called k-*partitions*), $1 \leq k \leq n$. Then*

1. $S(n, 1) = 1$,
2. $S(n, n) = 1$,
3. $S(n, k) = S(n-1, k-1) + kS(n-1, k)$, *for* $2 \leq k \leq n-1$.

Proof. First, $S(n, 1) = 1$ since there is only one partition of X into one part, namely $\mathcal{H} = \{X\}$. Then $S(n, n) = 1$, since there is again only one partition of X into n classes, namely $\mathcal{H} = \{\{x\},\ x \in X\}$.

It remains to prove 3. For this, let us fix $z \in X$. With z fixed, we distinguish the following types of partitions:

Type 1: partitions in which the set $\{z\}$ is a part,

Type 2: partitions in which the part containing z has other members.

Let us calculate the number of partitions of type 1. If the set $\{z\}$ is removed from the partition, we obtain a partition of the $(n-1)$-set $X \setminus \{z\}$ into $k-1$ parts, and there are $S(n-1, k-1)$ such partitions. Conversely, given such a partition, we can restore the part $\{z\}$.

We calculate now the number of partitions of type 2. Suppose $\mathcal{H}$ is a partition of type 2 with parts $X_1, X_2, \ldots, X_k$. We can define a pair of objects $(i, \mathcal{H}_0)$ with i taken such that $z \in X_i$ and with $\mathcal{H}_0$ taken as a partition of the $(n-1)$-set $X \setminus \{z\}$ into the k parts $X_1, \ldots, X_{i-1}, X_i \setminus \{z\}, X_{i+1}, \ldots, X_k$. There are k possible values of i and $S(n-1, k)$ possible partitions $\mathcal{H}_0$, so we have $kS(n-1, k)$ such pairs. Conversely, given such a pair $(i, \mathcal{H}_0)$, we can restore z to the part X_i and recover $\mathcal{H}$.

Since for a fixed z a partition can be either of type 1 or of type 2, we add these numbers and obtain the conclusion 3. ■

Remark 1.8 *The numbers $S(n,k)$ are known as the* Stirling[2] numbers of second kind. *Unfortunately there is no simple formula for these numbers, but using the recursion 3 one can construct the following table:*

$n\backslash k$	1	2	3	4	5	6	7	8	$\cdots$
1	1								
2	1	1							
3	1	3	1						
4	1	7	6	1					
5	1	15	25	10	1				
6	1	31	90	65	15	1			
7	1	63	301	350	140	21	1		
8	1	127	966	1701	1050	266	28	1	
$\cdots$									

Example 1.18 *The set $\mathbb{N}_5$ has the following 2-partitions, where we replaced $\cup$ with $+$ and we omitted the braces and commas for simplicity:*

$$\begin{aligned} 12345 \;=\; & 1234+5,\ 1235+4, 1245+3,\ 1345+2,\ 2345+1, \\ & 124+35,\ 125+34,\ 134+25,\ 135+24,\ 145+23, \\ & 234+15,\ 235+14,\ 245+13,\ 345+12. \end{aligned}$$

Indeed, we have obtained $S(5,2)=15$ partitions.

1.6.2 Distributions, surjections and multinomial numbers

We have seen that a partition is equivalent to the surjective placement of distinct balls in identical (unordered) boxes. If the boxes are ordered, the partition will be called **distribution**. Thus, defining a distribution means to place distinct objects in distinct boxes, such that no box remains empty, or, equivalent, to define a surjection from the set of balls to the set of boxes.

In Example 1.17, p. 18 we have

$$\text{box 1: } 1,2,9; \quad \text{box 2: } 3,5,7,10; \quad \text{box 3: } 4; \quad \text{box 4: } 6,8.$$

that is the surjection $\mathcal{D}: X \to \mathbb{N}_4$ is given by

x	1	2	3	4	5	6	7	8	9	10
$\mathcal{D}(x)$	1	1	2	3	2	4	2	4	1	2

.

In conclusion, the problem of counting distributions is equivalent to the problem of counting surjections. At this stage we can prove the following result.

Theorem 1.13 *Let $\mathcal{S}$ denote the set of surjections from an n-set X to a k-set Y. Then*

$$|\mathcal{S}| = k! \cdot S(n,k). \tag{1.14}$$

Proof. Each surjection $\mathcal{D}: X \to Y$ induces a partition of X into k parts. If such a partition is given, there are $k!$ surjections which induce it, since the k parts can be assigned to the k elements of Y in any bijective way. In conclusion, formula (1.14) holds true. ■

[2]James Stirling (1692-1770), Scottish mathematician.

Example 1.19 *Count the number of ways of dealing cards for a game of Bridge.*
Re-formulation. There are 52 objects (cards) and four boxes (the players), and we have to distribute 13 cards to each player. In this game it matters which player gets which cards. Thus, we do not require the number of partitions of a 52-set X into four 13-subsets, but the number of surjections from a 52-set X to the 4-set $Y = \{N, E, S, W\}$ with the property that each "box" of Y receives 13 members of X.

More generally, we may ask for the number of surjections from an n-set to a k-set of boxes $\{y_1, y_2, \ldots, y_k\}$ with the property that

$$\begin{aligned} &n_1 \text{ objects go into the first box } y_1, \\ &n_2 \text{ objects go into the second box } y_2, \\ &\cdots \\ &n_k \text{ objects go into the } k\text{-th box } y_k. \end{aligned}$$

This number is denoted by

$$\binom{n}{n_1, n_2, \ldots, n_k} \tag{1.15}$$

and is called **multinomial number**. In the following we give a formula for this number.

Theorem 1.14 *Given the positive integers $n, n_1, n_2, \ldots, n_k$ satisfying $n_1 + n_2 + \ldots + n_k = n$, we have*

$$\binom{n}{n_1, n_2, \ldots, n_k} = \frac{n!}{n_1! \cdot n_2! \cdot \ldots \cdot n_k!}. \tag{1.16}$$

Proof. First we "arrange" the objects in $X = \{x_1, x_2, \ldots, x_n\}$ as

$$x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)},$$

where π is a permutation of $\mathbb{N}_n$. Then we define a surjection from X onto Y by "introducing" the first n_1 objects of the list into the first box (y_1), the next n_2 objects into the second box (y_2), ..., the last n_k objects into the last box (y_k). We obtain the same surjection if we rearrange the first n_1 objects among themselves in any way. This can be done in $n_1!$ ways. Thus, from the $n!$ permutations π, there are $n_1! \cdot n_2! \cdot \ldots \cdot n_k!$ which induce a given surjection. In conclusion, the number of surjections with the specified property is indeed the one given in (1.16). ■

Remark 1.9 *The multinomial number (1.15) is also referred as the number of **permutations with repetitions**, of $n = \sum_{i=1}^{k} n_i$ symbols of k distinct types, where n_i are of type i, $i = 1, 2, \ldots, k$.*

Remark 1.10 *Theorem 1.14 remains true in the case when one or more of the numbers n_i is zero. In this case, the symbol in (1.15) is defined as the number of functions from an n-set to a set of k "boxes", with the property that n_i objects go into the box i, for $i = 1, \ldots, k$. In this case the functions are not necessarily surjections, since we allow some of the numbers n_i to be zero. With the convention $0! = 1$, formula (1.16) remains true.*

Example 1.20 *How many 11-letter words can be made from the letters of the word MISSISSIPPI?*

Solution. Each word corresponds to a surjection from the set $\{x_1, x_2, \ldots, x_{11}\}$ to the set of four "boxes" $\{I, M, P, S\}$, such that 4 objects go to the "box" I, one object to "box" M, 2 objects to "box" P and 4 objects to "box" S. The total number of words will be

$$\binom{11}{4,1,2,4} = \frac{11!}{4!1!2!4!} = 34650.$$

Let us note that in the particular case $k = 2$ the multinomial number equals the binomial number. For this reasons we expect to have a generalization of the binomial theorem. This generalization is known as the *multinomial theorem.*

Theorem 1.15 (The multinomial theorem) *Let n, k be positive integers. Then*

$$(x_1 + x_2 + \ldots + x_k)^n = \sum \binom{n}{n_1, n_2, \ldots, n_k} x_1^{n_1} x_2^{n_2} \ldots x_k^{n_k}, \tag{1.17}$$

where the sum is taken over all k-tuples of non-negative integers $(n_1, n_2, \ldots, n_k)$ such that $n_1 + n_2 + \ldots + n_k = n$.

Proof. When we calculate the left-hand side, the term $x_1^{n_1} x_2^{n_2} \cdots x_k^{n_k}$ arises by choosing n_i times the factor x_i, $i = 1, 2, \ldots, k$. In other words, the term $x_1^{n_1} x_2^{n_2} \cdots x_k^{n_k}$ corresponds to a function from the set of n factors (monomials) to the set $\{x_1, x_2, \cdots, x_k\}$, with the property that n_i of the factors go to x_i, for $i = 1, 2, \ldots, k$. There are

$$\binom{n}{n_1, n_2, \ldots, n_k}$$

such functions, and therefore this is the number of terms $x_1^{n_1} x_2^{n_2} \cdots x_k^{n_k}$ in the product. ■

1.6.3 Integer partitions

Definition 1.8 *Let $n \in \mathbb{N}$. Then $n = n_1 + n_2 + \ldots + n_k$, with $n_i \in \mathbb{N}$, $n \geq 1$, is called a k-**(integer) partition** of n. The number of k-partitions of n will be denoted by $P(n, k)$.*

Remark 1.11 *Defining a k-integer partition of n is equivalent to placing n identical balls in k identical boxes, such that no box remains empty (surjective placement).*

For the numbers $P(n, k)$ there is no nice recurrence formula similar to the one given in Theorem 1.12 for the Stirling numbers $S(n, k)$.

Example 1.21 *The 4-partitions of 8 are:*

$$5+1+1+1,\ 4+2+1+1,\ 3+3+1+1,\ 3+2+2+1,\ 2+2+2+2.$$

We can also consider **ordered** k-integer partitions, if, for instance, we consider that $3+3+1+1 \neq 3+1+3+1$. Thus, the construction of an ordered k-integer partition of n is equivalent to the placement of n identical balls in k distinct boxes, such that no box remains empty.

The problem of counting the ordered integer partition cannot be solved similarly to the case of ordered set partitions, where we multiplied the number $S(n,k)$ of unordered set partitions by the number $k!$ in which the parts can be permuted among themselves (see Theorem 1.13). Indeed, for $3+1+1$ there are not $6 = 3!$ distinct ordered partitions, but only three: $3+1+1$, $1+3+1$, $1+1+3$. Therefore, we have to use another approach.

Theorem 1.16 *The number of ordered k- integer partitions of n is $\binom{n-1}{k-1}$.*

Proof. One can see that each k-ordered partition of n is equivalent to the placement of n identical balls in k distinct boxes, such that no box remains empty. So, we start by placing a ball in each box. Thus, it remains to place arbitrarily the remained $n-k$ balls in k boxes. The associated bit string (see Section 1.4.5, p. 15) will contain $n-k$ ones and $k-1$ zeros. The number of bit strings of length $n-k+k-1 = n-1$, with exactly $k-1$ zeros, is indeed $\binom{n-1}{k-1}$. ■

Example 1.22 *The ordered 3-partitions of 6 are:*

$$4+1+1,\ 1+4+1,\ 1+1+4,\ 3+2+1,\ 3+1+2,$$
$$2+3+1,\ 2+1+3,\ 1+3+2,\ 1+2+3,\ 2+2+2.$$

There are indeed $\binom{5}{2} = 10$ ordered partitions.

1.7 Solving counting problems using recurrence relations

Example 1.23 (Towers of Hanoi puzzle) *We have three pegs numbered 1,2,3 and on one peg we have a stack of n discs, each smaller in diameter than the one below. An allowable move consists of removing a disk from one peg so that it is not above another disk of smaller size. How many allowable moves are needed to move the disks from one peg to another?*

Solution. To solve the problem of moving all disks from peg 1 to peg 2, we do the following:

1. (recursively) Solve the problem of moving $n-1$ disks from peg 1 to peg 3;
2. move disk n to peg 2;
3. (recursively) Solve the problem of moving $n-1$ disks on peg 3 to peg 2.

Let a_n denote the number of moves needed to move n disks from peg i to peg j. We have the recursion

$$a_n = 2a_{n-1} + 1, \quad a_1 = 1.$$

Further, $a_2 = 3$, $a_3 = 7$, so we can easily guess that $a_n = 2^n - 1$, fact which can be proved immediately by induction.

What happens if we get to a recurrence relation where it is impossible to guess the general term? We will need a more sophisticated theory for these situations.

1.8 Recursions and generating functions

1.8.1 The linear recursion

Let $k \in \mathbb{N}$. A simple kind of recursion is the following

$$u_{n+k} = \alpha_1 u_{n+k-1} + \alpha_2 u_{n+k-2} + \ldots + \alpha_k u_n, \ n \geq 0. \tag{1.18}$$

with α_i, $i = 1, 2, \ldots, k$, given real numbers such that $\alpha_k \neq 0$. This is called a **linear recursion** of degree k and we can find an explicit formula for the general term u_n whenever $u_0, u_1, \ldots, u_{k-1}$ are given. For this we need first to prove the following Lemma.

Lemma 1.17 *The set of sequences satisfying the recurrence (1.18) is a vector space of dimension k.*

Proof. If $(a_n)_n$ and $(b_n)_n$ satisfy (1.18), then a simple calculation shows that $\alpha \cdot (a_n)_n$ and $(a_n + b_n)_n$ satisfy (1.18), so we have a linear space. In order to find its dimension, we need to find a basis consisting of k sequences.

The idea is to associate the *characteristic equation* $R(x) = 0$, where

$$R(x) = x^k - \alpha_1 x^{k-1} - \ldots - \alpha_{k-1} x - \alpha_k. \tag{1.19}$$

The following situations are possible:

1. If r_1 is a root of R, then the sequence $(u_n)_n$, where $u_n = r_1^n$, satisfies (1.18). Indeed, we have $r_1^k = \alpha_1 r_1^{k-1} + \ldots + \alpha_{k-1} r_1 + \alpha_k$ and then, after multiplying this equality by r_1^n, we get the conclusion.

2. If r_1 is a nonzero double root of R, then the sequence $(b_n)_n$, where $b_n = n\, r_1^n$, satisfies (1.18). Indeed, from $R(r_1) = R'(r_1) = 0$ we obtain

$$\begin{aligned} k\, r_1^{n+k} &= \alpha_1 (k-1) r_1^{n+k-1} + \ldots + \alpha_{k-1} r_1^{n+1}, \\ r_1^{n+k} &= \alpha_1 r_1^{n+k-1} + \ldots + \alpha_{k-1} r_1^{n+1} + \alpha_k r_1^n. \end{aligned}$$

If we multiply the second identity by n and add them, we get the conclusion.

3. If r_1 is a root with multiplicity p for R, then the sequences $(u_n)_n$ with

$$u_n \in \{r_1^n, n\, r_1^n, n^2 r_1^n, \ldots, n^{p-1} r_1^n\}$$

are solutions of (1.18). To prove this we follow the similar arguments as before.

4. If $r_{1,2} = \alpha \pm \beta\, i = e^{a \pm b\, i}$ are complex roots of R, then we can write

$$r_1 = e^a e^{bi} = e^a (\cos b + i \sin b),$$

and therefore $r_{1,2}^n = e^{an}(\cos b\, n \pm i \sin b\, n)$. In this case the sequences $(c_n)_n$ and $(d_n)_n$, given by

$$\begin{aligned} c_n &= \frac{1}{2}(r_1^n + r_2^n) = e^{a\, n} \cos b\, n, \\ d_n &= \frac{1}{2}(r_1^n - r_2^n) = e^{a\, n} \sin b\, n, \end{aligned}$$

are *real* solutions of (1.18).

5. If $r_{1,2} = \alpha \pm \beta\, i = e^{a \pm b\, i}$ are complex roots of R with multiplicity p, then similarly we can prove that

$$c_n, d_n, n\, c_n, n\, d_n, n^2 c_n, n^2 d_n, \ldots, n^{p-1} c_n, n^{p-1} d_n$$

satisfy the recurrence (1.18).

In conclusion, to the k roots of R one can associate linearly independent solutions. The general solution $(Y_n)_n$ of the recurrence (1.18) will be a linear combination of these solutions. The coefficients of the linear combination will be uniquely determined from the "initial" conditions $Y_0 = u_0, \ldots, Y_{k-1} = u_{k-1}$. For instance, in the case when R has real distinct roots, then the general solution is $(Y_n)_n$, with $Y_n = \beta_1 r_1^n + \beta_2 r_2^n + \ldots + \beta_k r_k^n$. The real constants β_i are uniquely determined from the conditions $Y_i = u_i,\ i = 1, \ldots, k$, which can be written as

$$\begin{aligned}
&\beta_1 + \beta_2 + \ldots + \beta_k = u_0,\\
&\beta_1 r_1 + \beta_2 r_2 + \ldots + \beta_k r_k = u_1,\\
&\vdots\\
&\beta_1 r_1^{n-1} + \beta_2 r_2^{n-1} + \ldots + \beta_k r_k^{n-1} = u_{k-1}.
\end{aligned}$$

The unknowns $\beta_1, \ldots, \beta_k$ are uniquely determined since the determinant of the system is nonzero (as a Vandermonde determinant of the distinct roots r_i). One can prove that this determinant is nonzero in the rest of the cases. ■

1.8.2 Generating functions

As we have already seen in Example 1.23, the solution of a counting problem (and not only) can be expressed as a sequence $(u_n)_n$. In such cases, a common way to solve this type of problems is based on the representation of $(u_n)_n$ as a power series

$$g_u(x) = \sum_{n=0}^{\infty} u_n x^n. \tag{1.20}$$

In this situation, g_u is referred to as the **generating function**[3] for the sequence $(u_n)_n$. (It is of course necessary that the series converges somewhere if g_u is defined as a function of x. If we regard g_u as an element of a ring of polynomials, such a convergence is not necessary.) In some cases, the series (1.20) can be reduced to a finite sum, in which case g_u is a polynomial.

Example 1.24 *Let $m \in \mathbb{N}$ be fixed and let u_n denote the number of n-subsets of an m-set. It is known (see Section 1.4.4, p. 14) that*

$$u_n = \binom{m}{n},$$

whence the generating function will be

$$g_u(x) = \sum_{n=0}^{m} \binom{m}{n} x^n = (1+x)^m.$$

[3]They were introduced by De Moivre and Euler in the early eighteenth century.

In the case when a recurrence formula is given for $(u_n)_n$, the generating function can be useful in determining a general expression for u_n. The method uses three stages:

(i) from the recurrence relation we find an equation for $g_u(x)$,

(ii) we solve the equation for $g_u(x)$,

(iii) we find a formula for the coefficients u_n of $g_u(x)$ (via partial fraction decomposition or binomial theorem)in the following way: If we can determine g_u and its power series, we will identify the coefficients of the power series in order to find u_n.

Remark 1.12 *For linear and homogeneous recursion studied the previous section, one can also use the generating functions, but this method requires more calculation than using the characteristic polynomial.*

1.8.3 Non-homogeneous linear recursions

In some cases the method of generating functions can be used to solve non-homogeneous recursion. Such a recursion is

$$u_{n+2} = au_{n+1} + bu_n + f(n), \quad u_0, u_1 \text{ given.} \tag{1.21}$$

The applicability of the method depends on the particular form of f. Let

$$g_u(x) = \sum_{n=0}^{\infty} u_n x^n.$$

If we multiply this identity by $(a + bx)$ we obtain

$$\begin{aligned}(a+bx)g_u(x) &= au_0 + (au_1 + bu_0)x + (au_2 + bu_1)x^2 + \dots \\ &\qquad +(au_n + bu_{n-1})x^n + \dots\end{aligned}$$

From (1.21) we deduce that

$$\begin{aligned} au_1 + bu_0 &= u_2 - f(0), \\ au_2 + bu_1 &= u_3 - f(1), \\ &\vdots \\ au_n + bu_{n-1} &= u_{n+1} - f(n-1). \end{aligned}$$

Inserting this expression we get, after grouping the terms and multiplying the equality by x,

$$x(a+bx)g_u(x) = au_0 x + \sum_{n=1}^{\infty} u_{n+1}x^{n+1} - \sum_{n=1}^{\infty} f(n-1)x^{n+1}. \tag{1.22}$$

At this stage, we need to determine the sum of the power series

$$\sum_{n=1}^{\infty} f(n-1)x^{n+1} = h(x).$$

If we cannot find its sum h, the method fails.

Let us suppose that we have found h. We focus on (1.22) and we replace the first sum in the right-hand side with $g_u(x) - u_0 - u_1 x$. Thus, (1.22) becomes

$$g_u(x) = \frac{-u_0 + (au_0 - u_1)x - h(x)}{bx^2 + ax - 1}. \tag{1.23}$$

By expanding this function in a power series and identifying the coefficient of x^n, we obtain a general expression for u_n.

Example 1.25 *Find the general term u_n of the recursion*

$$u_{n+2} = -u_{n+1} + 2u_n + n, \text{ with } u_0 = -2, u_1 = 1.$$

Solution. From the calculations above, made in the general case, we find

$$h(x) = \sum_{n=1}^{\infty}(n-1)x^{n+1} = x^3 \sum_{n=2}^{\infty}(n-1)x^{n-2} = \frac{x^3}{(1-x)^2},$$

Then, from (1.23),

$$g_u(x) = \frac{-2+3x}{(1-x)^3(2x+1)} = \frac{A}{(1-x)^3} + \frac{B}{(1-x)^2} + \frac{C}{1-x} + \frac{D}{2x+1},$$

with $A = \frac{1}{3}, B = -\frac{7}{9}, C = -\frac{14}{27}, D = -\frac{28}{27}$. Further, we write

$$\begin{aligned}
\frac{1}{1-x} &= \sum_{n=0}^{\infty} x^n \quad | \cdot C \\
\frac{1}{(1-x)^2} &= \sum_{n=0}^{\infty}(n+1)x^n \quad | \cdot B \\
\frac{1}{(1-x)^3} &= \sum_{n=0}^{\infty}(n+1)(n+2)x^n \quad | \cdot A \\
\frac{1}{2x+1} &= \sum_{n=0}^{\infty}(-1)^n(2x)^n \quad | \cdot D
\end{aligned}$$

By identifying the coefficient of x^n we obtain

$$\begin{aligned}
u_n &= \frac{A}{2}(n+1)(n+2) + B(n+1) + C + D(-1)^n 2^n \\
&= \frac{1}{54}\left(9n^2 - 15n - 52 - 7(-1)^n 2^{n+3}\right).
\end{aligned}$$

Remark 1.13 *In a more general case, when the non-homogeneous linear recursion is*

$$u_{n+k} = a_1 u_{n+k-1} + a_2 u_{n+k-2} + \ldots + a_k u_n + g(n),$$

with $u_0, u_1, \ldots, u_{k-1}$ given, we start by calculating

$$(1 + a_1 x + a_2 x^2 + \ldots + a_k x^{k-1}) g_u(x)$$

and hope again that the terms containing $g(n)$ can be conveniently manipulated.

1.8.4 Catalan recursion

If we want to count the number u_n of binary trees with n vertices (see Section 3.4.1), we arrive at the following recursion, known as **Catalan**[4] **recursion**.

$$u_{n+1} = u_0u_n + u_1u_{n-1} + \ldots + u_ku_{n-k} + \ldots + u_nu_0.$$

with $u_0 = u_1 = 1$. The quantity u_n is called the n **th Catalan number**. Let us note that the number of terms in this recursion is not constant, as in the previous cases.

Consider the generating function

$$g_u(x) = \sum_{n=0}^{\infty} u_n x^n.$$

One has

$$\begin{aligned} g_u^2(x) &= (u_0 + u_1x + u_2x^2 + \ldots)(u_0 + u_1x + u_2x^2 + \ldots) \\ &= u_0 + 2u_0u_1x + (u_0u_2 + u_1^2 + u_2u_0)x^2 + \ldots \\ &+ (u_0u_n + u_1u_{n-1} + \ldots + u_nu_0)x^n + \ldots \\ &= u_0 + u_2x + u_3x^2 + \ldots + u_{n+1}x^n + \ldots, \end{aligned}$$

whence

$$xg_u^2(x) = x + \sum_{n=2}^{\infty} u_n x^n,$$

and further

$$xg_u^2(x) - g_u(x) + 1 = 0.$$

Solving this equation we obtain

$$g_{u1}(x) = \frac{1 + \sqrt{1-4x}}{2x}, \quad g_{u2}(x) = \frac{1 - \sqrt{1-4x}}{2x}.$$

Because $g_u(0) = u_0 = 1$, the only solution is

$$g_u(x) = \frac{1 - \sqrt{1-4x}}{2x}.$$

We expand now the function g_u using the formula

$$(1-y)^\alpha = 1 + \frac{\alpha}{1!}y + \frac{\alpha(\alpha-1)}{2!}y^2 + \ldots + \frac{\alpha(\alpha-1)\ldots(\alpha-n+1)}{n!}y^n + \ldots$$

for $\alpha = 1/2$ and $y = -4x$.

The coefficient of x^n will be

$$u_n = \frac{1 \cdot 3 \cdot \ldots \cdot (2n-1)}{2^{n+2}(n+1)!}4^{n+1} = \frac{1}{n+1}\binom{2n}{n}.$$

[4]Eugène Charles Catalan (1814-1894), Belgian mathematician.

1.8.5 Some properties of generating functions

Theorem 1.18 (Convolution) *If u_n can be written as a* convolution *of the sequences $(a_n)_n$ and $(b_n)_n$, so*

$$u_n = \sum_{k=0}^{n} a_k b_{n-k},$$

then

$$g_u(x) = g_a(x) g_b(x).$$

Proof. If we evaluate $g_a g_b$,

$$g_a(x) g_b(x) = (a_0 + a_1 x + \ldots + a_n x^n + \ldots)(b_0 + b_1 x + \ldots + b_n x^n + \ldots),$$

the coefficient of x^n is indeed

$$a_0 b_n + a_1 b_{n-1} + \ldots + a_n b_0 = \sum_{k=0}^{n} a_k b_{n-k}.$$

■

Theorem 1.19 *If*

$$b_n = \sum_{k=0}^{n} a_k,$$

then

$$g_b(x) = \frac{g_a(x)}{1-x}.$$

Proof. Using the geometric series we get

$$\begin{aligned} \frac{g_a(x)}{1-x} &= (a_0 + a_1 x + \ldots + a_n x^n + \ldots)(1 + x + x^2 + \ldots + x^n + \ldots) \\ &= a_0 + (a_0 + a_1)x + (a_0 + a_1 + a_2)x^2 + \ldots \\ &+ (a_0 + a_1 + \ldots + a_n)x^n + \ldots \end{aligned}$$

so the coefficient of x^n will be indeed $b_n = \sum_{k=0}^{n} a_k$. ■

Theorem 1.20 (Tails) *If*

$$b_n = \sum_{k=1}^{\infty} a_{n+k}, \quad n \geq 0,$$

then

$$g_b(x) = \frac{g_a(1) - g_a(x)}{1-x}.$$

Proof. Let $h_a(x) = g_a(1) - g_a(x)$. Its power series can be written as $\sum_{n=0}^{\infty} c_n x^n$, with

$$\begin{aligned} c_0 &= g(1) - a_0 = a_1 + a_2 + \ldots, \\ c_n &= -a_n, \text{ for } n \geq 1. \end{aligned}$$

According to Theorem 1.19 we have

$$\frac{h_a(x)}{1-x} = \sum_{n=0}^{\infty} b_n x^n, \text{ with } b_n = \sum_{k=0}^{n} c_k = c_0 + \sum_{k=1}^{n}(-a_k) = \sum_{k=1}^{\infty} a_{n+k}.$$

■

1.9 Solved problems

Problem 1.1 ***(Identical balls into distinct boxes with multiple balls per box allowed)*** *The number of ways k identical balls can be placed into n distinct boxes, with any number of balls allowed in each box is $\binom{n+k-1}{k}$.*

Solution. The required number can be found by a direct application of Theorem 1.10, p. 15.

Problem 1.2 *Which is the number of nonnegative integer solutions to the equation*

$$x_1 + x_2 + \ldots + x_6 = 10?$$

Solution. See Example 1.14, p. 16 or Problem 1.1. The required number will be again $\binom{15}{5}$.

Problem 1.3 ***(Identical balls into distinct boxes with no box allowed to be empty)*** *The number of ways k identical balls can be placed into n distinct boxes, with any number of balls allowed in each bin and no box allowed to remain empty, is $\binom{k-1}{n-1}$.*

Solution. We use the idea in the proof of Theorem 1.10, p. 15. We place one ball into each box, since no box should be empty. Thus, it rests to place the remaining $k - n$ balls into n boxes, in the same way we did in the previous problem. This number is therefore

$$\binom{n-1+k-n}{n-1} = \binom{k-1}{n-1}.$$

Problem 1.4 *Which is the number of positive integer solutions to the equation*

$$x_1 + x_2 + \ldots + x_6 = 10?$$

Solution. According to Problem 1.3, the required number is $\binom{9}{5}$.

Problem 1.5 *What is the number of nonnegative integer solutions to the equation*

$$x_1 + x_2 + x_3 + x_4 + x_5 = 36,$$

where $x_1 \geq 4,\ x_4 \geq 7$.

Solution. The problem can be reformulated as follows: What is the number of ways 36 identical balls can be placed into 5 distinct boxes, with at least 4 balls in the first box and at least 7 balls in the 4th box?

We start by placing 4 balls in the first box and 7 balls in the fourth box. It rests 25 balls to be placed arbitrarily into the 5 boxes, and this can be done in $\binom{29}{4}$ ways (see Problem 1.1).

Problem 1.6 (The derangement problem) *An inefficient secretary has n letters and n addressed envelopes. In how many ways can the secretary achieve the feat of putting every letter into the wrong envelope?*

Solution. We have to count the number of derangements of Π_n (see Definition 1.5, p. 14). Let $\pi : \mathbb{N}_n \to \mathbb{N}_n$ denote the permutation defined as

$$\pi(i) = j, \quad \text{if letter } i \text{ goes into envelope } j.$$

For $i \in \mathbb{N}_n$ we denote by A_i the set of permutations which fix the position i. According to Corollary 1.3, p. 9, the number of derangements will be

$$d_n = n! - \alpha_1 + \alpha_2 - \alpha_3 \cdots + (-1)^n \alpha_n,$$

where α_k is the number of permutations which fix k positions. We say that a permutation π *fixes* the position j if $\pi(j) = j$.

We can chose k positions in $\binom{n}{k}$ ways. For each fixation of k positions, there are $(n-k)!$ permutations of the remaining positions, therefore

$$\alpha_k = \binom{n}{k}(n-k)! = \frac{n!}{k!}.$$

Thus

$$d_n = n!\left(1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \ldots + (-1)^n \frac{1}{n!}\right).$$

Problem 1.7 (Poker hands) *In the poker game there are 52 cards and a player receives 5 cards. Find the number of each of the following hands:*
a. royal flush (ace, king, queen, jack, 10 in the same suit);
b. straight flush (5 cards of 5 consecutive ranks, all in 1 suit, but not a royal flush);
c. four of a kind (four cards in 1 rank and a fifth card);
d. full house (3 cards of 1 rank, 2 of another rank);
e. flush (5 cards in 1 suit, but neither royal nor straight flush);
f. straight (5 cards in 5 consecutive ranks, but not all of the same suit);
g. three of a kind (3 cards of 1 rank and 2 cards of 2 different ranks);
h. two pairs (2 cards in each of 2 different ranks and a fifth card of a third rank);
i. one pair (2 cards in 1 rank, plus 3 cards from 3 other ranks).

Solution.

a. 4 (there are 4 choices for a suit ($\clubsuit, \diamondsuit, \heartsuit, \spadesuit$) and one royal flush in each suit).

b. $4 \cdot 9 = 36$ (4 choices for a suit, in each suit 9 ways to get 5 cards in a row: $(A,2,3,4,5)$, $(2,3,4,5,6)$, $(3,4,5,6,7), \ldots, (9,10,J,Q,K)$).

c. $13 \cdot 48 = 624$ (13 choices for a rank, one way to select 4 cards in that rank, 48 ways to select the fifth card).

d. $13 \cdot \binom{4}{3} \cdot 12 \cdot \binom{4}{2} = 3744$ (13 ways to select a rank for the 3-of-a-kind, $\binom{4}{3}$ ways to choose 3 of this rank, 12 ways to select a rank for the pair and $\binom{4}{2}$ ways to get a pair of the other rank).

e. $4\binom{13}{5} - 40 = 5108$ (4 ways to select a suit, $\binom{13}{5}$ ways to choose 5 cards in that suit; subtract royal+straight flush).

f. $10 \cdot 4^5 - 40 = 10200$ (10 ways to choose 5 ranks in a row and 4 ways to choose a card from each rank; subtract royal+ straight flush).

g. $13\binom{4}{3}\binom{12}{2} \cdot 4^2 = 54912$ (13 ways to select one rank, $\binom{4}{3}$ ways to choose 3 cards of that rank, $\binom{12}{2}$ ways to pick other ranks and 4^2 ways to pick a card of each of those 2 ranks).

h. $\binom{13}{2}\binom{4}{2}\binom{4}{2} \cdot 44 = 123\,552$ ($\binom{13}{2}$ ways to select 2 ranks, $\binom{4}{2}$ ways to choose 2 cards in each of these ranks, 44 ways to choose a non matching fifth card).

i. $13\binom{4}{2}\binom{12}{3} \cdot 4^3 = 1\,098\,240$ (13 ways to select a rank, $\binom{4}{2}$ ways to choose 2 cards in that rank, $\binom{12}{3}$ ways to pick one card from each of those ranks).

Let us mention that there are $\binom{52}{5} = 2\,598\,960$ ways to chose 5 cards. The number of hands which are none of the above mentioned hands will be $1\,302\,540$, which is slightly greater than $\binom{52}{5}/2 = 1\,299\,480$.

Chapter 2

Discrete Probability Theory

2.1 Random events

Many processes in nature, economy, social sciences, medicine, etc. are subject to chance, that is, the outcomes of a process cannot be predicted. The exchange rate at a certain day in the future, the blood pressure of a person at a given moment, are only two examples. Such random events cannot be predicted with certainty, but the relative frequency with which they occur in a long series of trials is often stable. Events possessing this property are called *random* or *stochastic* events. Probability theory supplies mathematical models for such random phenomena.

A **random experiment** (or a **trial**) is a process in which various outcomes are possible, so that one cannot say in advance what outcome will be.

We say that two outcomes are **incompatible** if they cannot occur simultaneously. The possible pairwise incompatible outcomes of an experiment are called its **elementary events**. The set of elementary events is denoted by E and is called the **sample space**. For example, in the experiment of *rolling a die once*, the elementary events are $e_1, e_2, \ldots, e_6$, where e_i is the event "face number i appears". So, the sample space will be $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$. If we consider the experiment of *testing the lifespan of an electric bulb*, one has an infinity of elementary events e_x, $x \in [0, \infty)$, e_x being the event "*the lifespan is x minutes*".

Since this work is dedicated to *discrete* mathematics, we will consider in the sequel only experiments with a *finite* or *countable* number of outcomes.

Apart of the elementary events, one is often interested in events of a complex nature, for instance, when a die is rolled, "*an even number appears*". Let an experiment be given and let E be the set of its elementary events. Every subset of E is called an **event**. An event A occurs if and only if one of the elementary events forming A occurs. For example, in the experiment of rolling a die, the set $A = \{e_2, e_4, e_6\}$ can be interpreted as the event "*an even number appears*". Indeed, this event occurs in the rolling of a die if and only if one of the elementary events contained in A occurs.

Since $E \subseteq E$ and $\emptyset \subseteq E$, E and $\emptyset$ can also be regarded as events, according to the definition of an event. Since E consists of all elementary events and in any experiment exactly one elementary event takes place, E *always take place.* This is why E is called the **sure event**. On the other hand, $\emptyset$ contains no elementary event, therefore never takes place. It will be called the **impossible event**.

Let $A_1, A_2, \ldots, A_n \subseteq E$ be events. Then:

1. $A_1 \cup \ldots \cup A_n$ is an event called the **sum** of the events A_i, which takes place exactly when at least one of the events $A_1, \ldots, A_n$ occurs.

2. $A_1 \cap \ldots \cap A_n$ is an event called the **product** of the events A_i, which occurs if and only if $A_1, \ldots, A_n$ occur simultaneously.

The events $A_1, A_2 \subseteq E$ are called **incompatible** if $A_1 \cap A_2 = \emptyset$ (A_1 and A_2 cannot occur simultaneously).

Since $\overline{A} = E \setminus A \subseteq E$, $\overline{A}$ is also an event. It occurs precisely when A does not occur. The event $\overline{A}$ will be called the **complementary** event to A.

Finally let us mention that all the properties of the operations $\cup, \cap$ and complementarity available for sets will be also true for events.

2.2 The axioms of probability theory

Definition 2.1 (Kolmogorov) [1] *Let E be a sample space associated to a random experiment and let $\mathcal{P}(E) = \{A,\ A \subseteq E\}$. A **probability measure** on E is a function $P : \mathcal{P}(E) \to [0,1]$ satisfying the following conditions (axioms):*

1. $P(E) = 1$,

2. $P\left(\bigcup_{k=1}^{\infty} A_k\right) = \sum_{k=1}^{\infty} P(A_k)$ *if A_k are pairwise disjoint events.*

Consequences

Let E be a sample space associated to a random experiment and let $A, B \subseteq E$ be events. The following properties hold:

1. $P(\emptyset) = 0$.
2. $P\left(\bigcup_{k=1}^{n} A_i\right) = \sum_{k=1}^{n} P(A_k)$ if $A_k,\ k = 1, \ldots, n$, are pairwise incompatible events.
3. $P(\overline{A}) = 1 - P(A)$,
4. $P(B \setminus A) = P(B) - P(A \cap B)$.
   If $A \subseteq B$, then $P(B \setminus A) = P(B) - P(A)$.
5. $P(A \cup B) = P(A) + P(B) - P(A \cap B)$.
6. If $A \subseteq B$, then $P(A) \leq P(B)$.
7. If $A_i,\ i = 1, \ldots, n$, are arbitrary events, then

$$P\left(\bigcup_{i=1}^{n} A_i\right) = \sum_{i=1}^{n} P(A_i) - \sum_{1 \leq i < j \leq n} P(A_i \cap A_j) + \sum_{1 \leq i < j < k \leq n} P(A_i \cap A_j \cap A_k) - \cdots - (-1)^n P(A_1 \cap \ldots \cap A_n).$$

8. $P(A \cup B) \leq P(A) + P(B)$.

[1] Andrey Nikolaevich Kolmogorov (1903–1987), Russian mathematician.

9. $P\left(\bigcup_{k=1}^{\infty} A_i\right) \leq \sum_{k=1}^{\infty} P(A_k)$.

Proof.

1. Taking in condition **2.** $A_1 = E,\ A_k = \emptyset,$ for $k > 1$, we get
$$P(E) = P(E) + \sum_{i=2}^{\infty} P(\emptyset), \quad \text{so } P(\emptyset) = 0.$$
2. We take in condition **2.** $A_k = \emptyset$, for $k > n$.
3. We have $1 = P(E) = P(A \cup \overline{A}) = P(A) + P(\overline{A})$.
4. We write $B = (A \cap B) \cup (B \setminus A)$. Since $A \cap B$ and $B \setminus A$ are disjoint, we have $P(B) = P(A \cap B) + P(B \setminus A)$, whence the conclusion.
5. We write $A \cup B = A \cup (B \setminus A)$, so $P(A \cup B) = P(A) + P(B \setminus A)$. Replacing $P(B \setminus A)$ from the previous property, we get the conclusion.
6. Using property 4 for the case $A \subseteq B$, we get $0 \leq P(B \setminus A) = P(B) - P(A)$, whence the required inequality.
7. This property can be proved by induction on n. ■

Remark 2.1 *Definition 2.1 was given by Kolmogorov and **1.,2.** will be referred as* Kolmogorov's axioms*. There are other definitions for the probability, using some of the properties* 1-9*, in which case the Kolmogorov axioms appear as consequences.*

Proposition 2.1 *If the nature of an experiment is such that it has only finitely many elementary events, and if they are* equally possible*, then it follows from the Kolmogorov axioms that the probability of an event A is*

$$P(A) = \frac{\textit{number of elementary events favorable to } A}{\textit{number of all elementary events}}. \tag{2.1}$$

Formula (2.1) was used by Laplace[2] to define probability.

Proof. We will prove that a probability given by the Kolmogorov definition satisfies (2.1).

For the beginning we consider the sample space $E = \{e_1, \ldots, e_n\}$ with e_i equally possible elementary events. Then

$$1 = P(E) = P(e_1 \cup \ldots \cup e_n) = \sum_{i=1}^{n} P(e_i) = nP(e_j), \tag{2.2}$$

whence

$$P(e_j) = \frac{1}{n} \text{ for all } j \in \mathbb{N}_n.$$

In the third equality in (2.2) we used the fact that the elementary events are incompatible. Further, let $A \subseteq E$, $A = e_1 \cup e_2 \cup \ldots \cup e_k$. The probability of A will be

$$P(A) = P(e_1 \cup e_2 \cup \ldots \cup e_k) = \sum_{j=1}^{k} P(e_j) = k \cdot \frac{1}{n} = \frac{k}{n}.$$

[2]Pierre-Simon, Marquis de Laplace (1749-1827), French mathematician and astronomer.

Indeed, k is the number of elementary events favorable to A and n is the number of total events. In conclusion, (2.1) is proved. ■

Example 2.1 *Two dice are rolled at the same time and there is a prize when the total of faces is* ≥ 10. *What is the probability of winning?*

Solution. The sample space will be

$$E = \{e_{11}, e_{12}, \ldots, e_{16},\ e_{21}, e_{22}, \ldots, e_{26},\ \ldots,\ e_{61}, e_{62}, \ldots, e_{66}\},$$

while the event A of having the total of faces ≥ 10 can be written as

$$A = \{e_{46}, e_{55}, e_{56}, e_{64}, e_{65}, e_{66}\}.$$

Hence,

$$P(A) = \frac{|A|}{|E|} = \frac{6}{36} = \frac{1}{6}.$$

2.3 Conditional probabilities and independent random events

The probability of a random event A is altered, in general, when it is already known that another random event B with $P(B) \neq 0$ has taken place. The probability of A, under the condition that B (with $P(B) \neq 0$) has already occurred, is denoted by $P(A|B)$ and is called the **conditional probability of** A **under the condition** B or the **conditional probability of** A **given** B.

Example 2.2 *Suppose we have two urns,* U_1 *containing 5 white and 5 black balls and* U_2 *containing one white and 9 black balls. We perform the experiment of choosing blindly one of the urns and taking out of it, blindly, a ball. Then we consider the following events:*

$$\begin{aligned} B &: \text{“the drawn ball is white”},\\ A_i &: \text{“the ball was taken from the } i\text{-th urn”},\ i = 1, 2. \end{aligned}$$

Then

$$P(B|A_1) = \frac{5}{10} = \frac{1}{2}, \qquad P(B|A_2) = \frac{1}{10}.$$

In general, conditional probabilities are defined by the relations

$$\begin{aligned} P(A|B) &= \frac{P(A \cap B)}{P(B)}, \quad \text{if } P(B) \neq 0,\\ P(B|A) &= \frac{P(A \cap B)}{P(A)}, \quad \text{if } P(A) \neq 0, \end{aligned}$$

which implies

$$P(A \cap B) = P(B) \cdot P(A|B) = P(A) \cdot P(B|A). \tag{2.3}$$

Definition 2.2 *Two random events* A, B *are called* ***independent*** *(of each other) when the occurrence of one does not influence in any way the probability of the other occurring, that is, when* $P(A|B) = P(A)$.

Thus, for the independent events A, B, we have

$$P(A \cap B) = P(A) \cdot P(B). \tag{2.4}$$

Definition 2.3 *The random events $A_1, \ldots, A_n$ are called **totally independent** if for every $m \leq n$ and any m-tuple $(i_1, i_2, \ldots, i_m)$, $1 \leq i_1 \leq i_2 \leq \ldots \leq i_m \leq n$ we have*

$$P(A_{i_1} \cap A_{i_2} \cap \ldots \cap A_{i_m}) = P(A_{i_1})P(A_{i_2}) \ldots P(A_{i_m}).$$

Definition 2.4 *The random events $A_1, \ldots, A_n$ are called **pairwise independent** if A_i and A_j are independent for any $i \neq j$.*

It is immediate that total independence implies pairwise independence. However, the reverse implication is not true, as shown by the following example.

Example 2.3 *Let $E = \{e_1, e_2, e_3, e_4\}$ be a sample space, where the elementary events e_i have all the probability $P(e_i) = 1/4$. Consider the events $A = e_1 \cup e_2$, $B = e_1 \cup e_3$, $C = e_1 \cup e_4$. We will show that A, B, C are pairwise independent, but not totally independent events.*

On one hand, $P(A) = P(B) = P(C) = 1/2$,

$$P(A \cap B) = P(e_1 \cup (e_2 \cap e_3)) = P(e_1 \cup \emptyset) = P(e_1) = \frac{1}{4},$$

and analogously $P(A \cap C) = P(B \cap C) = 1/4$. In conclusion

$$\begin{aligned} P(A \cap B) &= P(A) \cdot P(B), \\ P(A \cap C) &= P(A) \cdot P(C), \\ P(B \cap C) &= P(B) \cdot P(C), \end{aligned}$$

whence the events A, B, C are pairwise independent.

On the other hand,

$$P(A \cap B \cap C) = P(e_1) = \frac{1}{4}, \quad P(A) \cdot P(B) \cdot P(C) = \frac{1}{8},$$

so A, B, C are not totally independent.

Theorem 2.2 *Let $A_1, A_2, \ldots, A_n$ be random events. Then*

$$\begin{aligned} &P(A_1 \cap A_2 \cap \ldots \cap A_n) = \\ &\quad P(A_1) \cdot P(A_2|A_1) \cdot P(A_3|A_1 \cap A_2) \cdot \ldots \cdot P(A_n|A_1 \cap \ldots \cap A_{n-1}). \end{aligned}$$

Proof. The proof is immediate using the induction on n and formula (2.3). ■

An immediate consequence is

Corollary 2.3 *If the events $A_1, A_2, \ldots, A_n$ are totally independent, then*

$$P(A_1 \cap A_2 \cap \cdots \cap A_n) = P(A_1) \cdot P(A_2) \cdot \ldots \cdot P(A_n).$$

2.4 The law of total probability. Bayes' formula[3]

Definition 2.5 *Let E be a sample space. A set $\mathcal{S} = \{A_1, A_2, \ldots, A_n\}$ of random events is called* ***complete system of events*** *(CSE) for E if*

1. $E = A_1 \cup A_2 \cup \ldots \cup A_n$,

2. $A_i \cap A_j = \emptyset$, *whenever* $i \neq j$.

A (CSE) can analogously be defined for countable many events.

Theorem 2.4 (Law of total probability) *Let E be a sample space, $\mathcal{S} = \{A_1, A_2, \ldots, A_n\}$ a (CSE) and B a random event. Then*

$$P(B) = \sum_{i=1}^{n} P(A_i) \cdot P(B|A_i). \tag{2.5}$$

Formula (2.5) is referred to as the **formula for total probability** (FTP).
Proof. We have

$$\begin{aligned} B = E \cap B &= (A_1 \cup A_2 \cup \ldots \cup A_n) \cap B \\ &= (A_1 \cap B) \cup (A_2 \cap B) \cup \ldots \cup (A_n \cap B). \end{aligned}$$

Since the events $A_i \cap B$ in the last sum are pairwise incompatible, we obtain

$$P(B) = \sum_{i=1}^{n} P(A_i \cap B).$$

Further, the use of formula (2.3) for the probabilities $P(A_i \cap B)$ leads to conclusion (2.5). ■

Remark 2.2 *Let us mention that the event B takes place together with one (and only one) of the events of a CSE.*

Example 2.4 *We have three urns of type 1 (containing two white balls and six black balls) and one urn of type 2 (containing one white ball and eight black balls). One urn is chosen at random and then a ball is drawn from it. What is the probability of the event B: "the drawn ball is white"?*

Solution. Consider the events A_i: "an urn of type i is chosen", $i = 1, 2$. Then $\mathcal{S} = \{A_1, A_2\}$ will be a CSE, since $E = A_1 \cup A_2$ and $A_1 \cap A_2 = \emptyset$. The event B occurs together with one of the events of $\mathcal{S}$. According to formula (FTP) we have

$$P(B) = P(A_1) \cdot P(B|A_1) + P(A_2) \cdot P(B|A_2).$$

Replacing $P(A_1) = 3/4,\ P(A_2) = 1/4,\ P(B|A_1) = 2/8,\ P(B|A_2) = 1/9$, we obtain $P(B) = 31/144$.

Suppose that the assumptions of the theorem on total probability are satisfied. Then one can also ask about the probability of A_i under the condition that the event B occurred. This probability will be

$$P(A_i|B) = \frac{P(A_i) \cdot P(B|A_i)}{P(B)} = \frac{P(A_i) \cdot P(B|A_i)}{\sum_{k=1}^{n} P(A_k) \cdot P(B|A_k)}. \tag{2.6}$$

[3]Thomas Bayes (1702-1762), British mathematician.

This formula is referred to as **Bayes' formula**. It is used to evaluate the probabilities of the events A_i which form a (CSE), *after* the result of an experiment is known. Actually these are the probabilities of the *causes* of the occurrence of an event.

Bayes' formula contains the probabilities

$$P(A_i), P(B|A_i), \quad i \in \mathbb{N}_n,$$

which can be calculated *before* the experiment. For this reason they are called **probabilities a priori**.

By performing the experiment, we find out that the event B occurred and we want to establish the probabilities

$$P(A_i|B), \quad \text{for } i = 1, \ldots, n,$$

where $P(A_i|B)$ represents the probability that the occurrence of B is due to the occurrence of event A_i (together with A_i). These probabilities are called **probabilities a posteriori**.

Example 2.5 *In the same experiment as in Example 2.4, suppose that a white ball has been drawn (B has been realized). What is the probability that it came from an urn of type 1 (the probability of being realized together with A_1)?*

Solution. Applying Bayes' formula we have

$$P(A_1|B) = \frac{P(A_1) \cdot P(B|A_1)}{P(B)} = \frac{\frac{3}{4} \cdot \frac{1}{4}}{\frac{31}{144}} = \frac{27}{31}.$$

2.5 Probabilistic schemes

2.5.1 Binomial scheme

Suppose we have an urn containing N_1 white balls and N_2 black balls. We extract a ball, note its color and then we put it back into the urn. Obviously, the structure of the urn remains unchanged from one extraction to another. If we denote by A the event "*the extracted ball is white*", then $\overline{A}$ will be the event "*the extracted ball is black*" and the probabilities of these events will be

$$P(A) = \frac{N_1}{N} = p, \quad P(\overline{A}) = \frac{N_2}{N} = q, \text{ with } N = N_1 + N_2.$$

Of course, $p + q = 1$ and p, q are unchanged from one extraction to another.

We perform n extractions according to the procedure described above. We ask the probability of the event

X_k : "among the n extracted balls, k are white and $n - k$ are black".

Solution. A favorable succession of events $A, \overline{A}$ is the event

$$B = \underbrace{A\ A \ldots A}_{k \text{ times}}\ \underbrace{\overline{A}\ \overline{A}\ \ldots\ \overline{A}}_{n-k \text{ times}}.$$

Thus, $P(B) = p^k q^{n-k}$. Since there are $\binom{n}{k}$ such favorable events with k occurrences of A and $n-k$ occurrences of $\overline{A}$ (see Example 1.12, p. 15) and since these $\binom{n}{k}$ events are incompatible, we have

$$P(X_k) = \binom{n}{k} p^k q^{n-k}, \text{ for } k = 0, 1, \ldots, n.$$

Remark 2.3 *The probability $P(X_k)$ is the coefficient of x^k in the polynomial $\varphi_n(x) = (p\,x + q)^n$. For this reason, this function is called the* generating function *for the binomial scheme.*

The binomial scheme was first studied by Bernoulli[4] For this reason, it is also referred to as **the Bernoulli scheme**. An application of this scheme is presented in Section 2.14, p. 62.

2.5.2 The multinomial scheme

This scheme is a generalization of the binomial scheme. Suppose we are given an urn containing a total of N balls of k colors denoted $c_1, c_2, \ldots, c_k$. Specifically, there are N_i balls of color c_i, for $i \in \mathbb{N}_k$, $N_1 + \ldots + N_k = N$. As in the previous case, we extract a ball, we note the color, and we put it back into the urn. If we denote by A_i the event "*the extracted ball has the color c_i*", then

$$p_i = P(A_i) = \frac{N_i}{N}, \text{ for } i \in \mathbb{N}_k$$

and $\mathcal{S} = \{A_1, A_2, \ldots, A_k\}$ will be a complete system of events, with the sure event E : "*a ball is extracted*". Obviously, $p_1 + p_2 + \ldots + p_k = 1$.

We perform n extractions in the way described above. We ask the probability of the event

$$X_{n_1,\ldots,n_k} : \text{"among the } n \text{ extracted balls, } n_i \text{ have color } c_i, \text{ for } i \in \mathbb{N}_k\text{"}.$$

Of course, $n_1 + n_2 + \ldots + n_k = n$.

Solution. We will show that

$$P(X_{n_1,n_2,\ldots,n_k}) = \frac{n!}{n_1! \cdot n_2! \cdot \ldots \cdot n_k!} \cdot p_1^{n_1} p_2^{n_2} \ldots p_k^{n_k}. \tag{2.7}$$

A favorable succession of events $A_1, A_2, \ldots, A_k$ is the event

$$B = \underbrace{A_1\ A_1 \ldots\ A_1}_{n_1 \text{ times}}\ \underbrace{A_2\ A_2 \ldots\ A_2}_{n_2 \text{ times}}\ \ldots\ \underbrace{A_k\ A_k \ldots\ A_k}_{n_k \text{ times}}.$$

Its probability is $P(B) = p_1^{n_1} p_2^{n_2} \ldots p_k^{n_k}$. Since there are

$$\binom{n}{n_1, n_2, \ldots, n_k}$$

such favorable events with n_i occurrences of color c_i for $1 \leq i \leq n$ (see Theorem 1.14, p. 20) and since these events are incompatible, formula (2.7) holds.

Theorem 1.15 allows us to state the following remark.

[4]James Bernoulli (also known as Jacob I) (1654-1705), Swiss mathematician.

Remark 2.4 *The probability in (2.7) is the coefficient of* $x_1^{n_1}x_2^{n_2}\dots x_k^{n_k}$ *in the multivariate polynomial*

$$\Phi_n(x_1, x_2, \dots, x_k) = (p_1x_1 + p_2x_2 + \dots + p_kx_k)^n.$$

For this reason, this function is called the generating function *for the multinomial scheme.*

2.5.3 The hypergeometric scheme

Suppose we have an urn containing N balls of which N_1 are white and $N_2 = N - N_1$ are black. Let n balls be drawn successively, without reintroducing them back into the urn. We ask the probability of the event

$$Y_{n_1,n_2} : \text{“among the } n \text{ extracted balls, } n_1 \text{ are white and } n_2 \text{ are black”}.$$

Solution. If we denote the white balls by $w_1, w_2, \dots, w_{N_1}$, and the black balls by $b_1, b_2, \dots, b_{N_2}$, then there are $\binom{N}{n}$ ways of extracting n balls from an urn of N balls and the favorable sequence of balls are of the form

$$w_{i_1}, w_{i_2}, \dots, w_{i_{n_1}},\ b_{j_1}, b_{j_2}, \dots, b_{j_{n_2}}.$$

There are $\binom{N_1}{n_1}$ ways of extracting n_1 white balls out of N_1 white balls. For each fixed sequence of white balls, there are $\binom{N_2}{n_2}$ ways of extracting n_2 black balls out of N_2 black balls. So, in total there are

$$\binom{N_1}{n_1}\binom{N_2}{n_2}$$

favorable extractions and therefore the required probability is

$$P(Y_{n_1,n_2}) = \frac{\binom{N_1}{n_1}\binom{N_2}{n_2}}{\binom{N}{n}} = \frac{\binom{N_1}{n_1}\binom{N_2}{n_2}}{\binom{N_1+N_2}{n_1+n_2}}. \tag{2.8}$$

An application of this scheme is presented in Section 2.14, p. 63.

2.5.4 The generalized hypergeometric scheme

One can generalize the hypergeometric scheme in the following way: suppose that the urn contains N balls of k colors, denoted $c_1, c_2, \dots, c_k$. More precisely, there are N_i balls of color c_i $(1 \le i \le n)$ and $N_1 + \dots + N_k = N$. We extract n balls and we ask the probability of the event

$$Y_{n_1,\dots,n_k} : \text{“among the } n \text{ extracted balls, } n_i \text{ have color } c_i,\ i \in \mathbb{N}_k\text{”}.$$

Of course, $n_1 + n_2 + \dots + n_k = n$.

Analogous arguments as for the hypergeometric scheme will lead us to the formula

$$P(Y_{n_1,n_2,\dots,n_k}) = \frac{\binom{N_1}{n_1}\binom{N_2}{n_2}\dots\binom{N_k}{n_k}}{\binom{N}{n}}. \tag{2.9}$$

2.5.5 The Poisson's urns scheme

In this scheme we have n urns denoted U_i, $1 \leq i \leq n$, each of which containing white and black balls, but in different proportions. We introduce the following notations:

$$\begin{aligned} p_i &= \text{the probability to extract a white ball from the urn } U_i, \\ q_i &= \text{the probability to extract a black ball from the urn } U_i, \end{aligned}$$

for $1 \leq i \leq n$ and with $p_i + q_i = 1$. We extract n balls, one from each urn, and we ask the probability of the event

Z_k : "among the n extracted balls, k are white and $n-k$ are black".

Solution. In order to simplify the presentation, we consider first a particular case, with $n = 4$ urns and we calculate the probability to have $k = 3$ white balls and $n - k = 1$ black ball. For $i \in \mathbb{N}_4$, we consider the events A_i :"*the ball extracted from U_i is white*". Then the complementary events $\overline{A}_i$ will be "*the ball extracted from U_i is black*".

The events illustrating the favorable extractions are:

$$\begin{aligned} B_1 &= A_1 \cap A_2 \cap A_3 \cap \overline{A}_4, \\ B_2 &= A_1 \cap A_2 \cap \overline{A}_3 \cap A_4, \\ B_3 &= A_1 \cap \overline{A}_2 \cap A_3 \cap A_4, \\ B_4 &= \overline{A}_1 \cap A_2 \cap A_3 \cap A_4, \end{aligned}$$

the events involved in each of the products being totally independent. Therefore, the probabilities of the favorable events are

$$\begin{aligned} P(B_1) = p_1p_2p_3q_4, \quad P(B_2) = p_1p_2q_3p_4, \\ P(B_3) = p_1q_2p_3p_4, \quad P(B_4) = q_1p_2p_3p_4. \end{aligned}$$

Further, the event of interest Z_3 can be written as

$$Z_3 = B_1 \cup B_2 \cup B_3 \cup B_4.$$

Since the events in this sum are incompatible, the required probability will be the sum of $P(B_i)$,

$$P(Z_3) = p_1p_2p_3q_4 + p_1p_2q_3p_4 + p_1q_2p_3p_4 + q_1p_2p_3p_4. \tag{2.10}$$

If we consider the polynomial

$$Q_4(x) = (p_1x + q_1)(p_2x + q_2)(p_3x + q_3)(p_4x + q_4),$$

then $P(Z_3)$ represents the coefficient of x^3 in the polynomial $Q_4(x)$.

In the general case, we consider the polynomial

$$Q_n(x) = \prod_{i=1}^{n}(p_ix + q_i)$$

and similar arguments are used to prove that the probability $P(Z_k)$ is the coefficient of x^k in $Q_n(x)$.

Remark 2.5 *If all the urns are identical, then $p_i = p$, $q_i = q$ and the polynomial $Q_n(x)$ reduces to*

$$Q_n(x) = (px + q)^n.$$

The coefficient of x^k will be

$$\binom{n}{k} p^k q^{n-k},$$

which is exactly the probability in the binomial scheme. Indeed, the fact that the urns are identical is equivalent to the situation when we have one urn and we reintroduce the ball into the urn after each extraction.

2.6 Random variables

There are many definitions of a random variable (r.v.), but we will adopt the following one:

Definition 2.6 *A real variable that takes values depending on the outcome of an experiment, thus depending on chance, is called a **random variable**. It is actually a real-valued function defined on the sample space.*

The number of bacteria per unit area in the study of drug control, the number of voters favoring a certain candidate are two examples of random variables.

Definition 2.7 *Let X be a random variable. The **distribution function** (or **repartition function**) of X is defined as $F : \mathbb{R} \to [0, 1]$,*

$$F(x) = P(X < x), \text{ for all } x \in \mathbb{R}.$$

From the point of view of probability theory, a random variable is completely characterized by its distribution function. It is to be regarded as known when its distribution function is given. The distribution function F defined above has the following properties:

P1. $P(a \leq X < b) = F(b) - F(a)$, for all $a, b \in \mathbb{R}$, $a < b$.
Proof. If we consider the events

$$\begin{aligned} A &: a \leq X < b, \\ B &: X < b, \\ C &: X < a, \end{aligned}$$

we have $B = A \cup C$ and $A \cap C = \emptyset$. Thus, $P(B) = P(A) + P(C)$ and therefore

$$P(A) = P(B) - P(C) = P(X < b) - P(X < a) = F(b) - F(a).$$

P2. The function F is nondecreasing.
Proof. Let $x_1 \leq x_2$. Then $P(x_1 \leq X < x_2) \geq 0$, whence, using **P1**, $F(x_2) - F(x_2) \geq 0$.

P3. The function F is continuous to the left:

$$F(x-0) = F(x), \quad \text{for all } x \in \mathbb{R}.$$

P4. $\lim_{x\to-\infty} F(x) = 0, \quad \lim_{x\to\infty} F(x) = 1.$

In practice, two types of variables are of particular importance: the *discrete* and the *continuous* random variables.

2.6.1 Discrete random variables

Definition 2.8 *A random variable X is called* ***discrete*** *(d.r.v.) if it can take only* finitely *or* countable *many values. Thus, it is determined by the values $x_1, x_2, \ldots, x_n$ (or $x_1, x_2, \ldots$) it can take and by the probabilities $p_i = P(X = x_i)$ with which it takes these values. The values p_i must satisfy the condition $\sum_i p_i = 1$.*

So, a d.r.v. can be written as

$$X : \begin{pmatrix} x_1 & x_2 & \ldots & x_n \\ p_1 & p_2 & \ldots & p_n \end{pmatrix}, \quad p_i > 0, \ \sum_{i=1}^{n} p_i = 1,$$

if it take a finite number of values, or as

$$X : \begin{pmatrix} x_1 & x_2 & \ldots & x_n & \ldots \\ p_1 & p_2 & \ldots & p_n & \ldots \end{pmatrix}, \quad p_i > 0, \ \sum_{i=1}^{\infty} p_i = 1,$$

if it take a countable number of values. In order not to consider separate cases we will sometimes adopt the second notation for a finite r.v., allowing $p_i = 0$ for $i = n+1, n+2, \ldots$, and we will denote

$$\mathcal{X} = \{x_1, x_2, \ldots, x_n, \ldots\} \text{ or } \mathcal{X} = \{x_1, x_2, \ldots, x_n\}.$$

Definition 2.9 *The mapping $f_X : \mathcal{X} \to \{p_1, p_2, \ldots, p_n, \ldots\}$,*

$$f_X(x_i) = p_i = P(X = x_i),$$

is called the ***probability function*** *of the d.r.v. X.*

The probability function f_X defines completely the r.v. X.

Proposition 2.5 *With the notations above, we have*

$$F(x) = \sum_{x_i<x} p_i.$$

Proof. Let $k \in \mathbb{N}$ be such that $x_k < x \leq x_{k+1}$. Then

$$\begin{aligned} F(x) &= P(X < x) \\ &= P(X = x_1 \ \cup \ X = x_2 \ \cup \ \ldots \ \cup \ X = x_k) \\ &= p_1 + p_2 + \ldots + p_k = \sum_{i=1}^{k} p_i = \sum_{x_i<x} p_i \end{aligned}$$

■

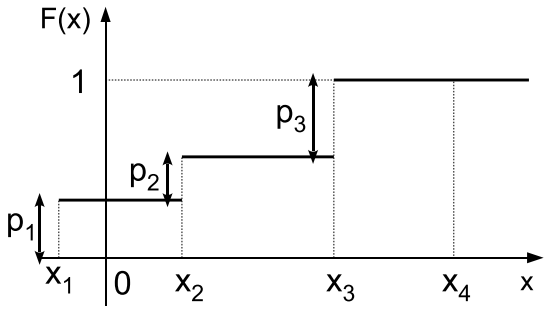


Figure 2.1: The repartition function of a discrete random variable.

Definition 2.10 *The random variables X and Y are called* ***independent*** *if*

$$P(X = x, Y = y) = P(X = x) \cdot P(Y = y).$$

The random variables $X_1, \ldots, X_n$ are called ***totally independent*** *if for all $m \leq n$ and for all m-tuple $(i_1, i_2, \ldots, i_m)$, $1 \leq i_1 \leq i_2 \leq \ldots \leq i_m \leq n$ we have*

$$\begin{aligned} &P(X_{i_1} = x_1, X_{i_2} = x_2, \ldots, X_{i_m} = x_m) \\ &= P(X_{i_1} = x_1) \cdot P(X_{i_2} = x_2) \cdot \ldots \cdot P(X_{i_m} = x_m). \end{aligned}$$

2.6.2 Continuous random variables

Definition 2.11 *A r.v. is called* ***absolutely continuous*** *when its distribution function F can be represented as*

$$F(x) = \int_{-\infty}^{x} f(t)\, dt.$$

In this case, the function $f : \mathbb{R} \to \mathbb{R}$ is called the ***density*** *of the distribution.*

Since $\lim\limits_{x \to \infty} F(x) = 1$, the density must satisfy the equality

$$\int_{-\infty}^{\infty} f(x)\, dx = 1.$$

Another immediate properties of a continuous r.v. are

$$\begin{aligned} &P(a \leq X < b) = F(b) - F(a) = \int_a^b f(x)\, dx, \\ &P(X = a) = 0. \end{aligned}$$

An example of continuous distribution is the **normal distribution**, also named Gaussian distribution, after Gauss:[5]

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}.$$

This probability distribution has a great importance in many fields. It

[5]Carl Friedrich Gauss (Gauß) (1777-1855), German mathematician and physicist, who contributed significantly to many fields, including number theory, analysis, differential geometry, geodesy, magnetism, astronomy and optics.

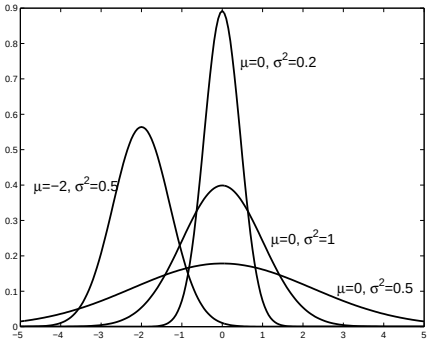


Figure 2.2: Some normal density functions.

consist of a family of distributions of the same general form, differing in their location and scale parameters: the mean μ and standard deviation σ, respectively. Its graph is often called the bell curve because the graph of its probability density resembles a bell (see Figure 2.2).

In the following we restrict ourselves to discrete random variables.

2.7 The sum of random variables

Definition 2.12 *Let X, Y be two discrete random variables. The sum of X, Y is also a d.r.v., denoted $X + Y$ and defined by*

$$P(X + Y = z) = \sum_{x \in \mathcal{X}} P(X = x,\, Y = z - x), \tag{2.11}$$

For the sum of n random variables $X_1, \ldots, X_n$, the definition can be easily adapted:

$$P(X_1 + \ldots + X_n = z) = \sum_{x_1 \in \mathcal{X}_1, \ldots, x_{n-1} \in \mathcal{X}_{n-1}} P(X_1 = x_1, \ldots, X_{n-1} = x_{n-1}, X_n = z - (x_1 + \ldots + x_{n-1})).$$

If X, Y are independent random variables, the sum (2.11) becomes

$$P(X + Y = z) = \sum_{x \in \mathcal{X}} P(X = x) \cdot P(Y = z - x).$$

Example 2.6 *Let $p \in (0, 1), q = 1 - p$ and X, Y be independent d.r.v. generated by binomial schemes:*

$$X : \begin{pmatrix} i \\ \binom{n}{k} p^i q^{n-i} \end{pmatrix},\ i = 0, \ldots, n, \quad Y : \begin{pmatrix} j \\ \binom{m}{j} p^j q^{m-j} \end{pmatrix},\ j = 0, \ldots, m.$$

For the sum $X + Y$ we have

$$\begin{aligned} P(X + Y = k) &= \sum_{j=0}^{k} P(X = j) \cdot P(Y = k - j) \\ &= \sum_{j=0}^{k} \binom{n}{j} p^j q^{n-j} \cdot \binom{m}{k-j} p^{k-j} q^{m-(k-j)} \\ &= p^k q^{n+m-k} \sum_{j=0}^{k} \binom{n}{j} \binom{m}{k-j} \\ &= p^k q^{n+m-k} \binom{m+n}{k}. \end{aligned}$$

In conclusion,

$$X+Y:\begin{pmatrix} k \\ \binom{m+n}{k}p^k q^{m+n-k} \end{pmatrix},\ i=0,\ldots,m+n.$$

2.8 Examples of discrete random variables

2.8.1 Binomial distribution

Definition 2.13 *The discrete random variable X has a* ***binomial distribution*** *with parameters n, p ($n \in \mathbb{N},\ p \in (0,1)$) if its probability function is $f_X : \{0,1,\ldots,n\} \to (0,1]$,*

$$f_X(k) = \binom{n}{k} p^k q^{n-k} \text{ for all } k \in \mathbb{N}_n \cup \{0\},\ \text{where } q = 1-p. \tag{2.12}$$

The definition is correct since $f_X > 0$ and $\sum_{k=0}^{n} f_X(k) = 1$, as

$$\sum_{k=0}^{n} \binom{n}{k} p^k q^{n-k} = (p+q)^n = 1.$$

With the notations in Definition 2.8, this r.v. can also be denoted as

$$X:\begin{pmatrix} k \\ \binom{n}{k}p^k q^{n-k} \end{pmatrix},\quad k \in \{0,1,\ldots,n\}. \tag{2.13}$$

The binomial distribution can be defined as follows: suppose we repeat a certain experiment n times and that the individual experiments in this series are totally independent. Suppose also that in every experiment an event A occurs with probability p, independent of the number of the experiment. The r.v. X will represent the number of occurrences of A in such a series of n experiments. As we have already seen in Section 2.5.1, the probability function f_X is indeed the one given in (2.12).

2.8.2 The hypergeometric distribution

Definition 2.14 *The discrete random variable X has a* ***hypergeometric distribution*** *with parameters n, a, b ($n, a, b \in \mathbb{N},\ n \le a,\ n \le b$) if its probability function is $f_X : \{0,1,\ldots,n\} \to (0,1]$,*

$$f_X(k) = \frac{\binom{a}{k}\binom{b}{n-k}}{\binom{a+b}{n}},\ \text{for all } k \in \{0,1,\ldots,n\}. \tag{2.14}$$

The definition is correct, since $f_X > 0$ and $\sum_{k=0}^{n} f_X(k) = 1$. The last equality is due to the Vandermonde[6] identity

$$\sum_{k=0}^{n} \binom{a}{k}\binom{b}{n-k} = \binom{a+b}{n}, \tag{2.15}$$

which follows by identifying the coefficients of x^n in the equality

$$(1+x)^{a+b} = (1+x)^a (1+x)^b.$$

The hypergeometric scheme can be generated using the hypergeometric scheme (see Section 2.5.3).

[6] Alexandre-Théophile Vandermonde (1735–1796), French musician and chemist.

2.8.3 Poisson distribution

Definition 2.15 *The discrete random variable X has a* ***Poisson***[7] ***distribution*** *with parameter $\lambda > 0$ if its probability function is $f_X : \{0, 1, 2, \ldots\} \to (0, 1]$,*

$$f_X(k) = \frac{\lambda^k}{k!} e^{-\lambda}. \tag{2.16}$$

We have $f_X > 0$ and

$$\sum_{k=0}^{\infty} f_X(k) = e^{-\lambda} \sum_{k=0}^{\infty} \frac{\lambda^k}{k!} = e^{-\lambda} \cdot e^{\lambda} = 1,$$

so the definition is correct.

2.8.4 Geometric distribution

Definition 2.16 *The discrete random variable X has a* ***geometric distribution*** *with parameter $p \in (0, 1]$ if its probability function is $f_X : \mathbb{N} \to (0, 1]$,*

$$f_X(k) = p\, q^{k-1}, \text{ for all } k \in \mathbb{N} \text{ and with } q = 1 - p. \tag{2.17}$$

The geometric distribution can be generated as follows: consider an experiment and let A be an event which can occur as a result of the experiment. Let $p = P(A)$. We repeat the experiment independently, until the first occurrence of A and we denote by A_i the event consisting in the occurrence of A at the i-th experience, $i = 1, 2, \ldots$. Then, for the random variable X representing the number of experiments performed until the first occurrence of A, we have

$$f_X(k) = P(X = k) = P(\overline{A}_1 \cap \overline{A}_2 \cap \ldots \cap \overline{A}_{k-1} \cap A_k) = p\, q^{k-1}.$$

Of course, $f_X > 0$ and

$$\sum_{k=1}^{\infty} p\, q^{k-1} = \frac{p}{1 - q} = 1,$$

the series involved being the geometric one (whence the name *geometric* for the distribution).

2.8.5 Negative binomial distribution

Definition 2.17 *The discrete random variable X has a* ***negative binomial distribution,*** *or* ***binomial with negative exponent*** *with parameter $p \in (0, 1]$ if its probability function is $f_X : \{n, n + 1, \ldots\} \to (0, 1]$,*

$$f_X(k) = \binom{k-1}{n-1} p^n q^{k-n}, \tag{2.18}$$

for all $k \in \{n, n + 1, \ldots\}$, $q = 1 - p$.

[7] Siméon Denis Poisson (1781-1840), French mathematician and physicist.

In order to prove the correctness, we use the general binomial series

$$\begin{aligned}(1+x)^\alpha &= 1+\alpha x+\frac{\alpha(\alpha-1)}{2!}x^2+\frac{\alpha(\alpha-1)(\alpha-2)}{3!}x^3+\dots \\ &= \sum_{j=0}^{\infty}\binom{\alpha}{j}x^j \quad \text{with } \alpha\in\mathbb{R},\ |x|<1. \end{aligned} \tag{2.19}$$

Here $\binom{\alpha}{j}$ represents the *binomial number*

$$\binom{\alpha}{j}=\frac{\alpha(\alpha-1)\dots(\alpha-j+1)}{j\,!},$$

defined in formula (1.9), p. 15.

In the particular case $x=-q$, $\alpha=-n$, the series (2.19) reads

$$\begin{aligned}(1-q)^{-n} &= \sum_{j=0}^{\infty}\binom{n+j-1}{j}q^j=\sum_{j=0}^{\infty}\binom{n+j-1}{n-1}q^j \\ &= \sum_{k=n}^{\infty}\binom{k-1}{n-1}q^{k-n},\end{aligned}$$

whence

$$\sum_{k=n}^{\infty}f_X(k)=p^n(1-q)^{-n}=1$$

and therefore the r.v. is well defined.

The negative binomial r.v. can be generated as follows: consider an experiment and let A be an event which can occur by performing the experiment. Let $p=P(A)>0$. We repeat the experiment independently, until the event A takes place n times, and then we stop. X is the random variable which represents the number of experiments performed until A was realized n times. The possible values for X are indeed $n, n+1, n+2, \dots$. Then, for a fixed k ($k\geq n$), $f_X(k)$ can be calculated as follows:

$$f_X(k)=P(X=k)=P(A_{k-1}^{n-1}\cap A_k^1),$$

where A_k^1 and A_{k-1}^{n-1} are the events

$$\begin{aligned}A_k^1 &: \text{“}A \text{ occurs in the } k\text{-th experiment”},\\ A_{k-1}^{n-1} &: \text{“}A \text{ occurs } n-1 \text{ times in the first } k-1 \text{ experiments”}.\end{aligned}$$

Their probabilities are, using the binomial scheme (see Section 2.5.1),

$$\begin{aligned}P(A_k^1) &= p,\\ P(A_{k-1}^{n-1}) &= \binom{k-1}{n-1}p^{n-1}(1-p)^{(k-1)-(n-1)}=\binom{k-1}{n-1}p^{n-1}q^{k-n}.\end{aligned}$$

Since these events are independent, formula (2.18) is proved.

2.8.6 Poisson's urns distribution

Definition 2.18 *The discrete random variable X has a **Poisson's urns distribution,** with parameters $p_i \in (0,1]$, $i \in \{1,\dots,n\}$ if its probability function is $f_X : \{0,1,\dots,n\} \to (0,1]$,*

$$f_X(k) = \widetilde{p}_k, \tag{2.20}$$

for all $k \in \{0,1,\dots,n\}$, where $\widetilde{p}_k$ is the coefficient of x^k in the polynomial

$$Q_n(x) = \prod_{i=1}^{n}(p_i x + q_i).$$

Here $q_i = 1 - p_i$. This r.v. is well defined since $f_X > 0$ and

$$\sum_{k=0}^{n} \widetilde{p}_k = Q_n(1) = \prod_{i=1}^{n}(p_i + q_i) = 1.$$

This distribution is generated by the Poisson's urns scheme (see Section 2.5.5).

2.9 Expected value and variance of a discrete random variable

Let X be a d.r.v. with possible values $x_1, x_2, \dots, x_n, \dots$, let $p_k = P(X = x_k)$ and $i \in \mathbb{N}$.

Definition 2.19 *The number*

$$\alpha_i = \sum_k x_k^i \, p_k,$$

*if the series converges absolutely, is called the i-**th moment** of X.*

The number

$$\mu_i = \sum_k (x_k - \alpha_1)^i p_k,$$

*is called the i-**th central moment** of X.*

Of a particular importance are the moments α_1 and μ_2.

Definition 2.20 *The number*

$$\alpha_1 = \sum_k x_k \, p_k,$$

*is called the **expectation** or **expected value** or **mean** of X and is denoted $E(X)$ or $M(X)$.*

The expected value is a measure of the center of distribution in the following sense: if p_k are interpreted as masses attached to the points $x_k \in \mathbb{R}$, then $E(X)$ is the center of mass of this system on the real axis:

$$\alpha_1 = E(X) = \frac{\sum x_k \, p_k}{\sum p_k}.$$

Example 2.7 *If we consider the random variable*

$$X : \begin{pmatrix} k \\ 1/6 \end{pmatrix}_{k\in\mathbb{N}_6}$$

representing the number of face which appears when rolling a die, then its expected value is

$$E(X) = \sum_{k=1}^{6} k \cdot p_k = \frac{1}{6}\sum_{k=1}^{6} k = 3.5 \ .$$

Definition 2.21 *The number μ_2 is called the **variance** of X and is denoted by $var(X)$ or $D^2(X)$. Thus,*

$$var(X) = \sum_{k} \left(x_k - E(X)\right)^2 p_k = E\left((X - E(X))^2\right).$$

*The square root of the variance is called the **standard deviation** or **dispersion** and is denoted by $\sigma(X)$ or $D(X)$:*

$$\sigma(X) = \sqrt{var(X)}.$$

Before presenting the main properties of the expectation and variance, we give the following definition.

Definition 2.22 *Let $X : \begin{pmatrix} x_k \\ p_k \end{pmatrix}$, $k \in I$, be a d.r.v., $i \in \mathbb{N}$ and $a \in \mathbb{R}$. Then the random variables $a + X$, aX and X^i are defined, respectively, as*

$$\begin{aligned} a + X &: \begin{pmatrix} a + x_k \\ p_k \end{pmatrix},\ k \in I, \\ a\,X &: \begin{pmatrix} a\,x_k \\ p_k \end{pmatrix},\ k \in I, \\ X^i &: \begin{pmatrix} x_k^i \\ p_k \end{pmatrix},\ k \in I. \end{aligned}$$

Proposition 2.6 *The expectation and the variance have the following properties:*

P1. $E(a) = a$, *if a is the constant r.v.* $a : \binom{a}{1}$.

P2. $E(aX) = aE(X)$.

P3. $E(X + Y) = E(X) + E(Y)$; *Indeed, if* $Z = X + Y$,

$$X : \begin{pmatrix} x_k \\ p_k \end{pmatrix},\ k \in \mathbb{N}_n, \quad Y : \begin{pmatrix} y_k \\ r_k \end{pmatrix},\ k \in \mathbb{N}_m,\ \text{then } Z : \begin{pmatrix} x_i + y_j \\ p_{ij} \end{pmatrix},$$

with $p_{ij} = P(X = x_i,\ Y = y_j),\ i \in \mathbb{N}, j \in \mathbb{N}_m$. Thus,

$$E(Z) = \sum_{i=1}^{n}\sum_{j=1}^{m}(x_i + y_j)p_{ij} = \sum_{i=1}^{n} x_i \sum_{j=1}^{m} p_{ij} + \sum_{j=1}^{m} y_j \sum_{i=1}^{n} p_{ij}.$$

But

$$\begin{aligned}\sum_{j=1}^{m} p_{ij} &= P(Z = x_i + y_1) + \ldots + P(Z = x_i + y_m) \\ &= P(Z = x_i + y_1 \ \cup \ \ldots \cup Z = x_i + y_m) \\ &= P(X = x_i) = p_i,\end{aligned}$$

for all $i \in \mathbb{N}_n$. *Analogously we calculate* $\sum_{i=1}^{n} p_{ij} = r_j$, *for all* $j \in \mathbb{N}_m$, *whence the conclusion*

$$E(X+Y) = \sum_{i=1}^{n} x_i p_i + \sum_{j=1}^{m} y_j r_j = E(X) + E(Y).$$

P4. $E(aX + bY) = aE(X) + bE(Y), \quad E\left(\sum_{i=1}^{n} a_i X_i\right) = \sum_{i=1}^{n} a_i E(X_i)$

P5. $E(X - E(X)) = 0.$

P6. *If* $X \geq 0$, *then* $E(X) \geq 0.$

P6'. *If* $X \geq Y$, *then* $E(X) \geq E(Y).$

P7. $var(X) = \alpha_2 - \alpha_1^2 = E(X^2) - (E(X))^2$. *Indeed,*

$$\begin{aligned}var(X) &= E\left((X - E(X))^2\right) \\ &= E\left(X^2 - 2E(X) \cdot X + (E(X))^2\right) \\ &= E(X^2) - 2E(X) \cdot E(X) + (E(X))^2 \\ &= E(X^2) - (E(X))^2.\end{aligned}$$

P8. $var(aX + b) = a^2 var(X)$. *Indeed,*

$$\begin{aligned}var(a\,X + b) &= E\left((aX + b - E(aX + b))^2\right) \\ &= E\left((aX + b - a\,E(X) - b)^2\right) \\ &= E\left(a^2(X - E(X))^2\right) \\ &= a^2 var(X).\end{aligned}$$

Consequences:

$$\begin{aligned}var(b) &= 0, \\ var(X + b) &= var(X), \\ var(aX) &= a^2 var(X).\end{aligned}$$

P9. *If* X, Y *are independent r.v.,* $E(X \cdot Y) = E(X)E(Y)$.

Indeed, let

$$X : \begin{pmatrix} x_k \\ p_k \end{pmatrix}, \ k \in \mathbb{N}_n, \quad Y : \begin{pmatrix} y_k \\ r_k \end{pmatrix}, \ k \in \mathbb{N}_m. \ \textit{Then} \ X \cdot Y : \begin{pmatrix} x_i y_j \\ p_{ij} \end{pmatrix},$$

with

$$p_{ij} = P(X = x_i \ \cap \ Y = y_j) = P(X = x_i) \cdot P(Y = y_j) = p_i r_j.$$

Then

$$\begin{aligned} E(X \cdot Y) &= \sum_{i=1}^{n}\sum_{j=1}^{m} x_i y_j p_{ij} = \sum_{i=1}^{n}\sum_{j=1}^{m} x_i y_j p_i r_j \\ &= \left(\sum_{i=1}^{n} x_i p_i\right)\left(\sum_{j=1}^{m} y_j r_j\right) = E(X)E(Y). \end{aligned}$$

P9'. *If $X_i,\ i \in \mathbb{N}_s$ are totally independent d.r.v., then*

$$E\left(\prod_{i=1}^{s} X_i\right) = \prod_{i=1}^{s} E(X_i).$$

Remark 2.6 *If two d.r.v. are not independent, property **P9** does not hold in general.*

The standard deviation and the variance are a measure of the scattering of the distribution about the expected value.

2.10 Covariance

As we have just seen, it is important to calculate the variance of a sum of r.v., knowing the individual variances. To do this, we need some preliminaries. We start with the following definition.

Definition 2.23 *Let X, Y be two r.v. such that $E(X), E(Y)$, $E(X \cdot Y), var(X)$ and $var(Y)$ exist. The **covariance** of X and Y is defined as*

$$cov(X,Y) = E\left((X - E(X)) \cdot (Y - E(Y))\right). \tag{2.21}$$

Proposition 2.7 *The covariance has the following properties:*

P1. $cov(X,X) = var(X)$.

P2. $cov(X,Y) = E(X \cdot Y) - E(X)E(Y)$.

P2'. *If X and Y are independent, then $cov(X,Y) = 0$.*

P3. *If $X_i,\ i \in \mathbb{N}_n$, then*

$$var\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} var(X_i) + 2\sum_{1 \le i < j \le n} cov(X_i, X_j). \tag{2.22}$$

Proof. *We have, using **P4** from Proposition 2.6 p. 50,*

$$\begin{aligned} var\left(\sum_{i=1}^{n} X_i\right) &= E\left(\left(\sum_{i=1}^{n} X_i - \sum_{i=1}^{n} E(X_i)\right)^2\right) \\ &= E\left(\left(\sum_{i=1}^{n} X_i - E(X_i)\right)^2\right). \end{aligned}$$

Employing now the formula

$$\left(\sum_{i=1}^{n} a_i\right)^2 = \sum_{i=1}^{n} a_i^2 + 2 \sum_{1\le i<j\le n} a_i a_j,$$

for $a_i = X_i - E(X_i)$, we get the conclusion.

Definition 2.24 *The random variables $X_1, X_2, \ldots, X_n$ are said to be **uncorrelated** if*

$$cov(X_i, X_j) = 0, \text{ whenever } i \neq j.$$

We are now able to establish a result regarding the variance of the sum of random variables. More precisely, the following result holds.

Proposition 2.8 *The following statements are true:*

P4. *Totally independent r.v. are uncorrelated.*

P5. *If $X_1, \ldots, X_n$ are uncorrelated r.v., then*

$$var(X_1 + X_2 + \ldots + X_n) = var(X_1) + var(X_2) + \ldots + var(X_n).$$

Proof *The second term in the right-hand side of (2.22) vanishes.*

P6. $var(X + Y) = var(X) + var(Y)$ *if and only if* $cov(X, Y) = 0$.

2.11 Expected values and variance for some discrete random variables

Binomial distribution

Theorem 2.9 *If X is a d.r.v. having a binomial distribution given in (2.13), then*

$$E(X) = np, \ var\,(X) = npq.$$

Proof.

$$E(X) = \sum_{k=0}^{n} k \binom{n}{k} p^k q^{n-k}.$$

To calculate this sum differentiate with respect to x the binomial formula

$$(q + px)^n = \sum_{k=0}^{n} \binom{n}{k} p^k x^k q^{n-k},$$

obtaining

$$n(q + px)^{n-1} p = \sum_{k=1}^{n} \binom{n}{k} p^k q^{n-k} k x^{k-1}. \tag{2.23}$$

For $x = 1$, we obtain

$$E(X) = n(p + q)^{n-1} p = np.$$

Further, using the property P7 of Proposition 2.6,

$$\text{var}\,(X) = E(X^2) - (E(X))^2, \tag{2.24}$$

where

$$E(X^2) = \sum_{k=0}^{n} k^2 \binom{n}{k} p^k q^{n-k}.$$

If now we multiply the identity (2.23) by x, we differentiate it (with respect to x) and then we replace $x = 1$, we get

$$E(X^2) = np\,(1 + (n-1)p).$$

Replacing it in (2.24), we finally obtain the conclusion

$$\text{var}\,(X) = np\,(1 + np - p) - n^2p^2 = np\,q.$$

■

Hypergeometric distribution

Theorem 2.10 *If X is a d.r.v. having a hypergeometric distribution given in (2.14), then*

$$E(X) = np,\ \ var\ (X) = np\,q\frac{a+b-n}{a+b-1},\ \ with\ p = \frac{a}{a+b},\ \ q = \frac{b}{a+b}.$$

Proof.

$$E(X) = \frac{1}{\binom{a+b}{n}} \sum_{k=0}^{n} k \binom{a}{k} \binom{b}{n-k}.$$

Since

$$k\binom{a}{k} = a\binom{a-1}{k-1}, \tag{2.25}$$

we can write

$$E(X) = \frac{1}{\binom{a+b}{n}} \sum_{k=1}^{n} a \binom{a-1}{k-1} \binom{b}{n-k} = a\frac{\binom{a+b-1}{n-1}}{\binom{a+b}{n}},$$

where in the last equality we used the Vandermonde equality (2.15). After simplifications, the conclusion holds.

For the variance we first evaluate

$$\begin{aligned} E(X^2) &= \frac{1}{\binom{a+b}{n}} \sum_{k=1}^{n} k^2 \binom{a}{k} \binom{b}{n-k} \\ &= \frac{1}{\binom{a+b}{n}} \left(\sum_{k=1}^{n} k \binom{a}{k} \binom{b}{n-k} + \sum_{k=2}^{n} k(k-1) \binom{a}{k} \binom{b}{n-k} \right). \end{aligned}$$

Next we use (2.25) and the relation

$$k(k-1)\binom{a}{k} = a(a-1)\binom{a-2}{k-2}. \tag{2.26}$$

Then again Vandermonde type equalities occur and we obtain

$$\begin{aligned} E(X^2) &= \frac{a}{\binom{a+b}{n}} \binom{a+b-1}{n-1} + \frac{a(a-1)}{\binom{a+b}{n}} \binom{a+b-2}{n-2} \\ &= \frac{n}{a+b} + n(n-1)\frac{a(a-1)}{(a+b)(a+b-1)}. \end{aligned}$$

Replacing now $E(X)$ and $E(X^2)$ in formula (2.24) and making some calculations, we obtain the conclusion. ■

Poisson distribution

Theorem 2.11 *If X is a d.r.v. having a Poisson distribution given in (2.16), then*

$$E(X) = \lambda, \quad var(X) = \lambda.$$

Proof.

$$E(X) = e^{-\lambda} \sum_{k=0}^{\infty} k \frac{\lambda^k}{k!} = \lambda e^{-\lambda} \sum_{k=1}^{\infty} k \frac{\lambda^{k-1}}{(k-1)!} = \lambda e^{-\lambda} e^{\lambda} = \lambda.$$

Then we evaluate $E(X^2)$, obtaining

$$\begin{aligned} E(X^2) &= e^{-\lambda} \sum_{k=0}^{\infty} k^2 \frac{\lambda^k}{k!} = \lambda e^{-\lambda} \sum_{k=1}^{\infty} k \frac{\lambda^{k-1}}{(k-1)!} \\ &= \lambda e^{-\lambda} \left(\sum_{k=1}^{\infty} (k-1) \frac{\lambda^{k-1}}{(k-1)!} + \sum_{k=1}^{\infty} \frac{\lambda^{k-1}}{(k-1)!} \right) \\ &= \lambda^2 e^{-\lambda} \sum_{k=2}^{\infty} \frac{\lambda^{k-2}}{(k-2)!} + \lambda e^{-\lambda} \sum_{k=1}^{\infty} \frac{\lambda^{k-1}}{(k-1)!} \\ &= \lambda^2 e^{-\lambda} e^{\lambda} + \lambda e^{-\lambda} e^{\lambda} = \lambda(\lambda + 1). \end{aligned}$$

Finally, $var(X) = E(X^2) - (E(X))^2 = \lambda(\lambda + 1) - \lambda^2 = \lambda$. ■

Geometric distribution

Theorem 2.12 *If X is a d.r.v. having a geometric distribution given in (2.17), then*

$$E(X) = \frac{1}{p}, \quad var(X) = \frac{q}{p^2}.$$

Proof.

$$E(X) = p \sum_{k=1}^{\infty} k q^{k-1} = p \left(\frac{1}{1-q} \right)'_q = \frac{p}{(1-q)^2} = \frac{1}{p}.$$

$$E(X^2) = p \sum_{k=1}^{\infty} k^2 q^{k-1}.$$

To evaluate this sum, we consider the power series

$$\sum_{k=1}^{\infty} k q^{k-1} = \frac{1}{(1-q)^2},$$

we multiply it by q and then we differentiate the equality with respect to q. We obtain

$$\sum_{k=1}^{\infty} k^2 q^{k-1} = \left(\frac{q}{(1-q)^2} \right)'_q = \frac{1+q}{(1-q)^3}.$$

By replacing it in $E(X^2)$ we obtain the conclusion. ■

Negative binomial distribution

Theorem 2.13 *If X is a d.r.v. having a negative binomial distribution, given in (2.18), then*

$$E(X) = \frac{n}{p}, \quad var(X) = \frac{nq}{p^2}.$$

Proof. In this case, if we use the methods above the calculations are quite complicated. This is why we will describe another method for calculating expectation and dispersion. Specifically, we will write the r.v. X as a sum of totally independent random variables whose expected values and variances can be calculated more easily.

So, we consider the following random variables:

$$\begin{aligned} X_1 &= \text{nr. of experiments performed until the first occurrence of } A, \\ X_k &= \text{nr. of experiments performed between the } (k-1)\text{-th} \\ &\quad \text{occurrence and the } k\text{-th occurrence of } A, \end{aligned}$$

for $k = 2, 3, \ldots, n$. These r.v. are totally independent and have a geometric distribution with the same parameter p.

According to Theorem 2.17 one has

$$E(X_k) = \frac{1}{p}, \quad var(X_k) = \frac{q}{p^2},$$

for all $k \in \mathbb{N}_n$.

Since the expectation of the sum of r.v. equals the sum of their expectancies (see **P9'** p. 52), we have

$$E(X) = E(X_1) + \ldots + E(X_n) = nE(X_1) = \frac{n}{p},$$

and thus the formula for $E(X)$ is proved.

For the variance, the analogous result is in general not true, unless the r.v. are totally independent. This result will be proved in the next section (Proposition 2.8, **P5**, p. 53). Fortunately, in our case the r.v. are totally independent, so

$$var(X) = var(X_1) + \ldots + var(X_n) = n \cdot var(X_1) = \frac{nq}{p^2}.$$

■

Poisson's urns distribution

Theorem 2.14 *If X is a d.r.v. having a Poisson's urns distribution, given in Definition 2.18, then*

$$E(X) = \sum_{k=1}^{n} p_k, \quad var(X) = \sum_{k=1}^{n} p_k q_k.$$

Proof. In this case the direct methods are again very complicated, so we will use the same idea of writing X as a sum of n totally independent random variables.

So, let X_k $(1 \leq k \leq n)$ be the r.v. representing the number of white balls extracted from the urn U_k. Then X_k are totally independent and X can be expressed as

$$X = X_1 + X_2 + \ldots X_n.$$

The random variable X_k has the distribution

$$X_k : \begin{pmatrix} 1 & 0 \\ p_k & q_k \end{pmatrix}.$$

A simple calculation shows that

$$E(X_k) = p_k, \quad E(X_k^2) = p_k, \quad var(X_k) = p_k q_k, \quad \text{for } k \in \mathbb{N}_n,$$

whence

$$\begin{aligned} E(X) &= \sum_{i=1}^{n} E(X_k) = \sum_{i=1}^{n} p_k, \\ var(X) &= \sum_{i=1}^{n} var(X_k) = \sum_{i=1}^{n} p_k q_k. \end{aligned}$$

We finish by noting that, in the case when the urns are identical $(p_k = p)$, we recover the expectation and the variance of the binomial distribution. ■

2.12 Markov's inequality, Chebyshev's theorem

Proposition 2.15 (Markov's inequality) [8] *Let X be a random variable and $a > 0$. Then*

$$P(|X| \geq a) \leq \frac{E(|X|^2)}{a^2}. \tag{2.27}$$

Proof. Let Z_a be the r.v.

$$Z_a = \begin{cases} 1, & \text{if } |X| \geq a, \\ 0, & \text{otherwise.} \end{cases}$$

This r.v. satisfies the inequality $|X|^2 \geq a^2 \cdot Z_a$, which further implies

$$E(|X|^2) \geq a^2 E(Z_a) = a^2 P(|X| \geq a).$$

■

The following result, known as Chebyshev's[9] theorem, can be used to determine a lower bound for the probability that a random variable Y of interest falls in an interval $[\mu - k\sigma, \mu + k\sigma]$ around the mean μ.

[8] Andrey Markov (1856-1922), Russian mathematician.

[9] Pafnuty Lvóvich Chebyshev (1821-1894), Russian mathematician

Theorem 2.16 (Chebyshev) *Let Y be a random variable with expected value μ and finite variance σ^2. Then, for any positive constant ε_1 we have*

$$P(|Y-\mu| \geq \varepsilon_1 \sigma) \leq \frac{1}{\varepsilon_1^2}. \tag{2.28}$$

Proof. If in inequality (2.27) one takes $X = Y - E(Y)$ and $a = \varepsilon_1\sigma$, we immediately get the conclusion. ■

The inequality (2.28) can also be written as

$$P(|Y-\mu| < \varepsilon_1 \sigma) \geq 1 - \frac{1}{\varepsilon_1^2},$$

or, if one denotes $\varepsilon = \sigma\varepsilon_1$,

$$\begin{aligned} P(|Y-E(Y)| \geq \varepsilon) &\leq \frac{\text{var}(Y)}{\varepsilon^2}, \text{ or} \\ P(|Y-E(Y)| < \varepsilon) &\geq 1 - \frac{\text{var}(Y)}{\varepsilon^2}. \end{aligned} \tag{2.29}$$

Example 2.8 *The number of customers per day at a certain sales counter denoted by Y has been observed for a long period of time and found to have a mean of 20 customers with a standard deviation of 2 customers*[10]*. The probability distribution of Y is not known. What can be said about the probability that Y will be between 16 and 24 tomorrow?*

Solution. We need to find $P(16 < Y < 24)$. For any $\varepsilon > 0$, inequality (2.29) can be written as

$$P(\mu - \varepsilon < Y < \mu + \varepsilon) \geq 1 - \frac{\sigma^2}{\varepsilon^2}.$$

For $\mu = 20$ and $\sigma = 2$, it follows that $\mu - \varepsilon = 16$ and $\mu + \varepsilon = 24$ if $\varepsilon = 4$. Thus

$$P(16 < Y < 24) \geq 1 - \frac{2^2}{4^2} = \frac{3}{4}.$$

This means that tomorrow's customer total will be between 16 and 24 with high probability (at least 3/4).

Notice that, if σ was 1,

$$P(16 < Y < 24) \geq 1 - \frac{1}{4^2} = \frac{15}{16}.$$

So the value of σ has considerable effect on probabilities associated with intervals.

2.13 Laws of large numbers

The Laws of Large Numbers (LLN), called also the *Golden Theorems of probabilities*, are theorems describing the long-term stability of a random variable. For example, when rolling a die a certain number of times, the average of points will be stable around the expected value 3.5 (see Example 2.7, p. 50), as the number of repetitions becomes large.

[10] How can one estimate the standard deviation after many observations, is a subject of statistics, not treated in this book.

Forms of LLN

The Laws of Large Numbers exist in two forms: the **Weak Law of Large Numbers (WLLN)** and the **Strong Law of Large Numbers (SLLN)**. Both state the convergence of the sample average towards the mean value, the difference between them being the type of convergence.

Definition 2.25 *A sequence $(X_n)_{n\in\mathbb{N}}$ of random variables **converges in probability** to a random variable X (denoted $X_n \xrightarrow{p} X$) if*

$$\lim_{n\to\infty} P\left(|X_n - X|\right) < \varepsilon = 1, \text{ for all } \varepsilon > 0. \tag{2.30}$$

*A sequence $(X_n)_{n\in\mathbb{N}}$ of random variables **converges almost sure** to a random variable X (denoted $X_n \xrightarrow{a.s.} X$) if*

$$P\left(\lim_{n\to\infty} X_n = X\right) = 1, \text{ for all } \varepsilon > 0. \tag{2.31}$$

One can show that, if $X_n \xrightarrow{a.s.} X$, then $X_n \xrightarrow{p} X$, but the reverse implication is not true in general. However, it is known the fact that, if $X_n \xrightarrow{p} X$, then there exists a subsequence $(X_{n_k})_{k\in\mathbb{N}}$ such that $X_{n_k} \xrightarrow{a.s.} X$. For these reasons, the convergence in probability is known as **weak** convergence, while the almost sure convergence is known as **strong** convergence.

Definition 2.26 (Weak Law of Large Numbers) *A sequence $(X_n)_{n\in\mathbb{N}}$ of random variables with $E(X_n) < \infty$ for all $n \in \mathbb{N}$ **obeys the WLLN** if*

$$\frac{1}{n}\sum_{k=1}^{n}\left(X_k - E(X_k)\right) \xrightarrow{p} 0, \qquad \text{that is}$$

$$\lim_{n\to\infty} P\left(\frac{1}{n}\sum_{k=1}^{n}\left(X_k - E(X_k)\right) < \varepsilon\right) = 1, \qquad \text{for all } \varepsilon > 0. \tag{2.32}$$

In the following we give some theorems regarding the WLLN.

Theorem 2.17 (WLLN, Markov) *Let $(X_n)_{n\in\mathbb{N}}$ be a sequence of random variables satisfying the following condition (known as* Markov's condition*):*

$$\lim_{n\to\infty} \frac{1}{n^2} var\left(X_1 + X_2 + \ldots + X_n\right) = 0.$$

Then $(X_n)_{n\in\mathbb{N}}$ obeys the WLLN.

Proof. For $n \in \mathbb{N}$ we consider the average

$$\overline{X}_n = \frac{1}{n}\left(X_1 + X_2 + \ldots + X_n\right).$$

Then, we have

$$\begin{aligned} P\left(\frac{1}{n}\sum_{k=1}^{n} X_k - E(X_k) \geq \varepsilon\right) &\leq P\left(\left|\frac{1}{n}\sum_{k=1}^{n} X_k - E(X_k)\right| \geq \varepsilon\right) \\ &= P\left(\left|\overline{X}_n - E(\overline{X}_n)\right| \geq \varepsilon\right) \\ &\leq \frac{\text{var}\left(\overline{X}_n\right)}{\varepsilon^2} = \frac{\text{var}(X_1 + \ldots + X_n)}{n^2\varepsilon^2}. \end{aligned}$$

In the last equality we have used the Cebyshev inequality given in Theorem 2.16, for $Y = \overline{X}_n$. If we make $n \to \infty$ one obtains

$$\lim_{n\to\infty} P\left(\frac{1}{n}\sum_{k=1}^{n} X_k - E(X_k) \geq \varepsilon\right) = 0, \text{ for all } \varepsilon > 0,$$

which is equivalent to (2.32), whence the sequence $(X_n)_{n\in\mathbb{N}}$ obeys the WLLN. ■

Theorem 2.18 (WLLN, Chebyshev) *Let $(X_n)_{n\in\mathbb{N}}$ be a sequence of pairwise independent random variables for which there exists the real constant $0 < M < \infty$ such that*

$$var(X_n) \leq M, \text{ for all } n \in \mathbb{N}.$$

Then the sequence $(X_n)_{n\in\mathbb{N}}$ obeys the WLLN.

Proof. The conclusion follows immediately from the fact that

$$\frac{1}{n^2}\text{var}(X_1 + \ldots + X_n) = \frac{1}{n^2}\left(\text{var}(X_1) + \ldots + \text{var}(X_n)\right) \leq \frac{M}{n}$$

and from the previous theorem. ■

Theorem 2.19 (WLLN, Poisson) *If in a sequence of independent trials of an experiment the probability that an event A occurs in each trial is p, then for all $\varepsilon > 0$ we have*

$$\lim_{n\to\infty} P\left(\left|\frac{m_n}{n} - p\right| < \varepsilon\right) = 1,$$

where m_n is the number of occurrences of A in n trials.

Poisson's Theorem states that the sequence of relative frequencies m_n/n of the occurrence of an event tends in probability to the probability of the event.

Proof. For an experiment we consider the random variable

$$X : \begin{pmatrix} 1 & 0 \\ p & q \end{pmatrix}, \text{ with } q = 1 - p,$$

which takes the value 1 if event A occurs and 0 otherwise. A simple calculation shows that $E(X) = p$ and $\text{var}(X) = pq$. Then, to each trial we consider a random variable $X_n = X$ and we see that

$$\overline{X}_n = \frac{1}{n}\left(X_1 + X_2 + \ldots + X_n\right) = \frac{m_n}{n} \text{ and } E(\overline{X}_n) = \frac{np}{n} = p.$$

The sequence of r.v. $(X_n)_{n\in\mathbb{N}}$ satisfies the Markov condition:

$$\frac{1}{n^2}\sum_{i=1}^{n}\text{var}(X_i) = \frac{pq}{n} \longrightarrow 0, \ n \to \infty,$$

therefore, by applying Markov's Theorem the conclusion follows immediately. ■

Definition 2.27 (Strong Law of Large Numbers) *A sequence* $(X_n)_{n\in\mathbb{N}}$ *of random variables with* $E(X_n) < \infty$ *for all* $n \in \mathbb{N}$ ***obeys the SLLN*** *if*

$$\frac{1}{n}\sum_{k=1}^{n}(X_k - E(X_k)) \xrightarrow{a.s.} 0, \qquad \textit{that is}$$

$$P\left(\lim_{n\to\infty}\frac{1}{n}\sum_{k=1}^{n}(X_k - E(X_k)) = 0\right) = 1. \tag{2.33}$$

We will give without proof two results regarding the SLLN.

Theorem 2.20 (Kolmogorov) *If* $(X_n)_{n\in\mathbb{N}}$ *is a sequence of independent random variables satisfying*

$$\lim_{n\to\infty}\sum_{k=1}^{n}\frac{1}{k^2}var(X_k) < \infty,$$

then

$$\frac{1}{n}\sum_{k=1}^{n}(X_k - E(X_k)) \xrightarrow{a.s.} 0,$$

i.e. the sequence $(X_n)_{n\in\mathbb{N}}$ *obeys the SLLN.*

Theorem 2.21 (Kolmogorov) *If* $(X_n)_{n\in\mathbb{N}}$ *is a sequence of independent identically distributed random variables such that* $E(X_n) = \mu < \infty$*. Then*

$$\frac{1}{n}\sum_{k=1}^{n}X_k \xrightarrow{a.s.} \mu,$$

i.e. the sequence $(X_n)_{n\in\mathbb{N}}$ *obeys the SLLN.*

2.14 Solved problems

Problem 2.1 (Birthday problem) *How many persons do we need to have in the room to make it a favorable bet (probability of success grater than* 0.5*) that two persons in the room will have the same birthday?*

In probability theory, this is also called *birthday paradox*. There are 365 possible birthdays, so it is tempting to guess that we would need about 1/2 of this number, 183. The result will be somehow surprisingly, in fact the required number being 23. For 60 or more person, the probability is greater than 99%, although it cannot actually be 100% unless there are 366 persons.

Solution. Let p_n denote the probability that, in a room with n people, there is no duplication of birthday. We suppose that for each person, there are 365 possible birthdays, all equally probable.[11] Then

$$p_n = \frac{365 \cdot 364 \cdot \ldots \cdot (365 - n + 1)}{365^n}.$$

[11]Even if we have supposed that the 365 birthdays are equally likely to fall in any particular day, statistical evidence shows this is not true.

For our problem, we need the minimum value of n for which $1 - p_n > 0.5$ (or, equivalent $p_n < 0.5$). Evaluating $q_n = 1 - p_n$ for different values of n, we find

$n = 15$	$q_n = 25.29\%$	$n = 30$	$q_n = 70.63\%$
$n = 20$	$q_n = 41.14\%$	$n = 40$	$q_n = 89.12\%$
$n = 21$	$q_n = 44.36\%$	$n = 50$	$q_n = 97.03\%$
$n = 22$	$q_n = 47.56\%$	$n = 57$	$q_n = 99.01\%$
$\mathbf{n = 23}$	$\mathbf{q_n = 50.72\%}$	$n = 100$	$q_n = 99.99996\%$

Problem 2.2 (Same birthday as you) *What is the probability that in a room with n people someone in the room has the same birthday as you?*

Solution. Let q_n denote the probability of the contrary event A_n, that nobody in the room has the same birthday as you. Then A_n can be written as a product of n independent events,

$$A_n = B_1 \cap B_2 \cap \cdots \cap B_n,$$

where B_k $(1 \leq k \leq n)$ is the event "*person k does not have the same birthday with you*". The probability of each B_k is

$$P(B_k) = \frac{364}{365},$$

since there are 364 favorable cases (all the days, except yours) and 365 possible cases. Therefore, the required probability will be

$$s_n = 1 - q_n = 1 - \left(\frac{364}{365}\right)^n.$$

A calculation show that, for $n = 23$ persons, the probability is $s_{23} = 6.1\%$, and s_n starts to be greater than 50% for $n \geq 253$, which is significantly greater than $365/2 = 185.5$.

Problem 2.3 (The problem of points) *This is an old and classical problem in probability theory.*

Consider two teams, A *and* B*, which are supposed to be evenly matched. They are competing for a prize, received by the first team which wins n games. After* A *has won i games and* B *has won j games $(i, j < n)$, the competition is interrupted unexpectedly and one has to decide how to divide up the prize money.*

Solution. This problem received a lot of interest early in the history of probability. The following list give some suggestions about the proportion that A should receive.

$$\begin{aligned} \text{Pacioli (1494)} &: i/(i+j), \\ \text{Tartaglia (1556)} &: (n+i-j)/2n, \\ \text{Forestani (1603)} &: (2n-1+i-j)/(4n-2). \end{aligned}$$

The problem was solved by Fermat and Pascal in 1654 and their exchange of letters in which they established the solution can be seen in [5]. The idea was that each team should receive a fraction of the prize equal to the probability that it will win the remaining match. So, the number of games

A needs to win is $k_1 = n - i$ and the number of games B needs to win is $k_2 = n - j$. The maximum number of games to be played in the remaining match is $k_1 + k_2 - 1$.

For winning the contest in exactly m games, A must win the last game and $k_1 - 1$ of the first $m - 1$ games. According to the binomial scheme (p. 38), the probability of winning $k_1 - 1$ of $m - 1$ games is

$$\binom{m-1}{k_1-1}\left(\frac{1}{2}\right)^{m-1},$$

therefore the probability that A wins in m games will be $2^{-m}\binom{m-1}{k_1-1}$.

The possible values for m are $k_1, k_1 + 1, \ldots, k_1 + k_2 - 1$. In conclusion, the probability that A wins is

$$\sum_{m=k_1}^{k_1+k_2-1} \frac{1}{2^m}\binom{m-1}{k_1-1}.$$

Analogously, the probability that B wins is

$$\sum_{m=k_2}^{k_1+k_2-1} \frac{1}{2^m}\binom{m-1}{k_2-1}.$$

Problem 2.4 (Capture-recapture experiments) *A biologist goes to a pond and captures $K = 60$ fishes, marks each with a dot of paint and then release them. After a while he goes back and captures another sample of $n = 50$, finding $k = 12$ marked and $n - k = 38$ unmarked fishes. What is his best guess about the size of the population of fishes in the pond? (We suppose that no fishes enter or leave the population between the two visits).*

Solution. Let N denote the number of fishes in the pond. The probability of getting k marked and $n - k$ unmarked in a sample of n is (using the hypergeometric scheme, p. 40)

$$p_N = \frac{\binom{K}{k}\binom{N-K}{n-k}}{\binom{N}{n}}.$$

In order to estimate the population, we chose N to maximize p_N. A simple calculation shows that

$$p_N = p_{N-1} \cdot \frac{N-K}{N-K-(n-k)} \cdot \frac{N-n}{N},$$

so the inequality

$$\frac{p_N}{p_{N-1}} \geq 1$$

will be equivalent to

$$N \leq \frac{Kn}{k}.$$

In conclusion, for

$$N = \left[\frac{Kn}{k}\right]$$

the probability p_N will be maximized.

Let us mention that, in the case when $N = \frac{Kn}{k}$, we have the equality

$$\frac{K}{N} = \frac{k}{n},$$

which means that the proportion of marked fishes in the population coincides with the proportion of marked fishes in the sample.

In our example $N = \frac{60 \cdot 50}{12} = 250$, so the probability p_N is maximized for $N = 250$.

Problem 2.5 *5% of men and 0.25% of women are colorblind. What is the probability a colorblind person is a man?*

Solution. Consider the experiment of testing a person at random and take the events

$$\begin{aligned} A &: \text{“the tested person is colorblind”}, \\ B_1 &: \text{“the tested person is a woman”}, \\ B_2 &: \text{“the tested person is a man”}. \end{aligned}$$

Then $S = \{B_1, B_2\}$ is a (CSE) since $B_1 \cap B_2 = \emptyset$ and $B_1 \cup B_2 = E$, E being the sure event “a person is tested”. So, A takes place together with one (and only one) of the events of S. According to the formula on total probability we have

$$P(A) = P(B_1) \cdot P(A|B_1) + P(B_2) \cdot P(A|B_2).$$

The probabilities involved here are

$$P(B_1) = \frac{1}{2}, \quad P(A|B_1) = \frac{5}{100}, \quad P(B_2) = \frac{1}{2} \quad P(A|B_2) = \frac{0.25}{100},$$

therefore $P(A) = 2.62\%$.

Suppose now that the experiment of testing was realized and the event A took place, meaning that the tested person is colorblind. We are asked about the probability $P(B_2|A)$, that is, the probability that the person is a man, knowing already that is colorblind. This can be also interpreted as the probability that the occurrence of A is due to the event B_2 in the (CSE) S. To calculate the required probability we use Bayes' formula

$$P(B_2|A) = \frac{P(B_1) \cdot P(A|B_1)}{P(A)} = 0.9523\ldots,$$

thus the probability that a colorblind person is a man is 95.23%.

Problem 2.6 (Genetics) *A woman has a brother with hemophilia but two parents who do not have the disease. If she has two sons without the disease, what is the probability she is a carrier?*

Solution. It is known that hemophilia is caused by a recessive gene h on the X chromosome and that a woman has the last pair of chromosome XX, while a man has XY. A baby will take one chromosome from mother and one from father.

Since the brother is ill, he must be X^hY (where X^h is from mother and Y from father). From the fact that the mother is healthy, we deduce that the mother is X^hX^H. This is because the healthy gene H is dominant over

the recessive h and then, since the mother is a carrier, she has an ill gene h. The father must therefore be X^HY, because the other possibility X^hY would mean that he were ill.

In conclusion, the parents are X^hX^H and X^HY.

In this situation, their children might be:

$$X^hX^H \quad X^HX^H \quad X^hY \quad X^HY \quad .$$

So, for a daughter, the chances of being a carrier are 50%. If she is a carrier, then the chance that her sons will have the disease is 50% (since the sons can be either X^hY or X^HY).

So, let us consider the following events:

$$\begin{aligned} B &: \text{“ she is a carrier”} \\ A &: \text{“ she has two healthy sons”} \end{aligned}$$

We are asked about the probability $P(B|A)$, which can be calculated with Bayes' formula. First we use the FTP formula

$$P(A) = P(A|B)P(B) + P(A|\overline{B})P(\overline{B}) = \frac{1}{4}\cdot\frac{1}{2} + 1\cdot\frac{1}{2} = \frac{5}{8}.$$

The probability $P(A|B)$ was calculated in the following way: she is X^hX^H, therefore a son can be either X^hY (in which case he is ill), or X^hX^H (in which case he is healthy). So the probability that one son is healthy is $1/2$.

Now, Bayes' formula gives

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)} = \frac{\frac{1}{8}}{\frac{5}{8}} = \frac{1}{5}.$$

Problem 2.7 *If one plays a roulette and bet €1 on black, then one wins €1 with a probability $\frac{18}{37}$ and loses €1 with a probability $\frac{19}{37}$.*

a) How much is the expected value of winning?

b) If the roulette is an American one, where the probability of winning is $\frac{18}{38}$, while the probability of loosing is $\frac{20}{38}$, how much is in this case the expected value of winning?

c) A man plays American roulette and bets \$1 on black 19 times. What are his expected winnings?

Solution.

a) The r.v. representing the winning is

$$X : \begin{pmatrix} 1 & -1 \\ \frac{18}{37} & \frac{19}{37} \end{pmatrix}.$$

Its expected value will be

$$E(X) = 1\cdot\frac{18}{37} + (-1)\cdot\frac{19}{37} = -0.0270.$$

It means that one loses about 2.70 cents per play.

b) In the second case, the r.v. representing the winning is

$$Y : \begin{pmatrix} 1 & -1 \\ \frac{18}{38} & \frac{20}{38} \end{pmatrix}$$

and its expected value will be

$$E(Y) = 1 \cdot \frac{18}{38} + (-1) \cdot \frac{20}{38} = -0.0526\,\$.$$

c) For $i \in \mathbb{N}_{19}$, let Y_i be the r.v. representing the winning at the i-th game. Then $E(Y_i) = E(Y) = -\frac{2}{38}$, whence the required expectation will be

$$E(Y_1 + \ldots + Y_{19}) = E(Y_1) + \ldots + E(Y_{19}) = 19 \cdot \left(\frac{-2}{38}\right) = -1\,\$.$$

Problem 2.8 *A man never buy bus tickets. He assumes that there is a probability 0.05 that he will be caught. The first offense costs nothing, the second costs €20, and the subsequent offenses cost €40 each. Under these assumptions, how does the expected cost of traveling 100 times without buying tickets compare with the cost of paying the ticket each trip, if the ticket for one trip costs €2?*

Solution. Let $p = 0.05$ denote the probability of being caught once. The r.v. representing the sum he owes in the case when he never pays the ticket

$$X : \begin{pmatrix} 0 & 20 & 20 + 40 \cdot 1 & \ldots & 20 + 40\,k & \ldots & 20 + 40 \cdot 98 \\ p_0 + p_1 & p_2 & p_3 & \cdots & p_{k+2} & \cdots & p_{100} \end{pmatrix},$$

where $p_k = \binom{100}{k} p^k (1-p)^{100-k}$. Then

$$E(X) = \sum_{k=0}^{98} (20 + 40k) \binom{100}{k+2} p^{k+2} (1-p)^{98-k} = 140.98€,$$

which is smaller than the sum spent in the case he pays each trip (€200). To calculate the sum, we used similar arguments with those used in calculating the expectation of the binomial distribution.

Problem 2.9 *Suppose that, when somebody travels with the bus, the cost of one trip is €2 and there is a probability p of being controlled. Each time when a person is caught without ticket, the penalty is €40. Which is the maximum value of p for which the expected cost of traveling 100 times without buying a ticket is smaller than the €200 (the sum owed in the case one pays each ticket)?*

Solution. The r.v. representing the sum owed in the case when one never pays the ticket is

$$X : \begin{pmatrix} 40k \\ \binom{100}{k} p^k (1-p)^{100-k} \end{pmatrix}, \quad k = 0, 1, \ldots, 100,$$

therefore if we consider the r.v. $\widetilde{X} = \frac{1}{40} X$, then $\widetilde{X}$ will have a binomial distribution. According to Theorem 2.9, its expected value will be $E(X) = 100p$, whence $E(X) = 4000p$. The inequality $4000p \leq 200$ implies

$$p_{\max} = \frac{200}{4000} = 0.05\,.$$

Chapter 3

Graph Theory

3.1 Introduction to graphs

3.1.1 Graphs

In real world, many situations can be described by means of a diagram consisting of set of points together with lines joining certain pairs of these points. For examples, the points could represent communication centers with lines representing communications links, or the points could be people with lines joining pairs of friends. In such diagrams one is mainly interested in whether or not two given points are joined by a line. A mathematical representation of situations of this type leads to the concept of graph.

Definition 3.1 *A **graph** G consists of a (finite) set V, whose members are called vertices, and a set E of unordered pairs with components in V, whose members are called **edges**. We usually write $G = (V, E)$ and denote the edges as $\{x, y\}$ or simply xy, where $x, y \in V$.*

Remark 3.1 *In a graph, the edge $\{x, y\}$ coincides with the edge $\{y, x\}$ and we also allow as edge a pair $\{x, x\}$.*

Example 3.1 *$G = (V, E)$, with*

$$V = \{a, b, c, d, f\},\ E = \{\{a, b\}, \{a, d\}, \{b, f\}, \{c, d\}, \{d, f\}\}.$$

Definition 3.2 *Let $G = (V, E)$ be a graph.*

- *We say that two vertices $x, y \in V$ are **adjacent** (or **neighbors**) whenever xy is an edge.*
- *An edge has a set of one or two distinct vertices called **endpoints**.*
- *If a vertex v is an endpoint of the edge e, then v is said to be **incident** on e and e is said to be **incident** on v.*
- *A **loop** (or **self-loop**) at a vertex v is an edge that joins the vertex v with itself. A **proper edge** is an edge that joins two different endpoints.*
- *A graph with no loops is called **simple graph**.*
- *The graph $G' = (V', E')$ is called **subgraph** of G if $V' \subseteq V$ and $E' \subseteq E$.*
- *If $V_1 \subseteq V$, the graph formed by the vertices of V_1 and all the edges in E between these vertices is called the subgraph **induced on the vertex subset** V_1.*

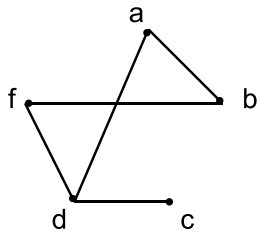


Figure 3.1: The pictorial representation of the graph in Example 3.1.

*A **spanning subgraph** of the graph G is a subgraph that contains all the vertices of G.*

*The graph G is called **complete** if $xy \in E$ for all $x, y \in V$, $x \neq y$.*

Remark 3.2 *Let $G = (V, E)$ be a graph and let*

$$\mathcal{G} = \{H,\ H \text{ subgraph of } G\}$$

be the set of all subgraphs of the graph G. On $\mathcal{G}$ one can introduce the following relation:

$$H_1 \preceq H_2 \iff H_1 \text{ subgraph of } H_2. \tag{3.1}$$

*One can easily prove that $(\mathcal{G}, \preceq)$ is a **poset**. For the definition of the poset and more details about order relations, see Appendix 3.9.4, p. 135.*

3.1.2 Degree of a vertex. Euler's theorem for graphs

Definition 3.3 *The **degree** of a vertex v in a* simple *graph $G = (V, E)$ is the number of edges of G incident on v. It will be denoted $\delta(v)$ or $deg(v)$.*

In a graph with loops, it is considered that each loop at v brings a contribution 2 to $\delta(v)$.

For the graph in Example 3.1, the degrees are

$$\delta(a) = 2,\ \delta(b) = 2,\ \delta(c) = 1,\ \delta(d) = 3,\ \delta(f) = 2.$$

The relation between the sum of degrees and the number of edges is given by Euler's[1] theorem.

Theorem 3.1 (Euler) *The sum of the values of $\delta(v)$, taken over all the vertices of a graph $G = (V, E)$, is equal to twice the number of edges:*

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

Proof. An edge $e = \{x, y\}$ brings a contribution 2 to the total degree: 1 to $\delta(x)$ and 1 to $\delta(y)$. ■

An immediate consequence of Euler's theorem is the following corollary:

Corollary 3.2 *In a graph, there is an even number of vertices having odd degree.*

This result is sometimes known as the **handshaking lemma**, in view of the following interpretation: *Given any set of people, the number of people who have shaken hands with an odd number of other members of the set is even.*

[1] Leonhard Euler (1707-1783), Swiss mathematician and physicist.

3.1.3 Walks, trails, paths, cycles in graphs

Frequently, we use graphs as models of practical situations involving routes. In these situations, the vertices represent towns/junctions and each edge represents a road or some other form of communication link. Therefore one needs to introduce some additional notions, in order to handle these practical situations.

Definition 3.4 [2]*Let $G = (V, E)$ be a graph.*

*A **walk** of **length** k ($k \in \mathbb{N}$) in G is a sequence*

$$\omega = \langle v_0, e_1, v_1, e_2, v_2, \ldots, e_k, v_k \rangle,$$

whose terms are alternately vertices and edges, such that, for $1 \leq i \leq k$, the ends of e_i are v_{i-1} and v_i. We say that ω is a walk from v_0 to v_k or a (v_0, v_k)-walk. The walk ω is often represented simply by its vertex sequence $\langle v_0, v_1, \ldots, v_k \rangle$ or by its edge sequence $\langle e_1, e_2, \ldots, e_k \rangle$.

*A walk ω_1 in G is called a **sub-walk** of the walk ω if ω_1 is a sequence $\omega_1 = \langle v_i, e_{i+1}, v_{i+1}, \ldots, e_j, v_j \rangle$ of consecutive terms of ω.*

*A **trail** in G is a walk in which all the edges (but not necessary the vertices) are distinct.*

*A **path** in G is a walk in which all vertices (and implicitly all edges) are distinct, except the first and the last, which can sometimes coincide.*

We will also use the word "path" to denote a graph or subgraph whose vertices and edges belong to path.

*A walk/trail/path in G is called **closed** if the first and the last vertex coincides. A walk/trail/path which is not closed is called **open**.*

*A **cycle** in G is a closed path in G.*

*The **distance** between two vertices $x, y \in V$, denoted $d(x, y)$ or $dist(x, y)$, is the minimum of the lengths of all (x, y) - paths in G:*

$$d(x, y) = \min\{length(\omega), \text{ with } \omega \text{ an } (x, y) \text{ - path}\}.$$

Definition 3.5 *Let $\omega_1 = \langle x_1, x_2, \ldots, x_k \rangle$, $\omega_2 = \langle y_1, y_2, \ldots, y_p \rangle$ be two walks of a graph $G = (V, E)$, such that $x_k = y_1$. The walk*

$$\omega_1 \cup \omega_2 = \langle x_1, x_2, \ldots, x_k, y_2, \ldots, y_p \rangle$$

*is called the **union** of the walks ω_1 and ω_2.*

*If a walk ω contains a closed sub-walk ω', then ω can be **reduced** to a walk denoted $\omega - \omega'$, by removing from ω all vertices and edges of ω', except the first vertex.*

In the following we give some properties of walks in a graph.

Proposition 3.3 *Every walk ω between two distinct vertices x, y is either a path or contains a closed sub-walk of ω.*

[2]The corresponding Romanian terms are: walk = *lanţ*, closed walk = *lanţ închis*, trail = *lanţ simplu*, path = *lanţ elementar*, cycle = *ciclu elementar*, closed trail = *ciclu*

Proof. If ω is not a path, then a subsequence of ω between repeated vertices defines a closed sub-walk of ω. ■

A simple consequence is the following:

Proposition 3.4 *If there exists a (x, y)-walk in G, then there exists also a (x, y)-path in G.*

Proof. If the walk ω is not a path, it contains a closed sub-walk ω_1. The walk $\omega - \omega_1$ is either a path, or contains a sub-walk ω_2. In the second case, we consider $(\omega - \omega_1) - \omega_2$ and repeat the procedure (a finite number of times), until we obtain a path. ■

Proposition 3.5 *Every closed trail T contains a sub-walk that is a cycle.*

Proof. Let T^* be a closed walk of minimum length. Then T^* has no proper closed sub-walks, hence its only repeated vertices are its first and last. In conclusion, T^* is a cycle. ■

Remark 3.3 *This property is not true if T is merely a closed walk.*

For the following result, we need to introduce some terminology.

Definition 3.6 *A collection of edge-disjoint cycles $C_1, C_2, \ldots, C_n$ is called a **decomposition** of the closed trail τ, if these cycles are sub-walks of τ and if the union of their edge-sets coincides with the edge-set of τ (with other words $\{C_1, C_2, \ldots, C_n\}$ forms a partition of the edge-set of τ).*

Proposition 3.6 *A closed trail can be decomposed into edge-disjoint cycles.*

Proof. We prove by induction on k (k = the number of edges). For $k = 1$, the trail is a loop. Suppose the property true for all closed trails with $\leq m$ edges and let τ be a closed trail with $m + 1$ edges. According to Proposition 3.5, τ contains a cycle, say C. Then $\tau - C$ is a closed trail having $\leq m$ edges. By the hypothesis of induction, $\tau - C$ can be decomposed into edge-disjoint cycles and therefore τ can be written as a union of these disjoint cycles and C. ■

Definition 3.7 (Operations on graphs) *Let $G = (V, E)$ be a graph.*

- ***Deleting an edge*** *$e \in E$ from the graph G, one obtains a subgraph denoted $G - e$, that contains all the vertices of G and all the edges of G except for e.*
- ***Deleting a vertex*** *$x \in V$ from the graph G, one obtains a subgraph denoted $G - x$, that contains all the vertices of G except x and all the edges of G except those incident on x.*
- ***Adding an edge*** *$e_1 = xy \notin E$ to the graph G, one obtains another graph, denoted $G + e_1$, with vertex set $V \cup \{x\} \cup \{y\}$ and edge set $E \cup \{e_1\}$.*

3.1.4 Connectivity in graphs

Definition 3.8 *Let $G = (V, E)$ be a graph.*

*G is called **connected** if, for every two vertices x, y, there exists an (x, y)-walk.*

*A **connected component** (or simply **component**) of G is a* maximal[3] *connected subgraph S of G, in the sense that S is not a proper subgraph of any connected subgraph of G.*

*An edge $e \in E$ is called a **cut edge** or **bridge** if the removal of e increases the number of components.*

When a small graph is given by a pictorial representation, it is quite easy to precise whether it is connected or not. However, when a graph is given in another way (incidence matrix, for instance), we need some efficient algorithms to decide whether it is connected or not. Such a method is presented in Section 3.2.1, p. 81.

The following theorem states that every non-connected graph G can be decomposed into connected components. Its importance is significant, since many properties of graphs can be establish by considering each component separately. For this reason, theorem about graphs are often proved only for connected graphs.

Theorem 3.7 ***(Decomposition into connected subgraphs)***
For every graph $G = (V, E)$ there exist a (finite) number k of connected subgraphs $G_i = (V_i, E_i)$, $i \in \mathbb{N}_k$, with the following properties:

1. $V_i \cap V_j = \emptyset, \quad E_i \cap E_j = \emptyset, \quad$ *for all* $i, j \in \mathbb{N}_k$, $i \neq j$,
2. $V = \bigcup_{i=1}^{k} V_i, \quad E = \bigcup_{i=1}^{k} E_i.$

Proof. On the set of vertices V we define the following relation:

$$x \sim y \iff x = y \quad \text{or} \quad \text{there exists an } (x, y) \text{ -walk.} \tag{3.2}$$

This relation is an equivalence relation on V, called the *reachability relation*. It induces a partition $\{V_1, V_2, \ldots, V_k\}$ of the vertex set V (see Appendix 3.9.4). For each $i \in \mathbb{N}_k$, we denote by $G_i = (V_i, E_i)$ the graph induced on the vertex subset V_i.

Next we need to prove that $E_s \cap E_t = \emptyset$ and $\bigcup_{i=1}^{k} E_i = E$. Suppose there exist $s \neq t$ such that $E_s \cap E_t \neq \emptyset$ and let $e = xy \in E_s \cap E_t$. Then $x, y \in V_s \cap V_t$, which is a contradiction. It remains to prove the inclusion $E \subseteq \bigcup_{i=1}^{k} E_i$. Let $e = xy \in E$. If $x \in V_s$ and $y \in V_t$ with $s \neq t$, then there exists a (x, y) - walk between a vertex in V_s and a vertex in V_t, which is a contradiction with the fact that V_s and V_t belong to different equivalence classes. In conclusion x, y belong to the same class of equivalence V_j. Thus, $e \in E_j \subseteq \bigcup_{i=1}^{k} E_i$. ■

An immediate consequence of Theorem 3.7 is the following corollary.

[3]For the definition of a maximal element, see Appendix 3.9.4, p. 136. The order relation in this case is the one defined in (3.1).

Corollary 3.8 ***(Decomposition into connected components)*** *The graphs $G_i = (E_i, V_i)$, $1 \leq i \leq k$, induced by the equivalence classes of the reachability relation* (3.2)*, are the* connected components *of the graph G.*

Proof. Suppose that there exists a connected subgraph of G, say $G' = (V', E')$, such that $G_i \preceq G'$. We will prove that $G' = G_i$.

From $G_i \preceq G$ we have $V_i \subseteq V'$ and $E_i \subseteq E'$. Suppose there exists $x \in V' \setminus V_i$. Since $G_i \preceq G'$ and G', G_i are connected, the vertex x is also connected to all vertices in V_i, in contradiction with the construction of V_i. Next, let $e = xy \in E'$. From $V' = V_i$ we have $x, y \in V_i$, and therefore $e \in E_i$, since (V_i, E_i) is the graph induced on the vertex subset V_i. In conclusion, $E' \subseteq E_i$, which completes the proof. ■

This corollary suggests the following alternate definition of component.

Definition 3.9 *A **component** of a graph $G = (V, E)$ is a subgraph induced by an equivalence class of the reachability relation on V.*

3.1.5 Multi-graphs, digraphs and multi-digraphs

In practice there are situations (like, for instance, electrical networks), when one needs to consider more than one edge between two vertices. In this situation one needs to enlarge the notion of graph with a new one that allow more edges joining the same two vertices.

Definition 3.10 ***Multiple edges*** *(also called **parallel edges** or **a multi-edge**), are two or more edges that are incident on the same two vertices.*

*A **multi-graph** is a graph which is permitted to have multiple edges.*

Mathematically, a multi-graph consist of:

· a set V of vertices,

· a set E of unordered pairs with components in V, whose members are called edges,

· a function $\mu : E \to \mathbb{N}$, $\mu(xy) =$ number of edges with the endpoints x, y.

Usually, a multi-graph is denoted $G = (V, E, \mu)$. The terminology introduced in Definitions 3.2, 3.4, is also valid for multi-graphs, with the remark that from the tree notations available for walks (see p. 69), only the one in which both vertices and edges are mentioned is correct. We will allow also loops, and for a multi-graph with no loops we employ the terminology *simple multi-graph.*

The concepts of graph and multi-graph may not be sufficient. For instance, when dealing with problems of traffic flow, it is necessary to know which roads on the network are one-way and in which direction traffic is permitted. Clearly, a graph of the roads network is not of much use in such a situation. What we need is a graph in which each link has a certain orientation – a directed graph.

Definition 3.11 *A **digraph** (or a **directed graph**) D consist of a (finite) set V, whose members are called vertices, and a set A, $A \subseteq V \times V$, whose members are called **arcs**. We usually write $D = (V, A)$ and denote the arcs as (x, y), with $x, y \in V$.*

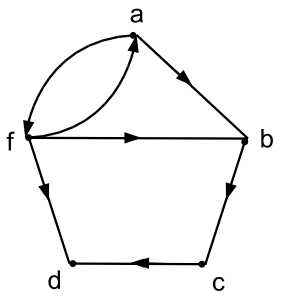


Figure 3.2: The digraph in Example 3.2.

Unlike the edges, the arcs are *ordered* pairs. Thus, an arc (x, y) joins the vertices x and y in a specified direction (*from* x *to* y). For $x \neq y$, the arc (x, y) does not coincide with the arc (y, x).

Example 3.2 *$D = (V, A)$, with*

$$\begin{aligned} V &= \{a, b, c, d, f\}, \\ E &= \{(a, b), (b, c), (c, d), (f, d), (f, b), (f, a), (a, f), (f, b)\}. \end{aligned}$$

Definition 3.12 *Let $D = (V, A)$ be a digraph.*

Two or more arcs joining the same pair of vertices in the same direction are called ***multiple arcs*** *(But two arcs joining the same vertices in opposite directions are* not *multiple arcs* ***!****).*

An arc joining a vertex to itself is a ***loop****.*

A loop-less digraph is called a ***simple*** *digraph.*

Two vertices joined by an arc in either direction are called ***adjacent****.*

The vertices are ***incident*** `to` *and* `from` *the arc joins them.*

An arc is ***incident*** `to` *and* `from` *the vertices it joins.*

A ***tail*** *of an arc is the vertex at which the arc originates.*

A ***head*** *of an arc is the vertex at which the arc terminates.*

A ***sub-digraph*** *of D is a digraph all of whose vertices and arcs are vertices and arcs of D.*

If $V_1 \subset V$, the sub-digraph constructed with the vertices of V_1 and all the arcs in A joining these vertices is called the sub-digraph ***induced*** *by V_1.*

The ***underlying*** *graph of a digraph is the graph obtained by replacing all the arcs of the digraph by undirected edges.*

The ***out-degree*** *of a vertex x is the number of arcs incident* `from` *x. The* ***in-degree*** *of a vertex x is the number of arcs incident* `to` *x. Loops count as one of each.*

Remark 3.4 *Let $D = (V, A)$ be a digraph and let*

$$\mathcal{D} = \{H,\ H \text{ subdigraph of } D\}$$

be the set of all subgraphs of the digraph D. As in the case of the graphs, on $\mathcal{D}$ one can introduce the following relation:

$$H_1 \preceq H_2 \iff H_1 \text{ sub-digraph of } H_2. \tag{3.3}$$

*One can easily prove that $(\mathcal{D}, \preceq)$ is a **poset**.*

Theorem 3.9 (Euler's theorem for digraphs) *The sum of the out-degree and of the in-degree of the vertices of a graph are both equal to the number of arcs.*

Proof. Every arc brings the following contributions:
· one, to the out-degree of the vertex to which it is incident,
· one, to the in-degree of the vertex to which it is incident. ■

3.1.6 Walks, trails, paths, cycles in digraphs

Definition 3.13 [4] *Let $D = (V, A)$ be a digraph.*

*A directed **walk** of length k ($k \in \mathbb{N}$) in D is a finite sequence $\omega = \langle v_0, a_1, v_1, \ldots, a_k, v_k \rangle$ whose terms are alternately vertices and arcs, such that, for $1 \leq i \leq k$, the arc a_i has head v_i and tail v_{i-1}. We say that ω is a directed (v_0, v_k)- walk.*

As for walks in graphs, a walk $\omega = \langle v_0, a_1, v_1, \ldots, a_k, v_k \rangle$ can also be given only by its vertex sequence $\omega = \langle v_0, v_1, \ldots, v_k \rangle$ or by its arc sequence $\omega = \langle a_1, \ldots, a_k \rangle$.

*A directed **trail** in D is a walk in which all the arcs (but not necessary the vertices) are distinct.*

*A directed **path** in D is a walk in which all vertices (and implicitly all arcs) are distinct, except the first and the last, which can sometimes coincide.*

*A directed walk/trail/path in D is called **closed** if the first and the last vertex coincides. A walk/trail/path which is not closed is called **open**.*

*A closed path in D is called **cycle** (sometimes **directed cycle** or **circuit**).*

The terminology introduced in Definitions 3.5 and 3.7 is also available, taking arcs instead of edges.

3.1.7 Connectivity and strong connectivity in digraphs

Regarding the connectivity of a digraph, things are slightly different than for graphs.

Definition 3.14 *A **connected digraph** is one whose underlying graph is a connected graph. A **disconnected digraph** is a digraph which is not connected.*

[4] The corresponding Romanian terms in the case of digraphs are: digraph = *graf orientat*, (but, there also exist the terminology "oriented graph", which means something else and does not translate by *graf orientat* **!**), walk = *drum*, closed walk = *drum închis*, trail = *drum simplu*, path = *drum elementar*, cycle = *ciclu elementar*, closed trail = *ciclu orientat, circuit.*

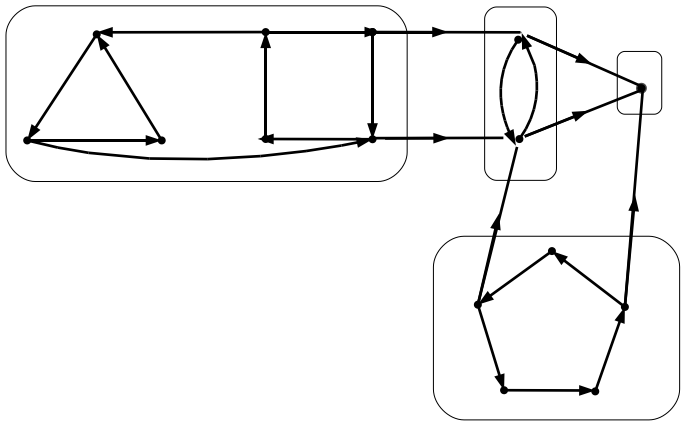


Figure 3.3: A directed graph and its strongly connected components.

A digraph is ***strongly connected*** *if there is a walk joining every pair of vertices.*

A digraph is ***weakly connected*** *if the underlying graph is connected.*

A ***strongly connected component*** *of a digraph D is a* maximal *strongly connected subgraph S of D, in the sense that S is not a proper subgraph of any strongly connected sub-digraph of D.*

Let us mention that strongly connectivity implies connectivity, but the reverse implication is in general not true.

Theorem 3.10 ***(Decomposition of a digraph into strongly connected components)*** *For every digraph $D = (V, A)$, there exists a partition $\{V_1, V_2, \ldots, V_k\}$ of the set V, such that the sub-digraphs $D_i = (V_i, A_i)$ induced by V_i are strongly connected components of D.*

Proof. We define the relation $\sim$ on V, by

$$v_i \sim v_j \iff i = j \quad \text{or} \quad \begin{array}{c}\text{there exist both a walk from} \\ v_i \text{ to } v_j \text{ and from } v_j \text{ to } v_i.\end{array}$$

Then $\sim$ is an equivalence relation on V, called the *mutual-reachability relation.* It induces a partition $\{V_l,\ l = 1, \ldots, k\}$ of V. For $l \in \mathbb{N}_k$, consider the sub-digraphs $D_l = (V_l, A_l)$ induced by V_l. Let $s \in \mathbb{N}_k$. We will prove that D_s is strongly connected.

So let $v_i, v_j \in V_s$, $v_i \neq v_j$. Since v_i, v_j belong to the same equivalence class, we will have $v_i \sim v_j$. Thus, there exist the walks $\omega_1 = \langle v_i, \ldots, v_j \rangle$ and $\omega_2 = \langle v_j, \ldots, v_i \rangle$, with arcs in A. We prove that these walks have all the arcs in A_s. Suppose the contrary, that there exists $x^* \notin V_s$ such that

$$\omega_1 = \langle v_i, \ldots, x^*, \ldots, v_j \rangle.$$

Then $\omega_3 = \langle v_i, \ldots, x^* \rangle$ is a walk from v_i to x^* and $\omega_4 = \langle x^*, \ldots, v_j \rangle \cup \omega_2$ is a walk from x^* to v_i. Therefore, $x^* = v_i$, which is a contradiction with $x^* \notin V_s$. In conclusion both ω_1 and ω_2 are walks with arcs from A_s, so $D_s = (V_s, A_s)$ is a strongly connected subgraph.

The fact that D_s is maximal can be shown by following the same arguments as in the proof of Corollary 3.8. ■

As for graphs, we have the following alternate definition of a strongly connected component.

Definition 3.15 *A **strongly connected component** of a digraph $D = (V, A)$ is a sub-digraph induced by an equivalence class of the mutual-reachability relation on V.*

Remark 3.5 *In the case of digraphs, we do not have necessarily a partition of the set of arcs, that is*

$$\bigcup_{i=1}^{k} A_i \neq A, \quad \textit{in general.}$$

This means that in some digraph there are some arcs which do not belong to any of the sub-digraphs of the partition. These arcs have the following property (see Figure 3.3):

If V_1, V_2 are the vertex sets of two strongly connected components of D, then all the arcs between V_1 and V_2 face the same way (either all are `from` V_1 or all are `to` V_1).

Unlike the digraphs, in graphs we always *have a partition of edges, as we could see in Theorem 3.7.*

Of course, in practice we need sometimes to consider multi-digraphs. The definition will be similar with the one of multi-graphs.

Definition 3.16 *A **multi-digraph** is a digraph which is permitted to have multiple arcs.*

Mathematically, a multi-digraph consist of:

· a set V of vertices,

· a set $A \subseteq V \times V$ of ordered pairs with components in V, whose members are called arcs,

· a function $\mu : A \to \mathbb{N}$, $\mu((x, y))$ = number of arcs from x to y.

Usually, a multi-digraph is denoted $D = (V, A, \mu)$. The terminology introduced in Definitions 3.2 and 3.4 is also valid for multi-digraphs.

3.2 Graphs representation

There are several ways of representing a graph/digraph: pictorial representation, adjacency matrix, incidence matrix, table of incident edges (respectively tables of outgoing arcs and incoming arcs).

3.2.1 Adjacency matrix

Definition 3.17 *The adjacency matrix A_G of a graph (digraph, multi-graph, multi-digraph) G with the vertex set $V = \{v_1, v_2, \ldots, v_n\}$ is a $n \times n$ matrix with entries*

$$(A_G)_{ij} = \begin{cases} \textit{the number of } (v_i, v_j) \textit{ - edges (arcs), if } i \neq j, \\ \textit{the number of loops at } v_i \textit{ if } i = j, \end{cases}$$

for $1 \leq i, j \leq n$.

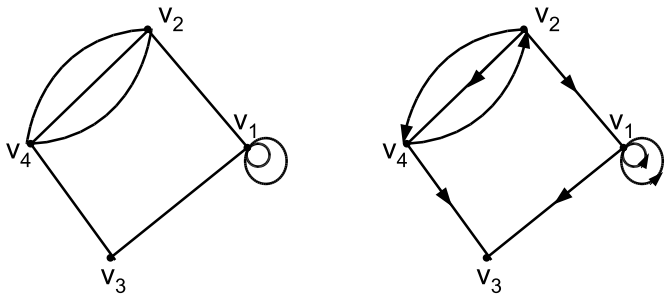


Figure 3.4: The graph G and digraph D in Example 3.3.

Throughout this section we use the term graph (resp. digraph) also for multi-graph (resp. multi-digraph).

Example 3.3 *For the graph G and digraph D in Figure 3.4, the adjacency matrices are, respectively,*

$$A_G = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{pmatrix} 2 & 1 & 1 & 0 \\ 1 & 0 & 0 & 3 \\ 1 & 0 & 0 & 1 \\ 0 & 3 & 1 & 0 \end{pmatrix} \end{array}, \qquad A_D = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{pmatrix} 2 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 \end{pmatrix} \end{array}.$$

Some immediate properties of the adjacency matrix are:

1. The adjacency matrix of a simple graph (digraph) has entries 0 and 1 and 0 on the main diagonal;

2. The adjacency matrix of a graph is symmetric;

3. If $D = (V, A)$ is a digraph with $V = \{v_1, \ldots, v_n\}$ and the entries of the adjacency matrix are a_{ij}, $1 \leq i, j \leq n$, then

$$\sum_{j=1}^{n} a_{ij} = \text{out-degree}(v_i), \quad \sum_{i=1}^{n} a_{ij} = \text{in-degree}(v_j).$$

If we want the number of the walks of length two between two vertices in Example 3.3, one can easily count that there are 2 such (v_1, v_2)-walks, no (v_2, v_4)-walk, 4 such (v_2, v_3)-walks, a.s.o. If we construct a matrix whose entry $\alpha_{i,j}$ $(1 \leq i, j \leq n)$ represents the number of (v_i, v_j)- walks of length two, this matrix equals

$$\begin{pmatrix} 6 & 2 & 2 & 4 \\ 2 & 10 & 4 & 0 \\ 2 & 4 & 2 & 0 \\ 4 & 0 & 0 & 10 \end{pmatrix},$$

representing exactly A_G^2. In general, we have the following property:

Proposition 3.11 *Let G be a graph (digraph) with the vertex set $V = \{v_1, v_2, \ldots, v_n\}$ and let A_G be its adjacency matrix. Then the value of the entry $a_{ij}^{(r)}$ of the r-th power of A_G equals the number of (directed) (v_i, v_j)-walks of length r.*

Proof. By induction on r. For $r = 1$, the property is true. Suppose it is true for r. One has

$$a_{ij}^{(r+1)} = \sum_{k=1}^{n} a_{ik}^{(r)} a_{kj}.$$

In the k-th term of this sum, the factor $a_{ik}^{(r)}$ represents the number of (v_i, v_k)- walks of length r and the factor a_{kj} represents the number of (v_k, v_j)- walks of length 1. Therefore, the product $a_{ik}^{(r)} a_{kj}$ equals the number of (v_i, v_j)- walks of length $r + 1$, with the last but one vertex v_k. Summing up over k, we obtain the total number of (v_i, v_j)- walks of length $r + 1$, and thus the conclusion is proved. ■

Remark 3.6 *The equality $a_{ij}^{(p)} = 0$ does not imply that there is no (v_i, v_j)- walk of length $< p$.*

Proposition 3.11 allows us to decide whether a graph is connected (or whether a digraph is strongly connected) or not. If a graph is connected, there must be paths (of any length) from each vertex to every other vertex. The length of each of these paths must be $\leq n-1$, otherwise a vertex would be visited more than twice. In conclusion, we have the following result.

Proposition 3.12 *If $\mathcal{A}$ is the adjacency matrix of a graph (or directed graph) G with n vertices and*

$$\mathcal{T}_{n-1} = \mathcal{A} + \mathcal{A}^2 + \ldots + \mathcal{A}^{n-1},$$

then G is connected (respectively strongly connected) if and only if each non-diagonal element of $\mathcal{T}_{n-1}$ is > 0.

Proof. Let $\mathcal{A}^k = (a_{ij}^{(k)})_{1 \leq i,j \leq n}$ for $k \in \mathbb{N}_{n-1}$, $\mathcal{T}_{n-1} = (t_{ij})_{1 \leq i,j \leq n}$ and let $i, j \in \mathbb{N}_n$, $i \neq j$.

$\Rightarrow$ Suppose G is strongly connected. Then there exists a (v_i, v_j)- path, of a certain length k, $k \in \mathbb{N}_{n-1}$. Thus, $a_{ij}^{(k)} > 0$ and therefore $t_{ij} > 0$.

$\Leftarrow$ If $t_{ij} > 0$, then there exists $k \in \mathbb{N}_{n-1}$ such that $a_{ij}^{(k)} > 0$. So, there exists a (v_i, v_j)- walk (of length k), whence G is strongly connected.

■

For the digraph in Example 3.3 we have

$$A_D^2 = \begin{pmatrix} 4 & 0 & 2 & 0 \\ 2 & 2 & 2 & 0 \\ 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 2 \end{pmatrix}, \; A_D^3 = \begin{pmatrix} 8 & 0 & 4 & 0 \\ 6 & 0 & 2 & 4 \\ 0 & 0 & 0 & 0 \\ 4 & 4 & 4 & 0 \end{pmatrix}, \; \mathcal{T}_3 = \begin{pmatrix} 4 & 0 & 2 & 0 \\ 2 & 2 & 2 & 0 \\ 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 2 \end{pmatrix},$$

therefore the digraph D is not strongly connected, and this is because there is no path from v_3 to any other vertex.

In the case we are only interested whether there is *at least one* (v_i, v_j)- *walk* rather than wanting to know how many walks, we can use **Boolean matrices**. A Boolean adjacency matrix of a graph (digraph) G with the vertex set $V = \{v_1, \ldots, v_n\}$ has the entries

$$b_{ij} = \begin{cases} 1, & \text{if there is at least one } (v_i, v_j)\text{- edge (arc)}, \\ 0, & \text{otherwise.} \end{cases}$$

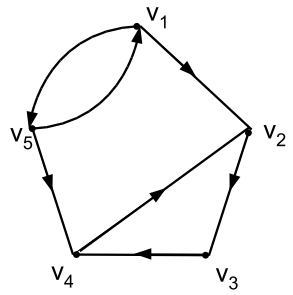


Figure 3.5: The digraph in Example 3.4.

For the graphs in Example 3.3 the Boolean matrices are

$$B_G = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}, \; B_D = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}.$$

The powers and sums of these matrices will be done using *Boolean*[5] *arithmetic*. In this arithmetic, the numbers are 1 and 0 (sometimes used as *true* and *false*), with the sum and product denoted $\oplus$ and $\odot$, respectively, and defined as

$\oplus$	0	1
0	0	1
1	1	1

$\odot$	0	1
0	0	0
1	0	1

The advantage of using Boolean arithmetic when working with a computer is the fact that less memory and less time is needed.

The operations $\oplus$ and $\odot$ on Boolean matrices are defined like in the case of real matrices, with $\oplus$ instead of $+$, $\odot$ instead of $\cdot$, respectively, and have the properties

$$\begin{aligned} &A \oplus A = A, \\ &A \odot I = I \odot A = A, \\ &A \odot (B \oplus C) = A \odot B \oplus A \odot C. \end{aligned}$$

For a Boolean matrix B we will adopt the notations

$$B^{[2]} = B \odot B, \quad B^{[p]} = B^{[p-1]} \odot B, \text{ for } p \geq 3.$$

Example 3.4 *Consider the digraph in Figure 3.5. Its Boolean adjacency matrix will be*

$$B = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix},$$

and then, with the notation $\mathcal{R}_4 = B \oplus B^{[2]} \oplus B^{[3]} \oplus B^{[4]}$,

[5]After the name of George Boole (1815-1864), British mathematician and philosopher, inventor of Boolean algebra, the basis of all modern computer arithmetic.

$$B^{[2]} = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix}, \; B^{[3]} = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix},$$

$$B^{[4]} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}, \; \mathcal{R}_4 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

We have a result similar to the one given in Proposition 3.11, namely:

Proposition 3.13 *Let B be the Boolean matrix of a graph (digraph) with n vertices and $p \in \mathbb{N}$. We denote by $a_{ij}^{[p]}$ the entries of $B^{[p]}$ and by $r_{ij}^{[p]}$ the entries of $\mathcal{R}_p = B \oplus B^{[2]} \oplus \ldots \oplus B^{[p]}$. The following statements are true, for $i \neq j$:*

1. *If $a_{ij}^{[p]} = 1$, then there exists a (v_i, v_j)- walk of length $= p$,*
2. *If $a_{ij}^{[p]} = 0$, then there exists no (v_i, v_j) - walk of length $= p$,*
3. *If $r_{ij}^{[p]} = 1$, then there exists a (v_i, v_j) - walk of length $\leq p$,*
4. *If $r_{ij}^{[p]} = 0$, then there exists no (v_i, v_j) - walk of length $\leq p$.*

Proof. The proof follows the same ideas as the proof of Proposition 3.11. ■

The calculation of the sum $\mathcal{R}_p = B \oplus B^{[2]} \oplus \ldots \oplus B^{[p]}$ can be too expensive for large n. We can deduce the same conclusions as in *3.* and *4.*, by using the matrix $\mathcal{B}_p = I \oplus \mathcal{R}_p$ instead of $\mathcal{R}_p$. This matrix changes only the main diagonal, but it has the following property:

Proposition 3.14 *If B is a Boolean matrix and $C = I \oplus B$, then*

$$C^{[p]} = I \oplus B \oplus B^{[2]} \oplus \ldots \oplus B^{[p]}, \quad \text{for all } p \in \mathbb{N}.$$

Proof. By induction. For $p = 1$, the conclusion is true. Suppose that is true for p. We have

$$\begin{aligned} C^{[p+1]} &= C^{[p]} \odot C = \left(I \oplus B \oplus B^{[2]} \oplus \ldots \oplus B^{[p]}\right) \odot (I \oplus B) \\ &= I \oplus (B \oplus B) \oplus (B^{[2]} \oplus B^{[2]}) \oplus \ldots \oplus (B^{[p]} \oplus B^{[p]}) \oplus B^{[p+1]} \\ &= I \oplus B \oplus B^{[2]} \oplus \ldots \oplus B^{[p+1]}, \end{aligned}$$

and using the principle of induction, the conclusion is proved. ■

An important application of this property is the following: Given two distinct vertices v_i, v_j of a graph (digraph), one can determine the length of a (v_i, v_j)- walk of minimum length, by successively calculating the boolean powers of C and recording the first apparition of 1 on the position (i, j). Once 1 appears on a certain position (i, j) of $C^{[p]}$, it will appear on the same position of all the matrices $C^{[p']}$ with $p' \geq p$.

Actually, there is a faster method, in which one evaluates the powers $C^{[2]}, C^{[4]}, \ldots, C^{[2^s]}, \ldots$ and some intermediate Boolean powers between those powers 2^s and 2^{s+1} where the position (i,j) changes from 0 to 1. For example, suppose that the minimum length of a (v_i, v_j)- walk is 51, but it is not known. First we calculate $C^{[2]}$, $C^{[4]}$, $C^{[8]}$, $C^{[16]}$, $C^{[32]}$, finding the entries

$$c_{ij}^{[2]} = c_{ij}^{[4]} = c_{ij}^{[8]} = c_{ij}^{[16]} = c_{ij}^{[32]} = 0.$$

Then, for $C^{[64]}$ we find $c_{ij}^{[64]} = 1$ (so $s = 5$), therefore there exists a (v_i, v_j)- path of length l, with $32 < l \leq 64$.

Further, we calculate $C^{[32]} \odot C^{[2^k]}$, for $k = 1, 2, 3, 4$ (in general we calculate for $k \leq s-1$, until 1 appears). In our case

$$c_{ij}^{[32+2]} = c_{ij}^{[32+4]} = c_{ij}^{[32+8]} = c_{ij}^{[32+16]} = 0,$$

so $48 < l \leq 64$. Next, we calculate $C^{[48]} \odot C^{[2^k]}$, for $k = 1, 2$ (in general for $k \leq s-2$, until 1 appears) and we find $c_{ij}^{[48+2]} = 0$, $c_{ij}^{[48+4]} = 1$. Thus, $50 < l \leq 52$, and we need one more calculation to find l, namely $c_{ij}^{[51]} = 1$. In total, we performed 6+4+2+1=13 operations, instead of 50 needed to calculate the successive Boolean powers of C until 51.

The Boolean powers of the matrix C are also useful for determining the strongly connected components of a digraph (respectively the components of a graph). More precisely we have the following result.

Theorem 3.15 (Foulkes method for finding the components) *Let G be a simple graph (digraph) with m edges (arcs), B its boolean adjacency matrix, $C = I \oplus B$, and $p \in \mathbb{N}$ such that $2^p \geq m$. Then two distinct vertices v_i, v_j are situated in the same (strongly) connected component of G if and only if the rows i and j in $C^{[2^p]}$ are identical.*

Proof. Let $i \neq j$ and let α_{ij} be the entry (i,j) of the matrix $C^{[2^p]}$. Every path in G has at most m arcs, so, if $\alpha_{ij} = 0$, then there is no (v_i, v_j)- path.

$\Rightarrow$ Suppose v_i, v_j are situated in the same component. Then it is immediate that $\alpha_{ii} = \alpha_{ij} = \alpha_{ji} = \alpha_{jj} = 1$. Let $\nu \neq i, \nu \neq j$.

If $\alpha_{i\nu} = 1$, there is a (v_i, v_ν)- walk in G. Since there exists also a (v_j, v_i)- walk, the union of these two walks will be a (v_j, v_ν)- walk, and therefore $\alpha_{j\nu} = 1$.

If $\alpha_{i\nu} = 0$, suppose $\alpha_{j\nu} = 1$, so there is a (v_j, v_ν)- walk. As before, we deduce that there exists a (v_i, v_ν)- walk, which is the union of a (v_i, v_j)- walk and the above (v_j, v_ν)- walk. Hence, $\alpha_{i\nu} = 1$, which is a contradiction. Therefore, $\alpha_{j\nu} = 0$.

In conclusion, in both cases we obtain $\alpha_{i\nu} = \alpha_{j\nu}$, meaning that the rows i and j of $C^{[2^p]}$ coincide.

$\Leftarrow$ Suppose $\alpha_{i\nu} = \alpha_{j\nu}$ for all ν.

From $\alpha_{ii} = 1$ we deduce $\alpha_{ji} = 1$, whence there exists a (v_j, v_i)- walk. In the same way, from $\alpha_{jj} = 1$ we deduce $\alpha_{ij} = 1$, whence there exists a (v_i, v_j)- walk. In conclusion, v_i and v_j are situated in the same connected component.

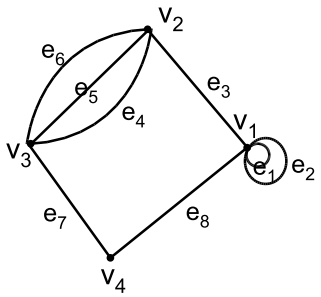


Figure 3.6: The graph in Example 3.5.

■

For the graph in Example 3.4 we have $C^{[4]} = \mathcal{R}_4$, so the graph has two strongly connected components, one containing the vertices v_1, v_5 and the other one containing the vertices v_2, v_3, v_4.

3.2.2 Incidence matrix

Definition 3.18 (Incidence matrix of a graph) *Let $G = (V, E)$ be a graph with $V = \{v_1, v_2, \ldots, v_n\}$ and $E = \{e_1, e_2, \ldots, e_m\}$. The incidence matrix of the graph G is an $n \times m$ matrix I_G with entries*

$$(I_G)_{ij} = \begin{cases} 0, & \text{if } v_i \text{ is not an end-point of } e_j, \\ 1, & \text{if } v_i \text{ is an end-point of } e_j \text{ and } e_j \text{ is a proper edge}, \\ 2, & \text{if } e_j \text{ is a loop at } v_i. \end{cases}$$

for $1 \leq i, j \leq n$.

Example 3.5 *For the graph in Figure 3.6, the incidence matrix is*

$$I_G = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{array} \\ \begin{pmatrix} 2 & 2 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array}.$$

Some properties of the incidence matrix of a graph are:

1. The sum of the entries in any row of an incidence matrix equals the degree of the corresponding vertex, if a loop is considered as having the degree 2;
2. The sum of the entries in any column of an incidence matrix is 2;
3. The incidence matrix of a graph G with $n \geq 2$ vertices has the rank $n - p$, where p is the number of connected components of G.

Definition 3.19 (Incidence matrix of a digraph) *Let $D = (V, A)$ be a digraph with $V = \{v_1, v_2, \ldots, v_n\}$ and $A = \{a_1, a_2, \ldots, a_m\}$. The incidence matrix of the graph D is an $n \times m$ matrix I_D with entries*

$$(I_D)_{ij} = \begin{cases} 0, & \text{if } v_i \text{ is not an end-point of } a_j, \\ 1, & \text{if } v_i \text{ is the head of } a_j, \\ -1, & \text{if } v_i \text{ is the tail of } a_j, \\ 2, & \text{if } a_j \text{ is a loop at } v_i. \end{cases}$$

for $1 \leq i, j \leq n$.

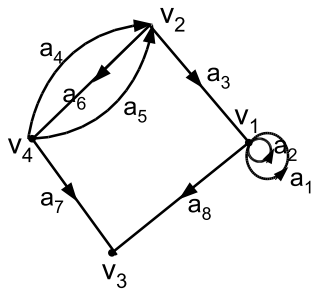


Figure 3.7: The graph in Example 3.6.

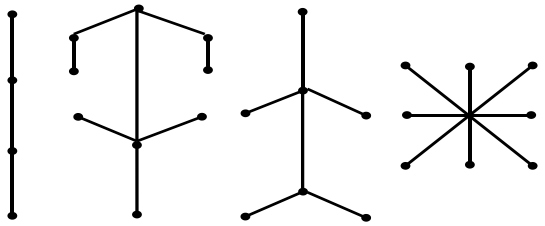


Figure 3.8: Examples of trees.

Example 3.6 *For the graph in Figure 3.7, the incidence matrix is*

$$I_D = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccccccc} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 \end{array} \\ \begin{pmatrix} 2 & 2 & 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 1 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array}.$$

An immediate property is that in a loopless digraph, every column sum is zero.

3.3 Trees, sorting and searching

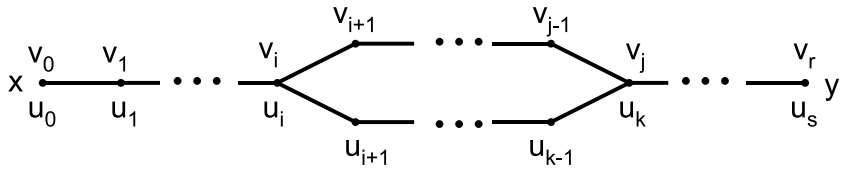
Definition 3.20 *A **tree** is a connected graph that contains no cycles. Any connected subgraph of a given tree is called **subtree**.*

*A **forest** is an acyclic graph (or, equivalently, a collection of trees).*

Examples of trees are given in Figure 3.8. We give some simple properties of trees.

Theorem 3.16 *In a tree T, any two distinct vertices x, y are connected by a unique path.*

Proof. Since T is connected, there is an (x, y) - path, say $\langle v_0, v_1, \ldots, v_r \rangle$, with $v_0 = x, v_r = y$. If there exists a different (x, y) - path, say $\langle u_0, u_1, \ldots, u_s \rangle$, with $u_0 = x, u_s = y$, then let i be the smallest index for which $u_{i+1} \neq v_{i+1}$. Since both paths finish at y, they will meet again, and we define j to be the smallest index such that $j > i$ and $v_j = u_k$ for some k. Then $\langle v_i, v_{i+1}, \ldots, v_j, u_{k-1}, \ldots, u_{i+1}, v_i \rangle$ is a cycle in T, contrary with the hypothesis. ■



Theorem 3.17 *The graph obtained from the tree $T = (V, E)$ by removing any edge has two components, each of which is a tree.*

Proof. Suppose $uv \in E$, $E' = E \setminus \{uv\}$, and let $S = (V, E')$. Define

$$V_1 = \{x \in V : \text{ the unique } (x, v)\text{- path in } T \text{ passes through } u\}.$$

For every $x \in V_1$, the unique (x, v)- path must end with the edge uv, otherwise T would have a cycle $\langle x, \ldots, u, v, \ldots, x\rangle$.

Let $V_2 = V \setminus V_1$. Each vertex in V_1 is joined by a path in S to u and each vertex in V_2 is joined in S to v. In conclusion V_1 and V_2 are the vertex sets of two components of S. Each component is (by definition) connected and it contains no cycle, since there are no cycles in T. Hence, the two components are trees. ■

Theorem 3.18 *If $T = (V, E)$ is a tree, then $|E| = |V| - 1$.*

Proof. By induction on $|V|$.

If $|V| = 1$, the conclusion holds, since the only possible tree has no cycle in this case.

Suppose the conclusion is true for a tree with $\leq k$ vertices. Let T be a tree with $|V| = k + 1$ and let uv be any edge of T. If $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ are the trees obtained by removing uv from T (see Theorem 3.17), we have

$$|V_1| + |V_2| = |V| \quad \text{and} \quad |E_1| + |E_2| = |E|.$$

Thus, we have

$$|E| = |E_1| + |E_2| + 1 = |V_1| - 1 + |V_2| - 1 + 1 = |V| - 1.$$

In the second equality we applying the induction hypothesis for the trees T_1 and T_2. ■

Remark 3.7 *The conclusions of Theorems 3.16–3.18 provide several alternative ways of defining a tree.*

3.3.1 Rooted trees

Frequently, one vertex of the tree is distinguishable in some way. For instance, in a "family tree" which traces the descendants of a king, we might emphasize the special position of the king by putting him at the top of the tree and all the sons "under" their parents.[6] In general we refer to distinguished vertex as the **root** and we assign a **direction** to each edge, namely from parent to child.

[6]Note that, when drawing a genealogical tree, historians usually put the king at the bottom and the descendants above him, so as to reproduce the form of a real tree. The two versions are, of course, fully equivalent.

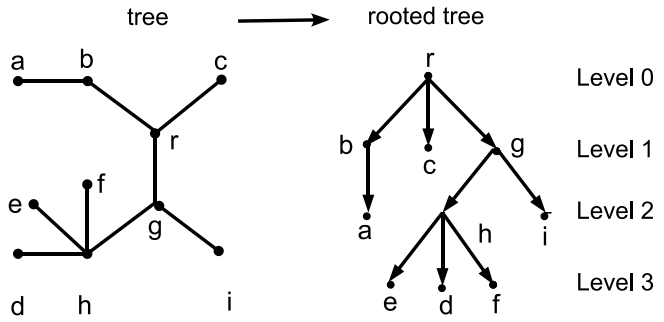


Figure 3.9: Construction of the tree rooted at a given vertex r of a tree.

Definition 3.21 *A **directed tree** is a digraph whose underlying graph is a tree.*

*A **rooted tree** or **arborescence** is a directed tree with a distinguished vertex r, called the **root**, such that for every other vertex v, there is a directed path from r to v.*

In drawing a rooted tree, the arrows are usually omitted from the arcs since the direction of the arc is always "away" from the root. It also follows that the directed path from the root to a given vertex is unique.

Definition 3.22 *In a rooted tree, the **depth** or **level** of a vertex v is the distance from the root, that is, the length of the unique path from the root to v. Thus, the root has depth 0.*

*A vertex in a rooted tree is said to be a **leaf** if it is at level i $(i \geq 0)$ and is not adjacent to any vertices at level $i+1$.*

*A vertex which is not a leaf is an **internal** vertex.*

*The **height** of a rooted tree is the maximum value of k for which level k is not empty.*

*If vertex x immediately precedes vertex y on the path from the root to y, then x is the **parent** or **father** of y and y is the **child** or **son** of x.*

*Let x, y be distinct vertices of a rooted tree. Then y is called **descendant** of x (and x is called an **ancestor** of y) if x is on the unique path from the root to y.*

Each vertex (except the root) has a unique father, but a vertex may have any number of sons (including 0). So, a vertex is a leaf if and only if it has no sons. Also, a vertex is a father if and only if it is an internal vertex. Rooted trees are actually directed graphs, the directions of arcs being from a father to his child. For simplicity, we omit the notations and the terminology for digraphs.

In many applications, it happens that every father has the same number of sons.

Definition 3.23 *If in a rooted tree each father has at most m sons, we speak of an **m-ary** rooted tree. In particular, when $m = 2$, we call it **binary** tree, while when $m = 3$ we call it **ternary**. The m-ary rooted trees in which each father has exactly m sons are called **complete**.*

3.3.2 Decision trees

We start this section with the following theorem:

Theorem 3.19 *The height of an m-ary rooted tree with k leaves is at least $\log_m k$.*

Proof. Let h denote the height of the tree. The required inequality

$$h \geq \log_m k \tag{3.4}$$

is equivalent to $m^h \geq k$, so it would be enough to prove that an $m-$ary rooted tree with height h has at most m^h leaves.

We prove this by induction on h. If $h = 0$, the tree consists of one vertex, which is both leaf and father, so $k = 1$, and thus the inequality is true. Suppose now the inequality true for $0 \leq h \leq h_0$ and let T be an m-ary rooted tree with height $h_0 + 1$. If we denote the root by r, then the tree $T - r$ (see Definition 3.7, p. 70) will consist of at most m trees $T_1, \ldots, T_m$ whose roots are the vertices at level 1 in T. Each T_i are thus rooted trees, of height $\leq h_0$. By hypothesis of induction, each of these T_i trees has at most m^{h_0} leaves. The leaves of T coincide with the leaves of $T_1, \ldots, T_m$, therefore the number of leaves of T will be smaller than $m \cdot m^{h_0} = m^{h_0+1}$.

In conclusion, by the principle of induction, the inequality $k \leq m^h$ is true for all $h \geq 0$. ■

Remark 3.8 *The number $\log_m k$ is in general not an integer. The inequality (3.4) may be rewritten as*

$$h \geq \lceil \log_m k \rceil, \tag{3.5}$$

where for a real number x, $\lceil x \rceil$ denotes the `ceil` *of x (the least integer z such that $z \geq x$).*

Theorem 3.19 is applied in studying the decision trees. A decision tree is an m-ary tree where:

- each internal vertex represents a decision,
- the possible results of that decision are represented by the edges leading to the vertices at the next level,
- the leaves of the tree represent the final outcomes of the procedure.

In the case when the results of the decision are only `true` or `false` statements (0 or 1), we deal with a binary decision tree.

Example 3.7 (The false coin problem) *Suppose we have a genuine coin labeled 0 and n other coins, indistinguishable from 0 by appearance, except for being labeled $1, 2, \ldots, n$. It is suspected that one coin may be false – either too light or too heavy. Show that at least $\lceil log_3(2n + 1) \rceil$ weighings on a balance are needed to decide which coin (if any) is false, and also to decide whether it is light or heavy. Devise a procedure which uses exactly this number of weighings, when $n = 4$.*

Solution. There are $2n + 1$ outcomes (leaves of the decision tree), namely

$$G \quad 1H \quad 1L \quad 2H \quad 2L \quad \ldots \quad nH \quad nL,$$

where G means that all the coins are good, kH means that coin k is heavier and kL means that coin k is lighter.

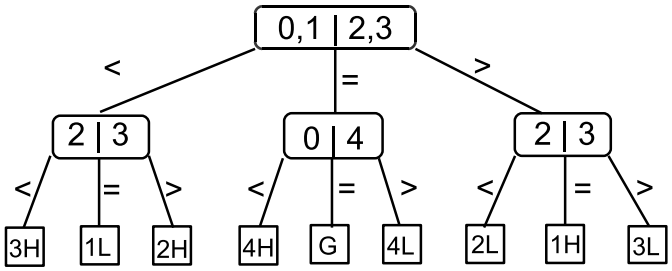


Figure 3.10: The decision tree for Example 3.7.

We need a ternary decision tree, since there are three possible results of each decision, when weighing one set of coins against other:

$<$: left-hand set of coins is lighter
$=$: equally weighted sets of coins
$>$: left-hand set of coins is heavier

According to Theorem 3.19, the height will be $\geq \lceil \log_3(2n+1) \rceil$.

In the case when $n = 4$ the height will be ≥ 2. The procedure which performs two weighing only is depicted in Figure 3.10.

We finish this section by mentioning that the bound $\lceil \log_m k \rceil$ given in (3.5) is a theoretical one in decision problems. In some situations it is possible to design a procedure which allows us to draw a conclusion with $\lceil \log_m k \rceil$ decisions only, but there are also situations when this number cannot be achieved.

3.3.3 Sorting trees

We want to arrange a list $x_1, x_2, \ldots, x_n$ of pairwise distinct numbers in increasing order. The algorithms used for solving this problem involve the comparison of two numbers and (or not) a transfer of data (switching). This kind of procedure can be represented by a decision tree, where a vertex is a comparison of two numbers, x_i and x_j. Since there are two possible results ($x_i < x_j$ and $x_i > x_j$), the decision tree is a binary tree.

One very simple algorithm used in sorting is the so-called *bobble sort algorithm*, where one compares every time two consecutive numbers, switching them if they are not in the correct order.

Algorithm 1 (Bobble sort algorithm)
for $j = 1$ *to* n *do*
  for $i = 1$ *to* $n - j$ *do*
    if $x_i > x_{i+1}$ *switch* x_i, x_{i+1}

A more efficient algorithm, which makes use of binary trees and needs less comparisons, is the *heap-sort algorithm.* In the following we will describe the stages of the heap-sort algorithm.

Constructing the unsorted tree

We start by assigning the members of the list $X = \{x_1, x_2, \ldots, x_n\}$ to the vertices of an unsorted binary tree, as follows:

x_1 root – level 0

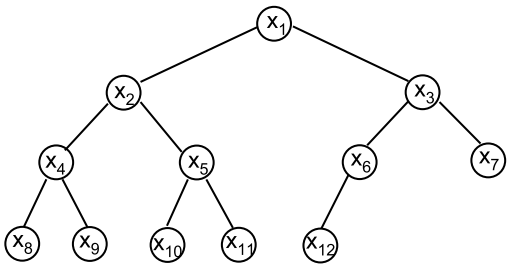


Figure 3.11: The binary tree associated to the list X.

x_2, x_3 – level 1
x_4, x_5, x_6, x_7 – level 2
...

Thus, in this tree x_k is the father of x_{2k} and x_{2k+1}, provided $2k+1 \leq n$. The last level is usually incomplete and if n is even, then $x_{n/2}$ has only one child.

The heap-sort algorithm continues in the following way: first, the unsorted tree is transformed into a special kind of tree known as a heap. Then the heap is transformed into a sorted list $y_1, y_2, \ldots, y_n$.

Definition 3.24 *A **heap** is a binary tree, with a number x_i associated to each vertex and with the property that each father is smaller than its sons $(x_i < x_{2i},\ x_i < x_{2i+1})$.*

Transforming the unsorted tree into a heap

We deal with fathers (internal vertices) in reverse order. Suppose that, when we come to deal with x_k, the two subtrees rooted at x_{2k} and x_{2k+1} have already been made into heaps.

If $x_k < x_{k+1}$ and $x_k < x_{2k+1}$ we don't do anything, since the subtree rooted at x_k is already a heap.

If not, we store x_k temporarily and move the smallest of x_{2k} and x_{2k+1} in the vacant vertex. This create a new vacancy.

If x_k is smaller than the sons of the vacant vertex or if there are no sons, we fill the vacancy with x_k.

If not, we fill the vacancy with the smaller of the sons and continue. Finally we will find a place for x_k when we reach a leaf, if not before.

Let `heap(k,n)` denote the procedure which, given a vertex x_k with the property that the trees rooted at x_{2k} and x_{2k+1} are heaps, transforms the subtree rooted at x_k into a heap. By applying this procedure for $k = \left[\frac{1}{2}n\right], \left[\frac{1}{2}n\right] - 1, \ldots, 2, 1$, the whole tree will be transformed into a heap.

Transforming the heap into a sorted list

In a heap, the root has the smallest value, so it will go into the first place y_1 in the sorted list.

The vacancy at the root is filled by the smallest last value x_n and the old leaf x_n is removed.

We have now a tree with $n-1$ vertices and two subtrees rooted at x_2 and x_3, which are heaps. We apply now the procedure `heap(1,n-1)` to restore the heap property of the whole tree.

In the new tree (which is a heap) the root has the smallest remaining value and this is assigned to y_2.

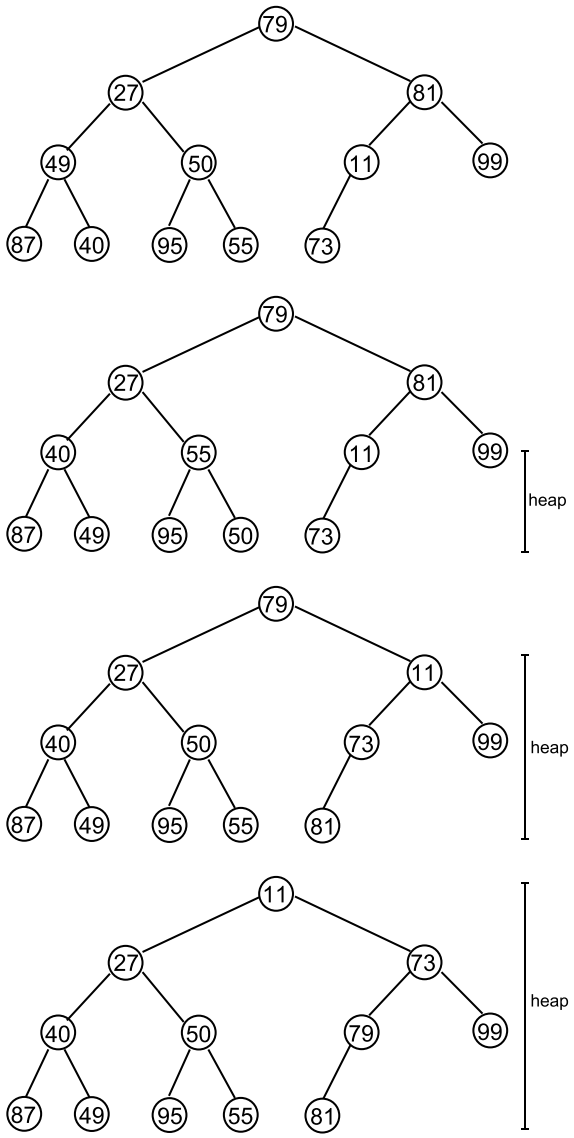


Figure 3.12: Step1: Transform the unsorted list X into a heap.

The last leaf is removed.
We restore now the heap by `heap(1,n-2)`
and so on.

Example 3.8 *Consider the list*

$$X = \{79, 27, 81, 49, 50, 11, 99, 87, 40, 95, 55, 73\}$$

The binary tree associated to this list is depicted in Figure 3.11, the steps of the construction of the first heap are shown in Figure 3.12. The next step is presented in Figure 3.13.

3.4 Binary trees and binary codes

3.4.1 Binary trees

The definition of a binary tree was given on p. 85. In a binary tree, we consider that a child is either a **left-child** or a **right-child**, in other words we order the children.

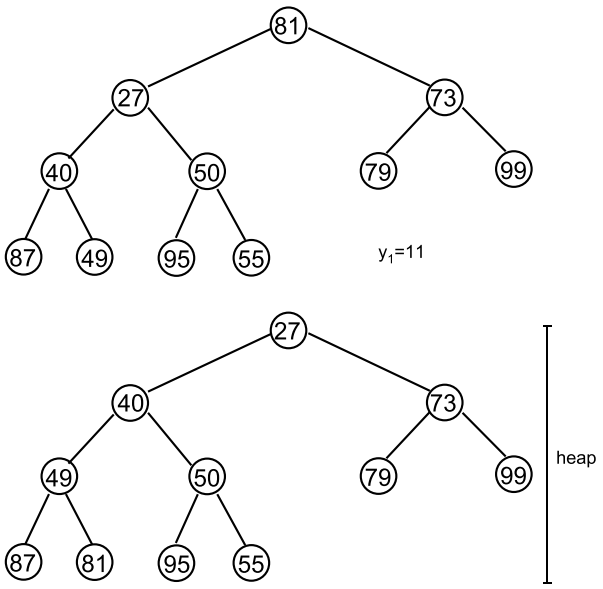


Figure 3.13: Step2: Extract the root $y_1 = 11$, delete the last leaf, and construct again a heap.

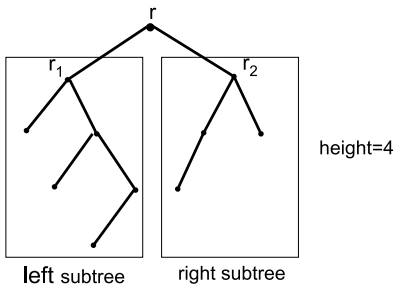


Figure 3.14: A binary tree of height 4, rooted at r, and its left and right subtrees of height 3 and resp. 2, rooted at r_1 resp. r_2.

Definition 3.25 *Let $T = (V, E)$ be a binary tree and $v \in V$. A **left(right) subtree** of the vertex v is the binary subtree induced by the left(right) child of v and all of its descendants.*

So, we can consider a tree as having three components: the root and its left and right subtrees. This allows us to state the following *recursive property of a binary tree:* If T is a binary tree of height h, then its left and right subtrees both have height $\leq h - 1$, with equality for at least one of them. The proof of this property is immediate, by induction on h.

Remark 3.9 *Other immediate properties of a binary tree are the following:*

1. *A complete binary tree of height h has $2^{h+1} - 1$ vertices.*
2. *Every binary tree of height h has at most $2^{h+1} - 1$ vertices.*

The proofs of the above properties follow immediately by induction.

3.4.2 Binary codes

For encoding the information, the computer uses bit strings. The most common encoding is known as ASCII code, in which each of the letters and other symbols has a seven-bit code.

Definition 3.26 *A **binary code** is an assignment of bit-strings to a set of symbols. Each bit-string is referred to as a **codeword**.*

For some applications, it is desirable to allow codewords to vary in length. So let us consider three examples of encodings:

(c_1) $x_1 \to 00$ $x_2 \to 01$ $x_3 \to 10$ $x_4 \to 11$
(c_2) $x_1 \to 0$ $x_2 \to 10$ $x_3 \to 110$ $x_4 \to 111$
(c_3) $x_1 \to 0$ $x_2 \to 1$ $x_3 \to 00$ $x_4 \to 01$

The codewords associated to the string $x_1x_3x_1x_4$ are:

(c_1) 00100011,
(c_2) 01100111,
(c_3) 000001.

The decoding can be done uniquely in the first two encodings, whereas in the third encoding, the bit-string can also be decoded as $x_1x_1x_1x_1x_1x_2$ or as $x_3x_3x_4$. In conclusion, we need to avoid this situation, generated by the fact a codeword is the beginning of another codeword, and thus we do not know where a code starts and where it stops.

To avoid this ambiguity, we introduce the following property:

Definition 3.27 *A **prefix code** is a binary code with the property that no codeword is an initial substring of any other codeword. The number of bits in the prefix code of a symbol is referred to as the **length** of the code.*

One can see that in the third encoding, the code $x_1 \to 0$ is an initial sequence of the codes $x_3 \to 00$ and $x_4 \to 01$, therefore (c_3) is not a prefix code.

The construction of prefix codes can be done by making use of binary trees.

Construction of prefix codes

Let T be a binary tree. If we label each left edge with 0 and each right edge with 1, then to each leaf one assigns a codeword formed by the sequence of edge labels, from the unique path from the root to that leaf. For example,

letter	a	b	c	d	e	f	g
codeword	000	0010	0011	0101	011	100	101

One can see that any path from root to a leaf is *not* an initial sub-path for another path from root to another leaf, so the codes are prefix codes.

Huffman codes

In a prefix code, it is desirable to use fewer bits for encoding the more frequently occurring symbols. A measure of a code's efficiency is the **average weighted length (AWL)** of its codewords, where the length of each codeword is multiplied by the frequency of the symbol it encodes. In the example above, if frequencies are

letter	a	b	c	d	e	f	g
frequency	0.2	0.05	0.1	0.1	0.25	0.15	0.15

, (3.6)

then the average weighted length is

$$3 \cdot 0.2 + 4 \cdot 0.05 + 4 \cdot 0.1 + 4 \cdot 0.1 + 3 \cdot 0.25 + 3 \cdot 0.15 + 3 \cdot 0.15 = 3.25.$$

The coefficients of the frequencies are the depths of the corresponding leafs in the tree.

Definition 3.28 *Let T be a binary tree and $s_1, s_2, \ldots, s_l$ its leaves, such that leaf s_i is assigned a weight w_i. Then the* ***average weighted depth (AWD)*** *of the binary tree T is*

$$\mathcal{W}(T) = \sum_{i=1}^{l} depth\,(s_i) \cdot w_i.$$

An algorithm for constructing a prefix code whose codewords have the smallest possible AWL was given in 1952 by Huffman[7] in his PhD thesis at MIT.

Algorithm 2 ***(Huffman prefix codes)***
Input: *a set $S = \{s_1, \ldots, s_l\}$ of symbols*
a list $W = \{w_1, \ldots, w_l\}$ of weights associated to S ($w_i \to s_i$)
Output: *a binary tree representing a prefix code for the set of symbols S, whose codewords have minimum average weighted length*

- *initialize F as a forest of isolated vertices labeled $s_1, \ldots, s_l$, with respective weights $w_1, \ldots, w_l$*
- *for $i = 1$ to $l - 1$*
  - *choose from forest F two trees, T_0 and T_1 of smallest weights*
  - *create a new binary tree whose root has T_0 and T_1 as its left and right subtrees, respectively*
  - *label the edge to T_0 with a 0 and the edge to T_1 with a 1*
  - *assign to the new tree the weight $w(T_0) + w(T_1)$*
  - *replace trees T_0 and T_1 in forest F by the new tree*
- *return a binary tree associated to the list W*

Definition 3.29 *The binary tree that is the output from Algorithm 2 is called the* ***Huffman tree*** *for the list of symbols. The corresponding prefix code is called the* ***Huffman code****.*

Example 3.9 *Consider the frequencies of the set of symbols $S = \{a, b, c, d, e, f, g\}$ given in formula (3.6).*

letter	*a*	*b*	*c*	*d*	*e*	*f*	*g*
codeword	*00*	*0010*	*0101*	*011*	*10*	*110*	*111*

The average weighted length of a codeword for this prefix code is

$$2 \cdot 0.2 + 4 \cdot 0.05 + 4 \cdot 0.1 + 3 \cdot 0.1 + 2 \cdot 0.25 + 3 \cdot 0.15 + 3 \cdot 0.15 = 2.7.$$

[7]David Albert Huffman (1925-1999), American electrical engineer.

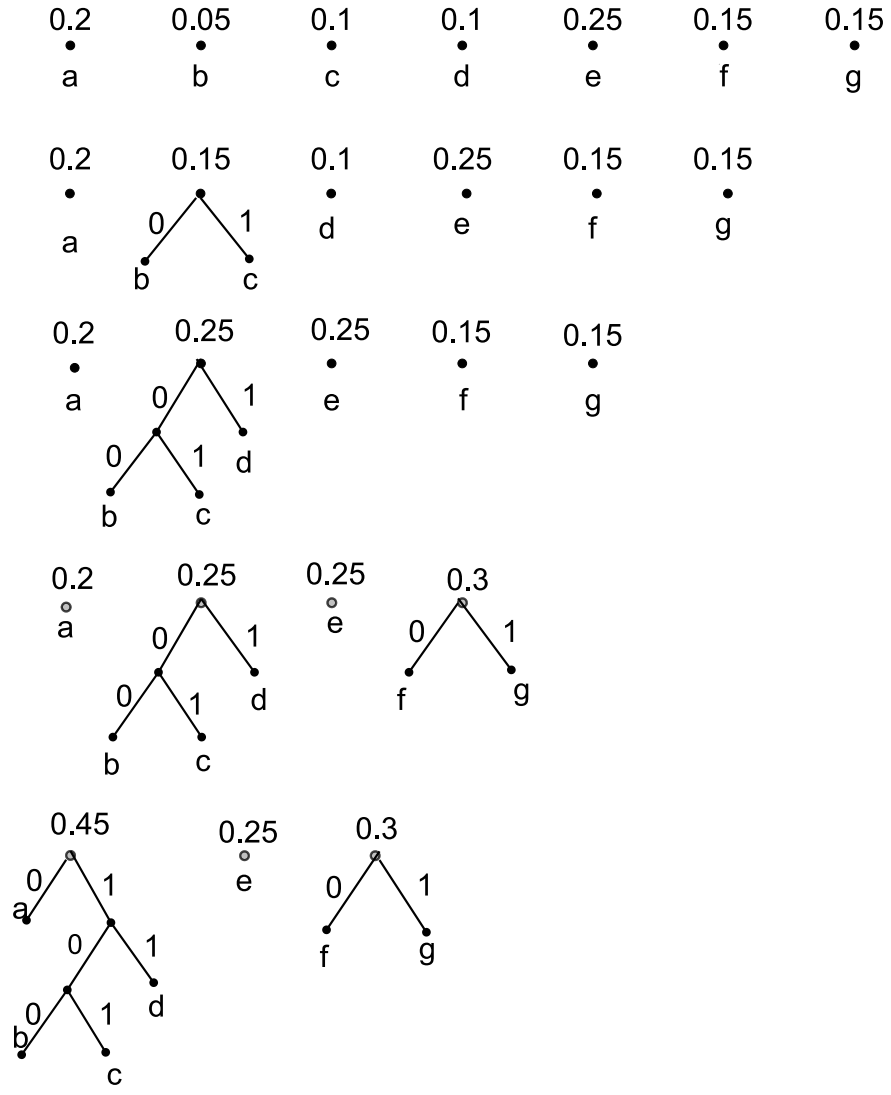


Figure 3.15: Iterations 1-4 in Huffman's algorithm for Example 3.9.

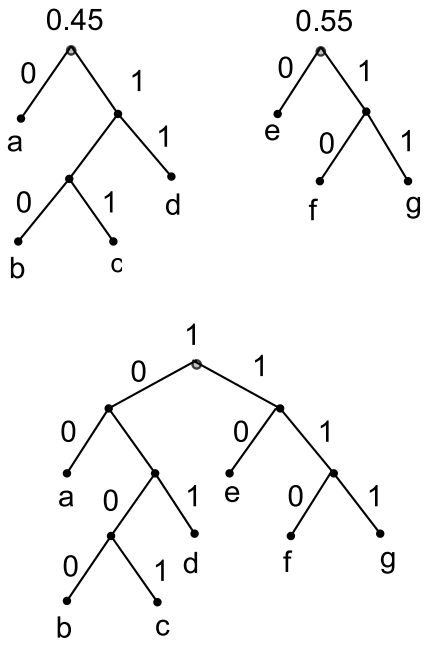


Figure 3.16: Iterations 5-6 in Huffman's algorithm for Example 3.9.

Decoding scheme using a Huffman tree

A given codeword determines the unique path from the root to that leaf which stores the corresponding symbol. For decoding, we scan the codeword from left to right and traverse the tree from the root. When a leaf is reached, its corresponding symbol is recorded. The next bit begins a new path from root to the next symbol. We continue until all the bits in the codeword have been read.

Example: By decoding the following bit-strings

$$1100001010001110,\ 0100001111110011110,\ 11000010110011$$

we obtain the strings *facade*, *baggage*, *faced*, respectively.

Correctness of Huffman's algorithm

We need to prove that Huffman's algorithm produces an *optimal* prefix code (with the smallest AWL). This can be done by induction, but first we need a preliminary result.

Lemma 3.20 *If the leaves of a binary tree are assigned weights and if each internal vertex is assigned a weight equal to the sum of its children's weights, then the tree's AWD equals the sum of the weights of its internal vertices:*

$$\mathcal{W}(T) = \sum_{v \in I} w(v)$$

(I denotes the set of internal vertices of T).

Proof. For each leaf (symbol) s_k, we denote $d_k = \text{depth}(s_k)$. The contribution of the symbol s_k to the right-hand side is $w_k \cdot d_k$, since the weight w_k appears at levels $0, 1, 2, \ldots, d_{k-1}$, so d_k times. The contribution of s_k to the left-hand side will be (from Definition 3.28) $\text{depth}(s_k) \cdot w_k = d_k w_k$. ■

Theorem 3.21 *Let $S = \{s_1, \ldots, s_l\}$ be a set of symbols. For every given list of weights $W = \{w_1, \ldots, w_l\}$, a Huffman tree has the smallest possible AWD among all binary trees whose leaves are assigned those weights.*

Proof. We use induction on l.

For $l = 2$, the tree has a root with weight $w_1 + w_2$, a left-child and a right-child, therefore $AWD = 1 \cdot w_1 + 1 \cdot w_2 \to \min$.

For a given $l_0 \geq 2$, assume that Algorithm 2 produces a Huffman tree H of minimum AWD for any list of l_0 weights. Let $w_1, w_2, \ldots, w_{l_0+1}$ be a list of $l_0 + 1$ weights and suppose that w_1 and w_2 are two of the smallest ones, chosen first by the algorithm. Thus, the leaves s_1 and s_2 are brothers. Let y denote their father and let $\overline{H} = H - s_1 - s_2$. Then $\overline{H}$ coincides with the tree obtained by applying Huffman's algorithm for the set $\{y, s_3, s_4, \ldots, s_{l_0+1}\}$, with $w(y) = w_1 + w_2$. But,

$$\text{depth}(s_1) = \text{depth}(s_2) = \text{depth}(y) + 1, \quad (\text{in } H)$$

therefore

$$\begin{aligned}
\mathcal{W}(H) &= \sum_{i=1}^{l_0+1} \text{depth}(s_i) \cdot w_i \\
&= \text{depth}(s_1)w_1 + \text{depth}(s_2)w_2 + \sum_{i=3}^{l_0+1} \text{depth}(s_i) \cdot w_i \\
&= w_1 + w_2 + \text{depth}(y)(w_1 + w_2) + \sum_{i=3}^{l_0+1} \text{depth}(s_i) \cdot w_i \\
&= w_1 + w_2 + \mathcal{W}(\overline{H}).
\end{aligned}$$

By the hypothesis of induction, $\overline{H}$ is optimal among all binary trees whose leaves are assigned weights $w_1 + w_2, w_3, \ldots, w_{l_0+1}$.

Suppose now that T^* is an optimal binary tree for the weights $w_1, w_2, w_3, \ldots, w_{l_0+1}$. Let x be an internal vertex of T^* of greatest depth and denote z and v its children. Without loss of generality, we can assume that the weights of z and v are w_1 and w_2, since otherwise we could swap their weights with w_1 and w_2 to produce a tree with smaller AWD.

Consider the tree $\overline{T} = T^* - z - v$. Then

$$\mathcal{W}(T^*) = \mathcal{W}(\overline{T}) + w_1 + w_2.$$

But $\overline{T}$ is a binary tree with leaves of weights $w_1 + w_2, w_3 \ldots, w_{l_0+1}$, and hence

$$\mathcal{W}(\overline{T}) \geq \mathcal{W}(\overline{H}).$$

Thus, $\mathcal{W}(T^*) \geq \mathcal{W}(H)$ and therefore H is optimal. ■

3.5 Spanning trees

Definition 3.30 *A **spanning tree** for the graph $G = (V, E)$ is a spanning subgraph [8] which is a tree.*

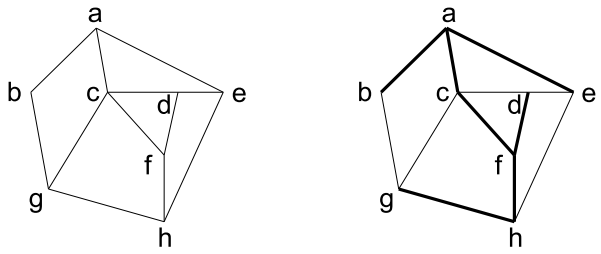


Figure 3.17: A graph and one of its spanning trees.

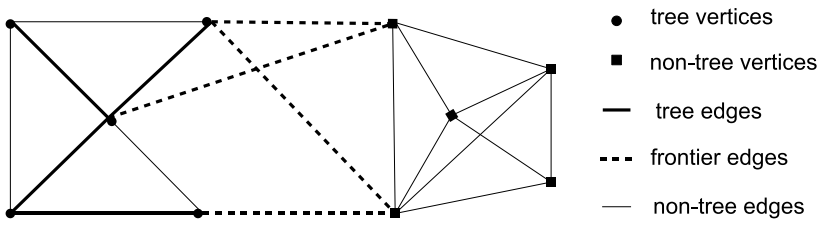


Figure 3.18: A tree in a graph and its frontier edges.

The natural questions that arise are whether a spanning tree exist, given a connected graph and if yes, how can we grow a spanning tree?

Before presenting some methods of growing spanning trees, we need to introduce some terminology.

Definition 3.31 *Let $G = (V, E)$ be a graph and $G_1 = (V_1, E_1)$ a subgraph of G. An edge of G is called a **frontier edge** for G_1 in G, if it has one endpoint in V_1 and one endpoint in $V \setminus V_1$.*

Many algorithms for growing a spanning tree are based on the following result.

Proposition 3.22 *Let $G = (V, E)$ be a graph, T a tree which is a subgraph of graph G, and e a frontier edge for T. Then the subgraph[9] $T + e$ is also a tree.*

Proof. The addition of the frontier edge e to the tree T cannot create a cycle, since one of the endpoints of e is outside of T. ■

Let us mention that by adding the edge e to T involves also adding a new vertex to T.

Basic tree growing

The basic tree growing scheme use vertex labels to keep track of the order in which vertices are added to the tree. By **vertex labeling** we mean an assignment of integers in $0, 1, 2, \ldots n-1$ (sometimes $1, 2, \ldots, n$) to the n vertices of the graph.

[8] See Definition 3.2, p. 67

[9] See Definition 3.7, p. 70.

Definition 3.32 *Let $G = (V, E)$ be a graph and $x \in V$ a vertex. The* ***component of*** *x* ***in*** *G, denoted $C_G(x)$, is the subgraph of G induced on the set of vertices that are reachable from x.*

Thus, if a graph $G = (V, E)$ is connected, then $C_G(x) = G$ for all $x \in V$. Conversely, if there exists $x \in V$ such that $C_G(x) = G$, then the graph G is connected.

In the following we present a basic algorithm for growing a spanning tree in the component $C_G(x)$. It starts with an edge and adds frontier edges successively to the tree, until all the vertices of the given graph are taken.

Algorithm 3 ***(Basic tree-growing algorithm with vertex labeling)***
Input: *a graph $G = (V, E)$ and a starting vertex $u \in V$.*
Output: *a spanning tree T of $C_G(u)$ and a vertex-labeling for $C_G(u)$.*

- *initialize tree T as vertex u*
- *write label 0 on vertex u*
- *initialize label counter $i = 1$*
- *while T does not yet span $C_G(u)$*
  - *choose a frontier edge e for T*
  - *let v be the endpoint of edge e that lies outside of T*
  - *add edge e and vertex v to tree T*
  - *write label i on vertex v*
  - *set $i = i + 1$*
- *return tree T and vertex labeling of $C_G(u)$*

Remark 3.10 (uniqueness) *Let us mention that, without a rule for choosing a frontier edge, the output tree of Algorithm 3* is not unique. *The uniqueness of the algorithm depends on some default priority based on the ordering of the edges or vertices.*

Remark 3.11 *Whenever Algorithm 3 is applied, the resulting spanning tree can be regarded as a rooted tree with root u.*

Proposition 3.23 *If an execution of the basic tree-growing algorithm starts at a vertex v of graph G, then the subgraph consisting of the labeled vertices and tree edges is a spanning tree of the component $C_G(v)$.*

Corollary 3.24 *A graph is connected if and only if the basic tree-growing algorithm labels all its vertices.*

In the case when the graph G is not connected, we will have a forest-growing algorithm.

Definition 3.33 *A* ***spanning forest*** *of a graph G is a collection of trees, such that each tree is a spanning tree of a different component of G.*

Let us note that a forest is a collection of trees. For forest-growing we give the following basic algorithm.

Algorithm 4 ***(Basic forest-growing algorithm with vertex labeling)***
Input: *a graph $G = (V_G, E_G)$*
Output: *a spanning forest F of G and a vertex- labeling for G.*

- *initialize forest F as empty graph*

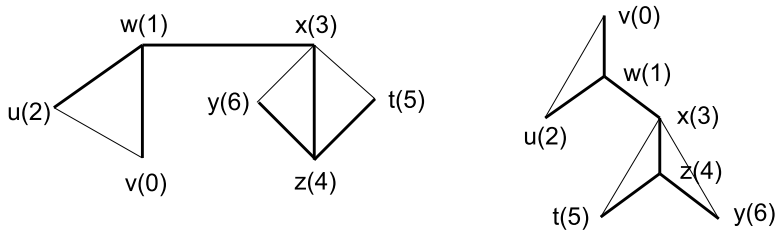


Figure 3.19: DFS tree.

- *initialize component counter* $t = 1$
- *while forest* F *does not yet span* G *(doesn't have yet* $|V_G|$ *vertices)*
  - *choose a vertex* v *which is not in the forest* F
  - *use basic tree-growing to construct spanning tree* T_t *in* $C_G(v)$
  - *add the number* $|V_F|$ *to each vertex label in tree* T_t
  - *add tree* T_t *to forest* F
  - *set* $t = t + 1$
- *return forest* F *and vertex- labeling for* G

3.5.1 Depth-first search and breadth-first search

Depth-first search and breadth-first search are actually tree-growing algorithms, based on Algorithm 3, with wider use. Even if they are sometimes used for search something, the word "search" is somehow misleading, but too well established to try to change it.

Depth-first search (DFS)

The idea of DFS is to select a frontier edge incident on the most recently acquired vertex (i.e., with highest label) of the growing tree. When this is not possible, the algorithm backtracks to the next most recently labeled vertex (i.e., the parent) and tries again. Thus, the search moves deeper into the graph (hence the name *depth-first*).

In the following we will describe the DFS algorithm. An example of DFS is illustrated in Figure 3.19. During the DFS, the label assigned to a vertex w is denoted $dfnumber(w)$.

Algorithm 5 (Depth-first search)
Input: *a connected graph* $G = (V, E)$ *and a starting vertex* $v \in V$
Output: *a depth-first spanning tree* T *and a vertex-labeling for* G

- *initialize tree* T *as vertex* v
- *initialize the set of frontier edges for* T *as empty*
- *set* $dfnumber(v) = 0$
- *initialize label counter* $i = 1$
- *while tree* T *does not yet span* G *(doesn't have yet* $|V|$ *vertices)*
  - *update the set of frontier edges for* T
  - *Let* e *be a frontier edge for* T *whose labeled endpoint has the* `largest` *possible* $dfnumber$.
  - *Let* w *be the unlabeled endpoint of edge* e
  - *Add edge* e *(and vertex* w*) to tree* T
  - *set* $dfnumber(w) = i$

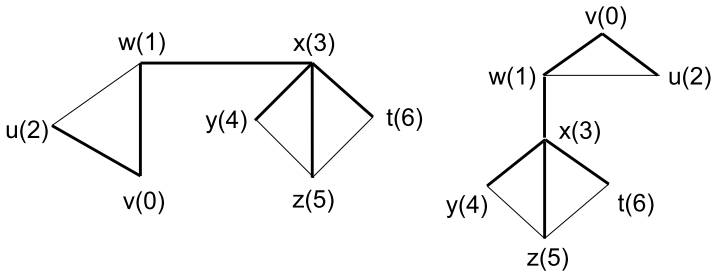


Figure 3.20: BFS tree.

- *set* $i = i + 1$

▪ *return tree* T *with its dfnumbers*

Breadth-first search (BFS)

Whereas in DFS we advance from the current vertex to a new one whenever is possible, in BFS we check all the vertices adjacent to the current vertex before going on to the next one. The selection of frontier edges will be as close to the starting vertex as possible, or, in term of priorities, the frontier edges incident on vertices having the smallest possible label will have the highest priority. An example of construction of BFS is given in Figure 3.20.

Algorithm 6 (Breadth-first search)
The algorithm writes identically with DFS Algorithm 5, except the word `largest`*, which have to be replaced with* `smallest`*.*

Proposition 3.25 *When BFS is applied to an undirected connected graph* $G = (V, E)$*, the end-points of each non-tree edge are either at the same level, or at consecutive levels (or, equivalently, if* r *denotes the starting vertex, the distances*[10] *to* r *are either equal or differ by one).*

Proof. Let $T = (V, E')$ be the spanning tree resulted by applying the BFS algorithm.

Suppose there exists an edge $e = xy \in E \setminus E'$ such that

$$d(r, x) = p, \quad d(r, y) = p + s, \ s > 1.$$

If ω denotes the (r, x)- path in T, then $\omega \cup e$ forms an (r, y)- path of length $p + 1 < p + s$, contradiction with the property that $p + s$ is the length of the shortest (r, y)-path. ■

Theorem 3.26 (Property of tree of distances of the BFS tree) *Let* $G = (V, E)$ *and let* T *be a BFS tree starting from* $v \in V$*. Then, for each vertex* x*, the distance* $d(v, x)$ *(in* G*) equals the length of the unique path in* T *between* v *and* x*. That is, the BFS tree is a* tree of distances *with respect to the starting vertex* v*.*

Proof. Let $k = d(v, x)$. We construct a partition of V into the disjoint classes

$$\ell_v(p) = \{y \in V, \ d(v, y) = p\}, \quad p = 0, 1, \ldots, m.$$

[10]The definition of the distance between two vertices was given on p. 69.

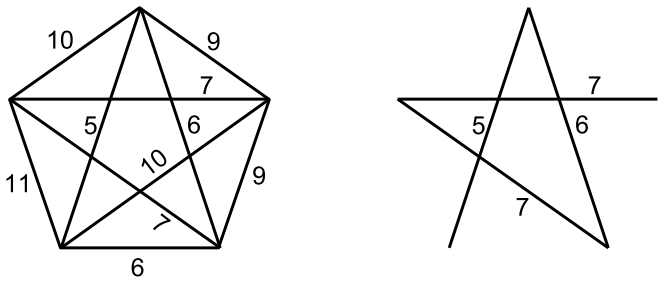


Figure 3.21: A graph and a minimum spanning tree.

So, $x \in \ell_v(k)$ and its parent in T will be a vertex $y_1 \in \ell_v(k-1)$. The parent of y_1 in T will be a vertex $y_2 \in \ell_v(k-2)$, and continuing we rich the vertex $y_k \in \ell_v(0)$, that is $y_k = v$.

Therefore we have obtained the path $\langle v, y_{k-1}, y_{k-2}, \ldots, y_2, y_1, x\rangle$ in T, which has the length k. ■

3.5.2 Weighted graphs and minimum spanning tree

Definition 3.34 *A **weighted graph** is a graph in which each edge is a assigned a number, called its* edge-weight. *The weight of the edge e will be denoted $w(e)$.*

Weighted graphs occur frequently in applications. In the friendship graph, for example, weights might indicate the intensity of friendship; in the communication graph, they could represent transportation cost, travel time, spatial distance, price of communication or maintenance.

Suppose that several computers in fixed locations are to be linked to form a computer network (see Figure 3.21). The cost of creating a direct connection between a pair of computer is known for each possible pair and is represented by the edge weights. Of course we need the least expensive connection. If we associate a graph in which the vertices are the computers, the edges are the cables linking them and the weight of each edge is the cost of connection, then our goal will be to determine a spanning tree of minimum weight.

Definition 3.35 *Let $G = (V, E)$ be a connected weighted graph, $T = (V, E_1)$ a spanning tree of G and $\omega = \langle e_1, e_2, \ldots, e_n\rangle$ a walk in G.*

*The **weight of a walk** $\omega = \langle e_1, e_2, \ldots, e_n\rangle$ is the sum of its edge-weights:*

$$w(\omega) = \sum_{i=1}^{n} w(e_i).$$

*The **weight of the spanning tree** T is the sum of the weights of its edges:*

$$w(T) = \sum_{e \in E_1} w(e).$$

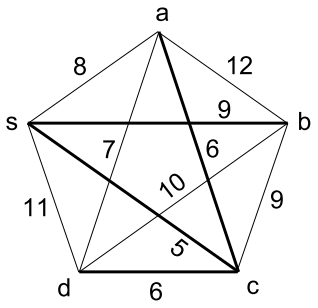


Figure 3.22: The MST obtained with Algorithm 7.

A tree $T_{\min}$ for which $w(T_{\min})$ is minimum is called ***minimum spanning tree*** *(MST). So,*

$$w(T_{\min}) = \min\{w(T),\ T \textit{spanning tree of } G\}.$$

Minimum spanning tree problem (MSTP): Let G be a connected weighted graph. Find a spanning tree of G whose total edge-weight is minimum.

A naive solution would be to write all spanning trees, to calculate their weights and take the minimum. In this case, if the graph G has n vertices and is complete, then there are n^{n-2} spanning trees,[11] which are impossible to be analyzed in real time. Therefore one needs more efficient algorithms for solving the (MSTP).

An algorithm which solves (MSTP) is known under the name **Prim's algorithm.** It was discovered in 1930 by Jarnìk[12] and later, independently, by Prim[13] in 1957 and Dijkstra in 1959. Therefore it is sometimes called the **DJP algorithm** or **Jarnìk algorithm**.

Algorithm 7 (Prim's algorithm for finding a MST)
Input: *a weighted connected graph $G = (V, E)$*
Output: *a minimum spanning tree T*

- *choose an arbitrary vertex s of graph G*
- *initialize the Prim tree T as vertex s*
- *initialize the set of frontier edges as empty*
- *while T does not yet span G (doesn't have yet $|V|$ vertices)*
  - *update the set of frontier edges for T*
  - *let e be a frontier edge with the smallest edge-weight $w(e)$*
  - *add edge e to tree T*
- *return Prim tree T*

For the graph in Figure 3.22, the steps of Prim's algorithm starting with the vertex s are given in the following table:

step	frontier edges	added edge
1	sd, sa, sc	sc
2	cd, ca, cb, sa, sd, sb	ac
3	ab, bc, cd, sb, sd	cd
4	ab, bc, db, sb	sb

[11] This result is known as *Cayley's tree formula.*
[12] Vojtěch Jarnìk (1897-1970), Czech mathematician.
[13] Robert Clay Prim (b. 1921), American mathematician and computer scientist.

Theorem 3.27 (Correctness of Prim's algorithm) *The output of Prim's algorithm is a minimum spanning tree.*

Proof. Let $G = (V, E)$ a connected weighted graph, T the output of Prim's algorithm applied to G and Y a minimum spanning tree of G. Denote by $T_k = (V_k, E_k)$ the Prim's tree constructed after k iterations.

For simplicity we start with the case when all edges have different weights.

$\circledast$ If $Y = T$, then T is a MST. If not, we should obtain a contradiction.

So, suppose $Y \neq T$. Let e be the first edge considered by the algorithm, which is in T but not in Y. Let $T_i = (V_i, E_i)$ be the set of vertices connected by the edges added before e. So, one endpoint of e is in V_i and the other is not in V_i. Since Y is a spanning tree of G, there exists a path in Y joining the two endpoints of e. As one moves along the path, one must meet an edge f joining a vertex in V_i to one not in V_i. At the $(i+1)$-th iteration, when e was added to T, f was a frontier edge, so f could also have been added to T. Since f was not added, we conclude that

$$w(f) > w(e). \tag{3.7}$$

Consider now the subgraph $Y_1 = Y - f + e$, which is a tree. Indeed, $Y + e$ has a cycle which contains f, and therefore $Y + e - f$ has no cycle and is connected. Let us note that Y_1 has in T one edge more than has Y. For Y_1 we have

$$w(Y_1) = w(Y) - w(f) + w(e) < w(Y), \tag{3.8}$$

which is a contradiction with the fact that Y is a MST. So, $T = Y$ and therefore T is a MST.

To finalize the proof, we have to consider the case where we allow equal weights. The proof above also works, until the point where we conclude (3.7). It will be replaced with $w(f) \geq w(e)$, whence inequality (3.8) changes into $w(Y_1) \leq w(Y)$. At this point we do not have a contradiction, but the conclusion that Y_1 is also a MST. We "replace" now Y with Y_1 and we repeat the proof from the point $\circledast$, for the "new" MST Y. Since Y_1 has in T one more edge than Y does, the algorithm will finish after at most $|V| - 1$ steps, where at the point $\circledast$ we must have $Y = T$. ■

We finish by mentioning that, in the case when any distinct edges have different weights, the minimum spanning tree is unique. In the case when we have equal weights, it is possible to have more spanning trees, and Prim's algorithm will find one of them.

Another algorithm used for finding a minimum spanning tree was given by Kruskal[14] in 1956. The algorithm first orders the edges by weight and then proceeds through the ordered list adding an edge to the partial graph provided that adding the new edge does not create a cycle.

Algorithm 8 (Kruskal's algorithm for finding a MST)
Input: *a weighted connected graph $G = (V, E)$.*
Output: *a minimum spanning tree T*

▪ *initialize the Kruskal graph K as a forest where each vertex in V is a separate tree*

[14] Joseph Bernard Kruskal (b. 1928), American mathematician, statistician and psychometrician.

- *create a set S containing all the edges in G*
- *while S is nonempty*
  - *remove an edge with smallest edge-weight from S*
  - *if that edge does not close a cycle in K (when added to K), then add it to K*
  - *else discard that edge*
- *return Kruskal tree K*

Remark 3.12 *At the step where we check if the edge e closes a cycle, we check actually whether it connect two different trees of the forest K. When e is added to the graph K, it will combine the two trees into a single tree. If the graph G is not connected, then Algorithm 8 does not create a MST, but a* minimum spanning forest*.*

Theorem 3.28 (Correctness of Kruskal's algorithm) *Let G be a connected weighted graph and let K be the subgraph of G produced by Kruskal's algorithm. Then K is a minimum spanning tree.*

Proof. First, by construction, K cannot have a cycle. Then K cannot be disconnected. Indeed, if we suppose that K has two components, then at one point of the algorithm, one adds the minimum weight edge which joins the two components. Thus, K is a spanning tree of G.

For simplicity, we start with the case where all edges have different weights. Let Y be a minimum spanning tree and let K_k denote the graph constructed at the end of the k-th iteration.

★ If $Y = K$, then K is a minimum spanning tree. If not, we should obtain a contradiction.

So suppose that $Y \neq K$. Let e be the first edge considered by the algorithm, which is in K but not in Y and let K_i denote the graph constructed until the point when edge e was added. Then $Y + e$ has a cycle (one cannot add an edge to a spanning tree and still have a tree). This cycle contains another edge f which, at the stage of the algorithm where e was added to K_i, has not been considered. The reason for which f was not considered is because $w(f) > w(e)$, and not because f would have been closed a cycle, if it were added to K_i. (Indeed, if f closed a cycle if added to K_i, then Y would contain a cycle, namely $K_i + f$, so f was a frontier edge at the beginning of $(i+1)$-th iteration).

The subgraph $Y_1 = Y + e - f$ is also a spanning tree, with the weight

$$w(Y_1) = w(Y) + w(e) - w(f) < w(Y), \tag{3.9}$$

which is a contradiction with the fact that Y is a MST. Thus the assumption $K \neq Y$ is false, whence $K = Y$, therefore K is a MST.

To finalize, we have to consider also the case when some weights are equal. We repeat the arguments above but at the point when we reach inequality (3.9), it transforms into

$$w(Y_1) = w(Y) + w(e) - w(f) \leq w(Y).$$

It implies this time that Y_1 is also a MST. In this case we "replace" Y with Y_1 and repeat the proof from the point ★ with the "new" Y, until we obtain $Y = K$. ■

Remark 3.13 *Another argumentation for the correctness will be given in Example 3.16, p. 111.*

3.5.3 Minimum spanning tree in directed graphs

Definition 3.36 *Let $D = (V, A)$ be a digraph. A **spanning rooted tree** of D is a rooted tree (see Definition 3.21, p. 85) which contains all the vertices of D.*

Proposition 3.29 *Let $D = (V, A)$ be a strongly connected digraph and let r be an arbitrary vertex. Then there exists a spanning rooted tree of D, rooted at r.*

As in the case of graphs, we can associate to a digraph $D = (V, A)$ a weight function $w : A \to \mathbb{R}_+$, and obtain a **weighted digraph**. The **weight** of D is defined as

$$w(D) = \sum_{a \in A} w(a).$$

For a strongly connected weighted digraph $D = (V, A)$ and a fixed vertex $r \in V$, we want to find a minimum-weighted spanning rooted tree $T^*(r)$, rooted at r, in the sense that

$$w(T^*(r)) \leq w(T(r)),$$

for any other spanning rooted tree $T(r)$ of D, rooted at r. One algorithm for growing such a rooted tree was discovered by Chu and Liu in 1965 and then independently by Edmonds in 1967. Before giving this algorithm, we introduce some notations.

For a set of vertices $U \subseteq V$, we introduce the following notations:

$$\begin{aligned} A^-(U) &= \{x \in V : (x, u) \in A \text{ and } u \in U\}, \\ A^+(U) &= \{x \in V : (u, x) \in A \text{ and } u \in U\}. \end{aligned}$$

Thus, $A^-(U)$ is the set of tails of arcs with heads in U and $A^+(U)$ is the set of heads of arcs with tails in U.

Algorithm 9 (Chu, Liu, Edmonds)
Input: - *a weighted strongly connected digraph $D = (V, A)$ with the weight $w : A \to \mathbb{R}_+$*
- *a vertex r*
Output: *a minimum spanning rooted tree, with the root at r*
- *for each vertex v different from r choose the arc with the head v, of minimum weight*
- *let S be the set of selected arcs (one arc for each v)*
- *initialize $M = S$*
- *while $T = (V, M)$ is not a rooted tree*
  - *let C be a directed cycle in T, denoted $C = \{v_1, v_2, \ldots, v_k, v_1\}$*
  - *for all $v_i \in A^-(C) \setminus C$ calculate $w(i, C) = \min\{w(i, v_j) + w(C) - w(v_{j-1}, v_j),\ v_j \in A^+(i) \cap C\}$, where v_{j-1} is the tail of the arc (v_{j-1}, v_j) of C with head v_j*
  - *select $i_0 \in A^-(C)$ for which $w(i_0, C)$ is minim*
  - *let $v_{j_0} \in C$ for which*

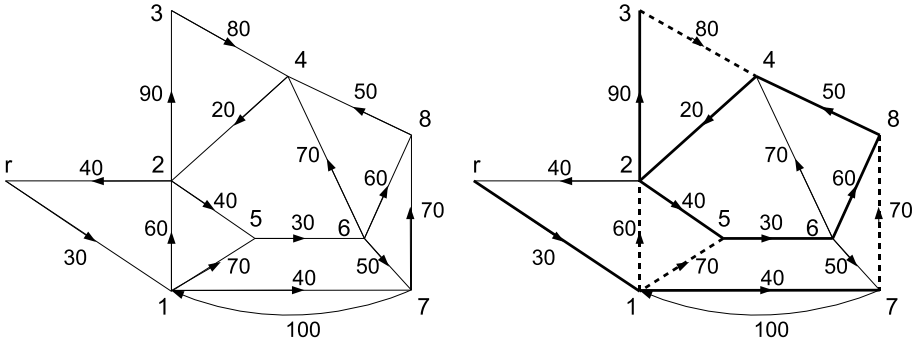


Figure 3.23: Left: The strongly connected digraph in Example 3.10. Right: the set S (bold) and the arcs candidates for replacing one of the arcs in the cycle C (dotted).

$$w(i_0, C) = w(i_0, v_{j_0}) + w(C) - w(v_{j_0-1}, v_{j_0})$$

- $M = M - (v_{j_0-1}, v_{j_0}) + (v_{i_0}, v_{j_0})$

▪ *return M, the required spanning directed tree, rooted at r*

Theorem 3.30 (Correctness of Chu-Liu-Edmonds algorithm) *The rooted tree obtained by applying Algorithm 9 to a digraph $D = (V, A)$ for a given starting vertex $r \in V$ is the minimum spanning tree for D, rooted at r.*

Proof. First we observe that the number of arcs in S is $|V| - 1$ and remains unchanged during the algorithm. Therefore, at the moment when the graph T does not contain any more cycles, it will be a spanning rooted tree. We prove that, when the weight function is injective, the spanning rooted tree has minimum weight. When the weight is not injective, we use the same arguments as in the proof of Theorem 3.27 (correctness of Prim's algorithm).

Let T denote the output tree of Algorithm 9 and let $T^* = T^*(r)$ be the minimum spanning tree rooted at r. Suppose that $T^*(r) \neq T$. Then there exist the arcs $a_1 = (i, j) \in T^* \setminus T$ and $a_2 = (k, j) \in T$.

If $a_2 \in S$, then $w(a_2) < w(a_1)$. If $a_2 \notin S$, then j is vertex in a cycle $C = \{v_j, v_{j+1}, \ldots, v_{j-1}, v_j\}$ in S, with $v_j = j$. From the criterion of selection of arc $a_2 \in T$, we conclude that

$$w(k, j) + w(C) - w(v_{j-1}, v_j) < w(i, j) + w(C) - w(v_{j-1}, v_j),$$

whence $w(a_2) < w(a_1)$. We denote by T_1 the rooted tree obtained from T^* by replacing the arc a_1 with the arc a_2. Then $w(T_1) < w(T^*)$, which contradicts the minimality of T^*. ■

Example 3.10 *We consider the digraph in Figure 3.23 (left) and we apply Algorithm 9, with the starting vertex r.*

At the first step, the set S is

$$S = \{(r, 1), (4, 2), (2, 3), (8, 4), (2, 5), (5, 6), (1, 7), (6, 8)\}.$$

This set of arcs form the cycle $C = \langle 2, 5, 6, 8, 4, 2\rangle$ (see Figure 3.23, right) of weight $w(C) = 200$. The set $A^-(C)$ of tails of arcs with heads in C is

$$A^-(C) = \{1, 3, 7\},$$

and for each of the vertices in this set we calculate

$$A^+(1) \cap C = \{2,5\},\ A^+(3) \cap C = \{4\},\ A^+(7) \cap C = \{8\}.$$

Then

$$\begin{aligned} w(1,C) &= \min\{w(1,v_j) + 200 - w(v_{j-1},v_j),\ v_j = 2,5\} \\ &= \min\{60+200-20, 70+200-40\} = 230, \\ w(3,C) &= w(3,4) + 200 - w(8,4) = 230, \\ w(7,C) &= w(7,8) + 200 - w(6,8) = 210. \end{aligned}$$

So, $i_0 = 7, v_{j_0} = 8$ and the arc $(6,8)$ will be replaced with the arc $(7,8)$. At this moment

$$M = \{(r,1),(4,2),(2,3),(8,4),(2,5),(5,6),(1,7),(7,8)\},$$

and it forms a rooted spanning tree, which has minimum weight equal to 210, so the algorithm stops.

3.5.4 Shortest path. Dijkstra's algorithm

Another optimization problem is the shortest path problem. Given, for example, a railway network connecting various towns, determine a shortest route between two specified towns in the network. The formulation of the problem is the following:

Shortest Path Problem: Let s, t be vertices of a connected weighted graph G. Find a path from s to t whose total edge-weight is minimum.

Remark 3.14 *If the weights are all 1, then the problem reduces to finding a path between s and t that uses a minimum number of edges. This version of SPP has already been solved with BFS algorithm.*

The SPP problem was solved in 1959 by Dijkstra.[15] His algorithm does actually slightly more than the problem requires: it finds the shortest path from a given vertex s to *each* of the vertices of the graph.

Definition 3.37 *Let $G = (V,E)$ be a weighted graph. The **distance** between two vertices $x, y \in V$, denoted by $dist(x,y)$, is the length of the shortest path between them:*

$$dist(x,y) = \min\{w(p),\ p \text{ path between } x \text{ and } y\}.$$

The idea of the algorithm is to grow a Dijkstra tree, starting at a given vertex s, by adding, at each iteration, a frontier edge whose non-tree endpoint is as close as possible to s. For simplicity, once the vertex s is fixed, for an arbitrary vertex x we will denote $\text{dist}(s,x)$ by $\text{dist}(x)$. At each iteration, one adds an edge (and implicitly a vertex) to the Dijkstra tree, and then one writes a label on the new vertex. Thus, the vertices in the Dijkstra tree at a given iteration are those to which shortest paths from s have already been found.

Let the P-value of the frontier edge e be

$$P(e) = dist(x) + w(e),$$

[15] Edsger Wybe Dijkstra ['dɛikstra] (1930-2002), Duch computer scientist.

where x is the labeled endpoint of e. The edge with the smallest P-value is given the highest priority. Moreover, the P-value of its highest priority edge will give the distance from s to the unlabeled endpoint of e. The algorithm labels each tree vertex with the distance between vertex s and that vertex.

Algorithm 10 (Dijkstra's shortest path)
Input: *a weighted connected graph $G = (V, E)$ with nonnegative weights and $s \in V$*
Output: *– a spanning tree T of G, rooted at s, whose path between s and each vertex v is the shortest path between s and v in G*
– a vertex labeling giving the distance from s to each vertex

- *initialize Dijkstra tree T as vertex s*
- *initialize the set of frontier edges for T as $\emptyset$*
- *set $dist(s) = 0$*
- *write label 0 on vertex s*
- *while tree T does not yet span G*
  - *for each frontier edge e for T*
    let x be the labeled endpoint of edge e
    let y be the unlabeled endpoint of edge e
    set $P(e) = dist(x) + w(e)$
  - *let e be a frontier edge for T that has the smallest P-value*
  - *let x be the labeled endpoint of edge e*
  - *let y be the unlabeled endpoint of edge e*
  - *add edge e (and vertex y) to tree T*
  - *$dist(y) := P(e)$*
  - *write label $dist(y)$ on vertex y*
- *return Dijkstra tree T and its vertex labels*

Theorem 3.31 (correctness of Dijkstra's algorithm) *Let G be a connected weighted graph and T the Dijkstra tree produced by Algorithm 10 starting at vertex s. Then, for each vertex x of T, the unique path in T between s and x is the shortest path in G between s and x.*

Proof. The proof follows the same ideas as the proof of Theorem 3.27. ■

Example 3.11 *Consider the weighted graph in Figure 3.24. If we apply Algorithm 10 starting at vertex s, the steps are the following:*

Step 1 $dist(s) = 0$,
$P(sz) = 8$
$P(sy) = 16$ $\Rightarrow \min = 8, \Rightarrow sz$ added, $dist(z) = 8.$
$P(sw) = 13$

Step 2
$P(sw) = 13$
$P(sy) = 16$
$P(zy) = 15$
$P(zv) = 18$ $\Rightarrow \min = 13 \Rightarrow sw$ added, $dist(w) = 13.$
$P(zw) = 19$
$P(zx) = 25$

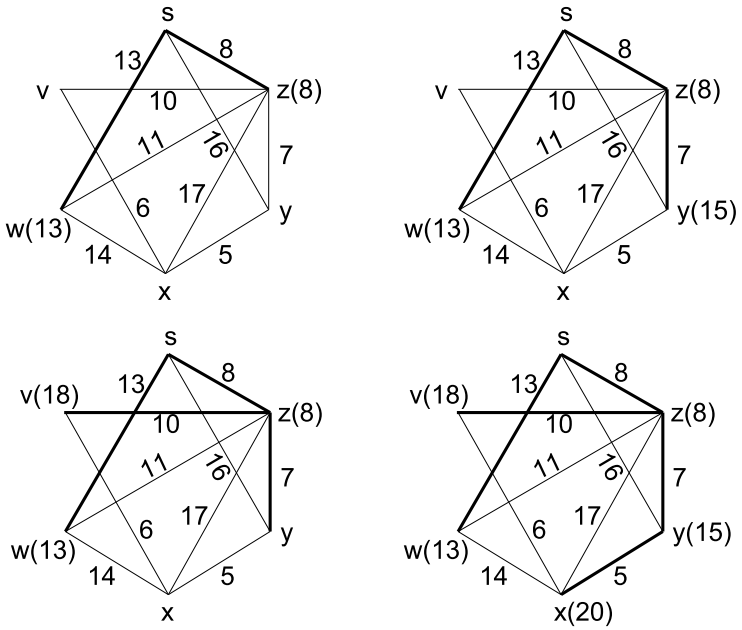


Figure 3.24: Dijkstra trees (bolded) after the steps 2,3,4,5.

Step 3

$P(sy) = 16$
$P(zy) = 15$
$P(zx) = 25$ $\Rightarrow \min = 15 \Rightarrow zy$ added, $dist(y) = 15.$
$P(zv) = 18$
$P(wx) = 27$

Step 4

$P(yx) = 20$
$P(zx) = 25$
$P(zv) = 18$ $\Rightarrow \min = 18 \Rightarrow zv$ added, $dist(v) = 18.$
$P(wx) = 27$

Step 5

$P(wx) = 27$
$P(yx) = 20$
$P(vx) = 24$ $\Rightarrow \min = 20 \Rightarrow yx$ added, $dist(x) = 20.$
$P(zx) = 25$

For directed graphs, there exist more algorithms for finding a directed spanning tree. Among them we mention the Moore-Dijkstra algorithm, the Bellman-Kalaba algorithm and the Ford algorithm, with their variations.

3.6 Greedy algorithms

An optimization problem is one in which one wants to find not only *a* solution, but the *best* solution.

A greedy algorithm is an algorithm that "gobbles up" all of its favorites first, without worrying about the consequences. It uses a single procedure again and again, until it can't do anything.

Mathematically, we say that it takes the best *immediate* (or **local**) solution, without worrying about the effect of the decision in the future, but

hoping that by choosing a *local optimum* at each step, one ends up at a **global optimum**.

Example 3.12 *Suppose we want to give rest an amount of €6.39 using the fewest possible bills and coins. A Greedy algorithm would do this: at each step, take the largest possible bill or coin. The available pieces are*

$$5€,\ 2€,\ 1€,\ 50\,c,\ 20\,c,\ 10\,c,\ 5\,c,\ 2\,c,\ 1\,c,$$

so our rest can be given as follows:

$$6.39 = 5€ + 1€ + 20\,c + 10\,c + 5\,c + 2 \cdot 2\,c. \quad (7\ \textit{pieces})$$

For euro, the greedy algorithm always gives the optimum solution.

But, Suppose we are given a fictional monetary system, with the coins $10u$, $7u$, $1u$. *With a greedy algorithm, one pays the amount of* $15u$ *as*

$$15u = 1 \cdot 10u + 5 \cdot 1u \quad (6\textit{pieces}).$$

There exists however a payment with only 3 coins, namely

$$15u = 2 \cdot 7u + 1 \cdot 1u.$$

So, GA gives ***a*** *solution, but* ***not an optimal*** *solution.*

Other examples of GA are Prim's, Kruskal's and Dijkstra's algorithms, but as we could see, they always lead to the optimal solution. So, for some optimization problems, a greedy algorithm can however find the optimal solution.

The natural questions which arises are: When does a GA find the optimal solution? If we know that it is possible not to find the optimal solution, why do we apply it? We will give a partial answer to the first question in the next section. Regarding the second question, the reason we use GA is that usually such algorithms are quick, easy to implement, and often give a good approximation to the optimum.

3.6.1 Greedy algorithm for the maximum weight problem

Let E be a finite set and $\mathcal{I}$ a family of subsets of E, called **admissible sets**. Then the pair $\mathcal{S} = (E, \mathcal{I})$ is called an **admissible system of (sub)sets**. If $w : E \to \mathbb{R}_+$ is a weight function, then for the subset $N \in \mathcal{I}$ we define its weight as

$$w(N) = \sum_{x \in N} w(x).$$

The maximum weight problem $P_{\max}$: Find an admissible set with maximum weight.

A greedy algorithm for solving this problem is the following:

Algorithm 11 (Greedy Algorithm for $P_{\max}$)
Input: $\mathcal{S} = (E, \mathcal{I})$ *a system of admissible subsets*
Output: *a solution of* $P_{\max}$ *(at least we hope for the maximality)*

- *initialize* $M = \emptyset$ *and* $A = E$

- *while* $A \neq \emptyset$
  - *choose* $e \in A$ *with maximum weight*
  - *set* $A = A - e$
  - *if* $M + e \in \mathcal{I}$, *then* $M = M + e$
- *return the set* M

So, each $e \in E$ was considered once, being either added or definitively eliminated. Thus, the algorithm is fast and simple, but it is not sure that it is optimal.

Example 3.13 *Let* $E = \{e_1, e_2, e_3\}$, *with the weights*

$$w(e_1) = 3,\ w(e_2) = w(e_3) = 2,$$

and $\mathcal{I} = \{\{e_1\}, \{e_2\}, \{e_2, e_3\}\}$.

The solution of $P_{\max}$ *is* $M = \{e_2, e_3\}$, *having* $w(M) = 4$, *while the solution obtained with Algorithm 11 is* $M = \{e_1\}$.

The fact that GA did not find the optimal solution is caused by the fact that there is no link between the sets in $\mathcal{I}$.

Definition 3.38 *A system of sets* $\mathcal{S} = (E, \mathcal{I})$ *is called* ***hereditary*** *if it is* closed *under inclusion, i.e.,*

$$\text{if } A \in \mathcal{I} \text{ and } A' \subseteq A,\ \text{then } A' \in \mathcal{I}.$$

Example 3.14 *Let* $E = \mathbb{R}^3$ *and* $\mathcal{I}$ *the set of all sets of linearly independent vectors. Then* $\mathcal{S} = (\mathbb{R}^3, \mathcal{I})$ *is an hereditary system of subsets, since any subset of linearly independent set of vectors is linearly independent.*

Example 3.15 *Let* $G = (V_G, E_G)$ *be a graph and* $\mathcal{I}$ *the set of subsets of* E_G *whose induced subgraphs are acyclic subgraphs of* G. *Then* $S = (E_G, \mathcal{I})$ *is an hereditary system of sets.*

In the sequel we will consider the problem $P_{\max}$ for hereditary systems of sets.

Proposition 3.32 *If the system* $\mathcal{S} = (E, \mathcal{I})$ *has the property that for every* $w : E \to \mathbb{R}_+$ *greedy algorithm 11 leads to a solution of* $P_{\max}$, *then* $(E, \mathcal{I})$ *is an hereditary system of sets.*

Proposition 3.33 *By applying GA to an hereditary system of sets* $(E, \mathcal{I})$ *and a given weight function* $w : E \to \mathbb{R}$, *we obtain a* ***maximal*** *set, in the sense*

$$\text{there exists no } x \in E \text{ such that } M \cup \{x\} \in \mathcal{I}.$$

Definition 3.39 *An hereditary system of sets* $\mathcal{M} = (E, \mathcal{I})$ *is called a* ***matroid*** *if it satisfies the following condition, which is referred to as the* ***augmentation property****:*

$$(AP): \quad \begin{array}{c} \text{If } I, J \in \mathcal{I} \text{ and } |I| < |J|,\ \text{then there exists } e \in J \setminus I \\ \text{such that } I \cup \{e\} \in \mathcal{I}. \end{array}$$

Proposition 3.34 *If an hereditary system of sets* $(E, \mathcal{I})$ *has the property that for every weight* $w : E \to \mathbb{R}_+$ *the GA leads to a solution of* $P_{\max}$, *then this system has also the augmentation property.*

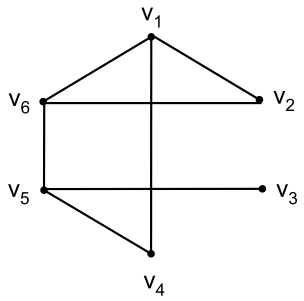


Figure 3.25: The graph for the timetable problem.

Theorem 3.35 *A system of sets $(E, \mathcal{I})$ has the property that for every $w : E \to \mathbb{R}_+$, greedy algorithm leads to a solution of $P_{\max}$, if and only if $(E, \mathcal{I})$ is a matroid.*

Example 3.16 (matroid) *Let $G = (V, E)$ be a graph and let $\mathcal{I}$ be the set of edges with the property that the graph generated by these edges does not contain cycles. Then $\mathcal{S} = (E, \mathcal{I})$ is a matroid.*

Thus, Kruskal's algorithm (p. 102) leads to a minimum spanning tree.

3.6.2 Greedy algorithm for vertex coloring

Suppose we want to timetable six one-hour lectures $V = \{v_1, v_2, \ldots, v_6\}$. Among the potential audience there are people who wish to hear both v_1 and v_2, v_1 and v_4, v_3 and v_5, v_2 and v_6, v_4 and v_5, v_5 and v_6, v_1 and v_6. How many hours are necessary in order that the lectures can be given without clashes? We can represent the situation by a graph (see Figure 3.25), whose vertices are the six lectures and the edges represent potential clashes. A possible timetable avoiding clashes is:

Hour 1	Hour 2	Hour 3	Hour 4
v_1 and v_3	v_2 and v_4	v_5	v_6

Mathematically we construct a partition of V into four parts, such that no part contains a pair of adjacent vertices of the graph. Actually, we define a function

$$c : \{v_1, v_2, v_3, v_4, v_5, v_6\} \to \{1, 2, 3, 4\},$$

which assign to each vertex (lecture) the hour scheduled for it. Usually we speak about colors assigned to the vertices and what is important is that vertices which are adjacent in the graph must have different colors.

Definition 3.40 *A **vertex coloring** of a graph $G = (V, E)$ is a function $c : V \to \mathbb{N}$ with the property*

$$c(x) \neq c(y) \quad \textit{whenever } xy \in E.$$

*The **chromatic number** of G, denoted $\chi(G)$, is defined as the least integer k for which there is a vertex coloring of G using k colors.*

Returning to Figure 3.25, our timetable is equivalent to a vertex coloring using four colors. But the chromatic number of the graph is three, since there

exist the coloring

Color 1	Color 2	Color 3
v_1	v_2 and v_5	v_3, v_4 and v_6

Furthermore, there is no coloring with only two colors since v_1, v_2, v_6 are mutually adjacent, so they must have different colors. In conclusion, the chromatic number of the graph is 3.

In general it is a difficult problem to find $\chi(G)$ and there is no fast algorithm (polynomial time) which do this. a method for constructing a vertex coloring with a reasonable number of colors is to assign colors to the vertices in order, in such a way that each vertex receives the first color which has not already be assigned to one of its neighbors. Thus, we take the best choice at each step, without looking ahead to see if that choice will create problems later, in other words we use a greedy algorithm.

Algorithm 12 (Greedy algorithm for vertex coloring)
Input: *a graph $G = (V, E)$ with $|V| = n$*
Output: *a vertex coloring for G*

- *assign color 1 to v_1*
- *for i=2 to n*
  - *let S be the empty set of colors*
  - *for $j = 1$ to $i - 1$*
    if v_j is adjacent to v_i
    then add the color of v_j to S
  - $k = 1$
  - *while color k is in S do $k = k + 1$*
  - *assign color k to vertex v_i*
- *return a color for each vertex in V*

We finish by giving a result concerning the chromatic number of a particular class of graphs.

Definition 3.41 *A graph is called **planar** if it can be drawn in a plane without edges crossing.*

Theorem 3.36 (The four-color theorem) *Any map in a plane can be colored using four colors in such a way that regions sharing a common boundary (other than a single point) do not share the same color.*

This result was first conjectured[16] by Guthrie in 1853. At that time, it was not difficult to prove that five colors are sufficient. The four-color theorem is the first major theorem which was proved using a computer. Since it was conjectured, there were many attempts at proving, each proof shown after years to be incorrect. Finally, it was proven in 1976 by Appel and Haken, using an algorithm of Koch. Their proof reduced the infinitude of possible maps to 1936 configurations (later reduced to 1476), checked one by one by computer. Later, in 1996, it was proven that there are actually only 633 possible configurations.

[16]In mathematics, a conjecture is a mathematical statement which appears likely to be true, but has not been formally proven to be true under the rules of mathematical logic. Once a conjecture is formally proven true it is elevated to the status of theorem and may be used afterwards without risk in the construction of other formal mathematical proofs. Until that time, mathematicians may use the conjecture on a provisional basis, but any resulting work is itself conjectural until the underlying conjecture is cleared up.

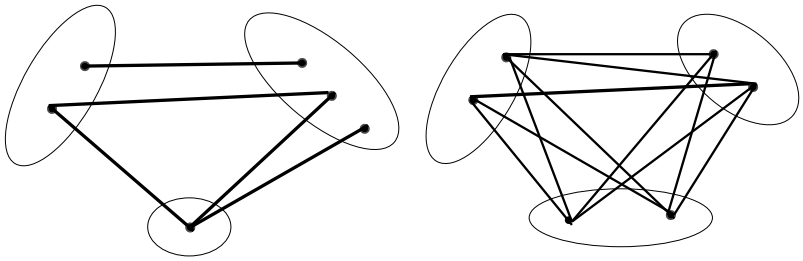


Figure 3.26: A 3-partite graph (left) and a 3-partite complete graph (right).

3.7 Bipartite graphs

Definition 3.42 *Let $r \in \mathbb{N},\ r \geq 2$. A graph $G = (V, E)$ is called r-**partite** if V admits a partition into r classes such that every edge has its endpoints in different classes (or, equivalently, vertices in the same partition class must not be adjacent).*

Remark 3.15 *A 2-partite graph is called **bipartite**. It is denoted $G = (V_1 \cup V_2, E)$, where V_1 and V_2 are the two classes of the partition. An immediate property in a bipartite graph is*

$$\sum_{v \in V_1} deg(v) = \sum_{v \in V_2} deg(v).$$

We give now an important characterization of bipartite graphs.

Theorem 3.37 (Characterization of bipartite graphs) *A graph is bipartite if and only if it contains no odd cycle.*

Proof.

$\Rightarrow$ Let $G = (V_1 \cup V_2, E)$ be bipartite graph. Then G either has no cycle, or, if there exists a cycle, it has the vertices successively in V_1 and V_2, with the last vertex in the same class as the first. Such a cycle has an even number of edges.

$\Leftarrow$ Let $G = (V, E)$ be a graph without odd cycle (G has therefore no cycle or even cycle). Let $G_0 = (V_0, E_0)$ a connected component of G and denote T_0 the spanning tree obtained by BFS algorithm, starting at a given vertex $s \in V_0$. Consider the sets

$$\begin{aligned} V_1 &= \{s\} \cup \{x \in V,\ d(x,s) = \text{ even}\}, \\ V_2 &= \{x \in V,\ d(x,s) = \text{ odd}\}. \end{aligned}$$

Then every edge $e_0 \in T_0$ has one endpoint in V_1 and the other in V_2. Let $e = \{x, y\}$ a non-tree edge. Then $d(x, s)$ and $d(y, s)$ are either equal or differ by 1 (see Proposition 3.25). If they were equal, e would close an odd cycle, fact which contradicts the hypothesis. In consequence

$$|d(x,s) - d(y,s)| = 1,$$

which implies that one of the vertices x, y is in V_1 and the other is in V_2.

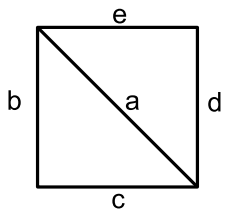


Figure 3.27: The graph in Example 3.17.

■

An application of bipartite graphs is the *job assignment*. If X is a set of people and Y a set of jobs, such that each person in X is qualified for some jobs in Y, then the question is: how can we assign people to jobs, so that a maximum number of people get jobs for which they are qualified?

In terms of graphs, we consider the bipartite graph $G = (X \cup Y, E)$, where $xy \in E$ if and only if x is qualified for job y. Such an assignment is called a **matching** in G.

3.7.1 Matchings

Definition 3.43 *Let $G = (V, E)$ be a graph.*

- *A set $M \subseteq E$ is called **matching** in G if no two edges in M are adjacent in G.*
- *A matching M **saturates** the vertex $v \in V$, and v is said to be M- saturated, if some edge of M is incident on v; otherwise v is **unsaturated** by M.*
- *A **perfect** matching is a matching M in which every vertex of G is M-saturated.*
- *If $e = xy \in M$, we say that M **matches** x with y (or x is matched with y by M).*
- *A **maximum**(-cardinality) matching is a matching with the greatest number of edges (no other matching has a greater cardinality).*
- *A **maximal** matching is a matching that is not a proper subset of any other matching in G.*

Example 3.17 *In the graph in Figure 3.27,*

$\{b\}, \{c\}, \{d\}, \{e\}$ *are matchings which are not maximal.*
$\{b, d\}, \{c, e\}$ *are maximum matchings,*
$\{a\}$ *is a maximal matching, but not a maximum matching.*

The commonest applications of matchings are machine scheduling and job assignment.

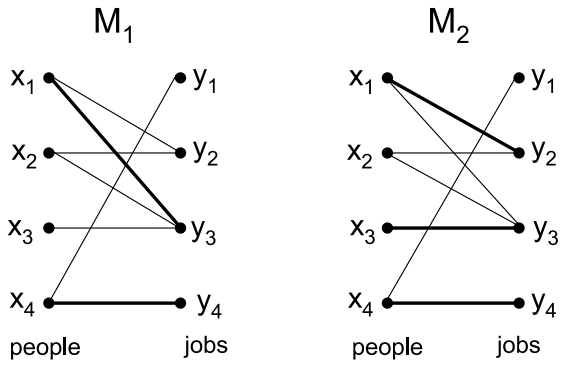


Figure 3.28: The matchings M_1 and M_2.

3.7.2 Matchings in bipartite graphs

Consider the bipartite graph in Figure 3.28. The set $M_1 = \{x_1y_3,\ x_4y_4\}$ is not a maximum matching, but $M_2 = \{x_1y_2,\ x_3y_3,\ x_4y_4\}$ is a maximum matching, since there is no other matching with more than three edges. Indeed, if we consider that X is a set of people and Y is a set of jobs, then it is impossible for all four people to get jobs for which they are qualified. Because $\{x_1, x_2, x_3\}$ are together qualified only for two jobs y_2 and y_3, one of them cannot get a job however the jobs are filled.

In many applications one wishes, if possible, to find a matching of G that saturates every vertex in X.

Definition 3.44 *If $|M| = |X|$ (all people get jobs), we say that M is a* ***complete matching****.*

Of course, M_1 and M_2 are not complete matchings. The first step in studying matchings is to decide if a complete matching is possible. We have seen in the above example that if there are three people and two jobs, a complete matching cannot exist. Afterwards, we are interested to find necessary and sufficient conditions for the existence of such a matching.

Let $G = (X \cup Y, E)$ a bipartite graph and $S \subseteq X$. We define the set of all vertices adjacent to vertices in S,

$$J(S) = \{y \in Y,\ xy \in E \text{ for some } x \in S\},$$

meaning the set of jobs for which people in S are collectively qualified.

Remark 3.16 *If $|J(S)| < |S|$, then someone in S will not get a job. So, if J is a complete matching, then*

$$|J(S)| \geq |S| \text{ for all } S \subseteq X. \tag{3.10}$$

The condition given in (3.10) is referred to as Hall's[17] condition.

Theorem 3.38 (Hall, 1935) *The bipartite graph $G = (X \cup Y, E)$ has a complete matching if and only if Hall's condition (3.10) is satisfied.*

Proof.

[17]Philip Hall (1904-1982), English mathematician.

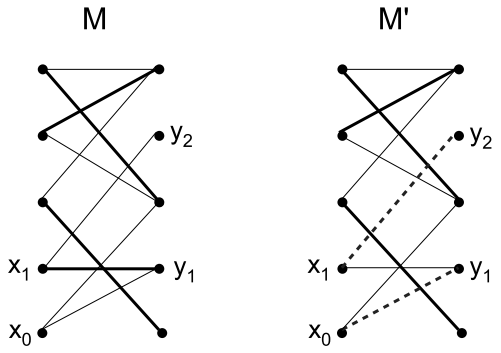


Figure 3.29: The matchings M and M' in the proof of Hall's theorem. The edge x_1y_1 in M is removed and the edges $x_0y_1,\ x_1y_2$ not in M are added.

$\Rightarrow$ Let M be a complete matching and let $S \subseteq X$. The vertices in Y matched by M with those in S form a subset of $J(S)$ with size $\geq |S|$. Hence, $|J(S)| \geq |S|$.

$\Leftarrow$ Suppose $|J(S)| \geq |S|$ for all $S \subseteq X$. Given a matching M for which $|M| < |X|$, we will show how to construct a matching M' with $|M'| = |M| + 1$ (such a matching M' will always exist !).

Let x_0 be any vertex not matched by M. Then

$$|J(\{x_0\})| \geq |\{x_0\}| = 1,$$

so there is at least one edge $e = x_0y_1$ in M.
If y_1 is unmatched, then $M' = M \cup \{e\}$.
If y_1 is matched in M, with x_1 say, then

$$|J(\{x_0, x_1\})| \geq |\{x_0, x_1\}| = 2,$$

so there exists $y_2,\ y_2 \neq y_1$, adjacent to x_0 or x_1.
If y_2 is unmatched, stop.
If y_2 is matched, to x_2 say, repeat the argument,
so there exists y_3, adjacent to at least one of x_0, x_1, x_2.
...
Continuing in this way, we must eventually stop at an unmatched vertex y_r (since G is finite).

Each $y_i,\ i \in \mathbb{N}_r$, is adjacent to at least one of the vertices $x_0, x_1, \ldots, x_{i-1}$, therefore we have a path

$$p = \langle y_r, x_{r-1}, y_{r-1}, \ldots, x_1,\ y_1, x_0 \rangle, \quad \text{in which } x_iy_i \in M.$$

We construct the matching M' such that x_iy_i of the path p are not in M', but the alternate edges are in M' (see Figure 3.29).

Since the terminal edges y_rx_{r-1} and y_1x_0 are both in M', we have $|M'| = |M| + 1$ (the number of added edges equals one plus number of removed edges).

■

To summarize, the idea of the proof is the construction of a path whose edges are alternately in M and not in M. In general, for a graph $G = (V, E)$ and

for a matching M in G, we say that a path is an **alternating path** for M if its edges are alternately in $E \setminus M$ and M. An **augmenting path** for M is an alternating path for M whose initial and final vertices are unsaturated by M. For example, an augmenting path for the matching M in Figure 3.29 is $\langle x_0, y_1, x_1, y_2 \rangle$.

Let us note that, since the first and last edges in an augmenting path do not belong to M, the path has one edge fewer in M than it has not in M.

From the proof of the theorem we can also deduce that, if Hall's condition is satisfied and M is not complete, then an augmenting path for M exist.

The idea of the proof constitutes also a practical device for the construction of complete matchings, supplying actually an algorithm for this construction.

Remark 3.17 *Hall's theorem is also known as* ***the marriage theorem****, due to the following interpretation: X is a set of men and Y is a set of women, some of those know each other. We require the condition that each man marries to one of the women he knows. In Hall's theorem, $J(S)$ denotes the set of women known by at least one man in S. A matching is a set of pairs who marry.*

3.7.3 Maximum matching

In general, a bipartite graph will not have a complete matching, so the problems which arise in this case are: how can we find the maximum size of a matching and how can we find an assignment which results in the largest possible number of people getting suitable jobs. The solution can be deduced from Hall's theorem.

First, let us observe that, if $|S| > |J(S)|$, then some people won't get jobs, actually at least $|S| - |J(S)|$ people won't get jobs. This remark suggests the following definition.

Definition 3.45 *The* ***deficiency*** *d of the bipartite graph $G = (X \cup Y, E)$ is defined as*

$$d = \max\{|S| - |J(S)|, \quad S \subseteq X\}.$$

Remark 3.18 *For $S = \emptyset$ we have $|S| = |J(S)| = 0$, so $d \geq 0$. Then, in terms of deficiency, Hall's theorem can be reformulated as follows: There exists a complete matching if and only if $d = 0$.*

The next theorem deals with the size of a maximum matching in the general case.

Theorem 3.39 *The size of a maximum matching M in a bipartite graph $G = (X \cup Y, E)$ is*

$$|M| = |X| - d,$$

where d is the deficiency of G.

Proof. From Definition 3.45 we deduce that there exists $S_0 \subseteq X$ such that

$$|S_0| - |J(S_0)| = d.$$

In any matching, at least d members of S_0 remain unmatched, so

$$|M| \leq |X| - d.$$

It remains to show that there exists a matching with size equal to $|X| - d$.

Let D be a new set, with size $|D| = d$. We construct the graph $G^* = (X^* \cup Y^*, E^*)$, with

$$\begin{aligned} X^* &= X, \\ Y^* &= Y \cup D, \\ E^* &= E \cup K, \end{aligned}$$

where K is the set of all possible edges linking X and D. Then, for the graph G^* we have $J^*(S) = D \cup J(S)$, hence

$$|J^*(S)| - |S| = d + |J(S)| - |S| \geq 0.$$

Thus, G^* satisfies Hall's condition and therefore G^* has a complete matching M^*.

By removing from M^* the d edges which have a vertex in D, we obtain the required matching in G. ■

Remark 3.19 *Theorem 3.39 is not very efficient to find the size of a maximum matching, because in order to calculate d we must examine all $2^{|X|}$ subsets of X.*

A more practical approach is based on the fact that, if we have an alternating path for a matching M, then we can construct a better matching M'. In order to make this idea work, we need the following result, known as Berge's theorem.

Theorem 3.40 ***(Berge,[18] 1957)*** *The matching M in a graph G is a maximum matching if and only if G contains no augmenting path for M.*

Proof.

$\Leftarrow$ We show that, if G contains no alternating path for M, then M is maximum. Suppose that M is not maximum and let M^* be a maximum matching. Then $|M^*| > |M|$.

Let F be the set of edges in

$$M \Delta M^* = (M \cup M^*) \setminus (M \cap M^*) = (M \setminus M^*) \cup (M^* \setminus M).$$

With the edges in F we form a graph H, in which every vertex will have the degree one or two, since it can be incident on at most one edge of M and one edge of M^*. So, the components of H are either paths or cycles and in each of these paths/cycles the edges in M alternate with edges not in M. Thus, in any cycle, the number of edges in M (in $M \setminus M^*$) equals the number of edges not in M (in $M^* \setminus M$). But,

$$|M^*| > |M| \quad \Rightarrow \quad |M^* \setminus M| > |M \setminus M^*|,$$

so it is impossible to have only cycles in F. In conclusion, there exists at least one component which is a path with an odd number of edges, and this is an augmenting path for M. This is a contradiction.

[18] Claude Berge (1926-2002), French mathematician.

$\Rightarrow$ Let M be a maximum matching. We prove that G contains no augmenting path for M. We suppose the contrary, that G contains an augmenting path in M, denoted $P = \langle e_1, e_2, \ldots, e_{2s+1} \rangle$. Since $e_1 \notin M$, we deduce that

$$e_1, e_3, \ldots, e_{2s+1} \notin M \quad \text{and} \quad e_2, e_4, \ldots, e_{2s} \in M.$$

Define the set $M_1 \subseteq E$ obtained from M by removing $e_2, e_4, \ldots, e_{2s}$ and adding $e_1, e_3, \ldots, e_{2s+1}$. Then M_1 is a matching in G which contains one more edge than M, a fact that contradicts the maximality of M. ■

From the proof of Berge's theorem, we can deduce the following algorithm for finding a maximum matching:

Algorithm 13 (find a maximum matching for a given M)
Input: *a graph G*
Output: *a maximum matching in G*

- *initialize M with any edge*
- $\circledast$ *search an augmenting path for M*
  If an augmenting path is found, construct a better matching M' in the usual way, and return to $\circledast$, with M' replacing M
  If no augmenting path can be found, STOP: M is a maximum matching.

The search of an augmenting path can be made by a modified BFS procedure. Choose an unmatched vertex x_0 and construct a tree of "partial" alternating paths, starting from x_0, as follows:

1. At level 1, insert all the vertices $y_1, y_2, \ldots, y_k$ adjacent to x_0 in G. If any one of these vertices y_i is unmatched, STOP: $\langle x_0, y_i \rangle$ is an augmenting path.
2. If all level-1-vertices $y_1, y_2, \ldots, y_k$ are matched, at level 2 insert the vertices $x_1, x_2, \ldots, x_k$ with which they are matched.
3. At level 3, insert all new vertices adjacent to the level-2-vertices. If any one of the them is unmatched, STOP: the path leading from this vertex x_0 is an augmenting path.
4. If all level 3 vertices are matched, insert at level 4 the vertices with which they are matched,
   and so on.

The construction may be halted because there are no new vertices to insert at an odd-numbered level. When this happens, there is no augmenting path beginning at the chosen vertex x_0. We must, however, repeat the procedure for every unmatched vertex in X before we can be sure that no alternating path can be found in G.

3.8 Hamiltonian graphs and Eulerian graphs

We begin by mentioning that all the results given in this section for graphs are also valid for multi-graphs, unless we specify the contrary.

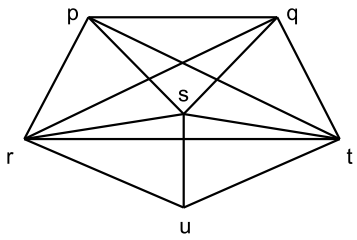


Figure 3.30: A map of towns and the roads linking them.

Consider the map in Figure 3.30, of six towns and the roads connecting them.

A highway engineer (E) wishes to inspect all the roads, in either direction, being prepared to start and finish in different places, whilst his friend, inspector (H), wishes to dine in a restaurant in each town. Each wishes to achieve his objective in a way as efficient as possible. So, (E) states his aim as "I wish, if possible, to visit every road once and only once", whilst (H) says "I wish, if possible, to visit each town once and only once and return to my starting point. The question is: "Can suitable routes for them be found"?

For inspector (H) one possibility is the closed walk

$$\omega = \langle p, q, t, s, u, r, p \rangle.$$

For engineer (E), let x denote the starting vertex and y the final vertex and suppose for the moment that $x \neq y$. He uses one edge incident on x when he starts, and each time he returns to x he must arrive and depart by new edges. In this way, he uses an odd number of edges at x, and thus the degree of x should be an odd number (x should be an odd vertex). Similarly, the degree of y should be odd. All the remaining vertices should be even, since every time he arrives at an intermediate vertex he also departs, and thereby uses two edges.

In conclusion, a route for (E), starting and finishing at distinct vertices x and y is possible if and only if x and y are odd vertices and the rest are even. In our case, the degrees are

v	p	q	r	s	t	u
$\delta(v)$	4	4	5	5	5	3

,

so there is no route for (E).

If $x = y$, again there is no solution, since all the vertices should be even.

3.8.1 Hamiltonian graphs

In general, (H)'s route is a cycle which contains all vertices of a given graph. Such cycles were first studied by Hamilton,[19] so a cycle with this property is called a **Hamiltonian cycle**. A graph which is connected and contains a Hamiltonian cycle is called a **Hamiltonian graph**. In the case when we do not impose the restriction that the initial and the final vertices coincide, such a path which visits each vertex exactly once is called a **Hamiltonian path**.

[19]William Rowan Hamilton (1805-1865), Irish mathematician, physicist and astronomer.

In general, it is not so easy to decide if a graph is Hamiltonian or not. Some results given sufficient conditions for the existence of a Hamiltonian path are the following:

Theorem 3.41 ***(Dirac,[20] 1952)*** *A graph with $n > 2$ vertices is Hamiltonian if each vertex has the degree $\geq n/2$. This theorem is valid only for graphs, not for multi-graphs.*

Theorem 3.42 ***(Ore,[21] 1960)*** *A connected graph with $n > 2$ vertices is Hamiltonian if the sum of the degrees of two non-adjacent vertices is $\geq n$. An immediate consequence is that all complete graphs are Hamiltonian.*

The best characterization of Hamiltonian graphs, which generalizes the previous results of Ore and Dirac, was given by in 1972 by Bondy[22] and Cvátal.[23] Before we give this theorem we need to introduce some terminology.

Definition 3.46 *Given a graph G with n vertices, the* ***closure*** *of G is the graph uniquely constructed from G by adding, for all non-adjacent pairs of vertices x, y with $deg(x) + deg(y) \geq n$, the edge xy.*

Theorem 3.43 (Bondy-Chvátal, 1972) *A graph is Hamiltonian if and only if its closure is Hamiltonian.*

As a consequence of this theorem, every platonic solid,[24] considered as a graph, is Hamiltonian.

Among the algorithms for constructing a Hamiltonian cycle we mention the Ham algorithm.

3.8.2 Eulerian graphs

The problem of finding a route for (E) was easily settled: the answer was "no". This problem was first discussed in 1736 by Euler, while solving the famous *Seven Bridges of Königsberg* problem.

The **Seven Bridges of Königsberg Problem** is a famous solved mathematics problem inspired by an actual place and situation. The city of Königsberg, Prussia (now Kaliningrad, Russia) is set on the Pregel River, and included two large islands which were connected to each other and the mainland by seven bridges (see Figure 3.31). The question is whether it is possible to walk with a route that crosses each bridge exactly once. In 1736, Leonhard Euler proved that it was not possible. In proving the result, Euler formulated the problem in terms of graph theory, by abstracting the case of Königsberg first, by eliminating all features except the landmasses and the bridges connecting them; second, by replacing each landmass with a dot (vertex), and each bridge with a line (edge), as in Figure 3.32. Thus, he introduces a new mathematical structure – the graph.

If we calculate the degrees of the vertices in the graph of the Seven Bridges Problem (Figure 3.32), we see that the requirement that all the vertices have even degree is not satisfied.

[20] Gabriel Andrew Dirac (1925-1984), British mathematician.
[21] Øystein Ore (1899-1968), Norwegian mathematician.
[22] John Adrian Bondy, American mathematician.
[23] Vašek Chvátal (b. 1946), Czech mathematician.
[24] The platonic solids are: tetrahedron, cube, octahedron, dodecahedron, icosahedron.

Figure 3.31: A map of Königsberg and its famous seven bridges.

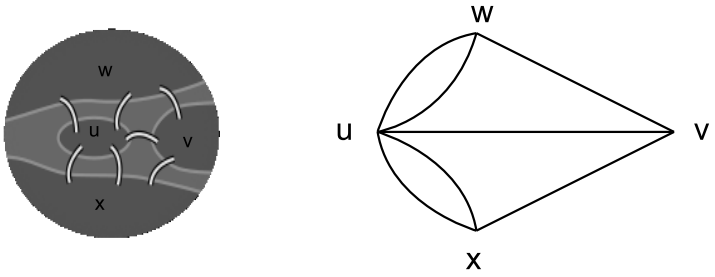


Figure 3.32: The Königsberg graph.

Definition 3.47 *A trail that uses each edge of the graph* exactly once *is called an* ***Eulerian trail****.*

An Eulerian trail which is closed is called ***Eulerian tour*** *or* ***Eulerian circuit****.*

A graph is called ***Eulerian graph*** *if it contains an Eulerian tour.*

As we have already mentioned, Euler observed that a necessary condition for the existence of Eulerian tours is that all vertices in the graph have an even degree; this means the Königsberg graph is not Eulerian. As for the existence of an Eulerian trail, either all, or all but two vertices should have an even degree. The natural question is whether these necessary conditions are also sufficient.

Carl Hierholzer published the first complete characterization of Eulerian graphs in 1873, by proving that in fact the Eulerian graphs are exactly the graphs which are connected and where every vertex has an even degree.

Theorem 3.44 (Hierholzer, 1873) *Let $G = (V, E)$ be a connected graph. Then the following statements are equivalent:*

a) G is Eulerian

b) Every vertex has even degree

c) G is the union of edge-disjoint cycles

Proof.

$a) \Rightarrow b)$ This implication was already discussed.

$b) \Rightarrow c)$ Since every vertex has even degree, G cannot be a tree, therefore G contains a cycle, say C_1.

If $C_1 = G$, then the conclusion is proved.

If not, consider the graph $G_1 = G - C_1$. Since the edges that were removed from G form a cycle, the degree in G_1 of each vertex in C_1 is reduced by two, and therefore every vertex of G_1 is even. Hence, G_1 is not a tree, therefore it contains a cycle C_2.

If $G = C_1 \cup C_2$, the conclusion is proved.

If not, we continue in the following way and we must finish with the conclusion $G = C_1 \cup C_2 \cup \ldots \cup C_n$.

$c) \Rightarrow a)$ Let T^* be a closed trail in G of maximum length. According to Proposition 3.6, T^* is the union of edge-disjoint cycles.

If T^* includes all edges in G, then the conclusion is proved.

If not, then there exists a cycle, say C, which is not a part of T^*. Since G is connected, there exists a vertex $v \in C^* \cap C$ and a closed trail obtained in the following way:

- traversing T^* until v is first encountered
- "detouring" around C back to v
- traversing then the remaining portion of T^*

By constructing this trail, we obtain a contradiction with the maximality of T^*. In conclusion, the existence of C which is not a part of T^* is false, and therefore T^* includes all edges in G.

■

Constructing Eulerian trails and tours

Being given a connected graph with at most two vertices of odd degree, we can construct an Eulerian trail or an Eulerian tour out of this graph by using Fleury's algorithm, which dates from 1883. We start with a vertex of odd degree if the graph has none, then start with any vertex. At each step we move across an edge whose deletion does not result into two connected components, unless we have no choice, then we delete that edge. At the end of the algorithm there are no edges left, and the sequence of edges we moved across forms an Eulerian tour if the graph has no vertices of odd degree or an Eulerian trail if there are two vertices of odd degree.

Definition 3.48 *For an arbitrary graph G, an edge e with the property that $G - e$ has more components than G is called a* ***bridge****.*

Algorithm 14 (Fleury, 1883)
Input: *- a connected graph $G = (V, E)$, all of whose vertices having (a) even degree; (b) exactly two vertices x, y of odd degrees*
- a starting vertex v_0
Output: *(a) an Euler tour; (b) an Euler (x, y)-trail*

- *set current trail as empty, current vertex as v_0 and $A = E$*
- *while $A \neq \emptyset$*
  - *select an edge e incident on the current vertex, but choosing a bridge only when there is no alternative*

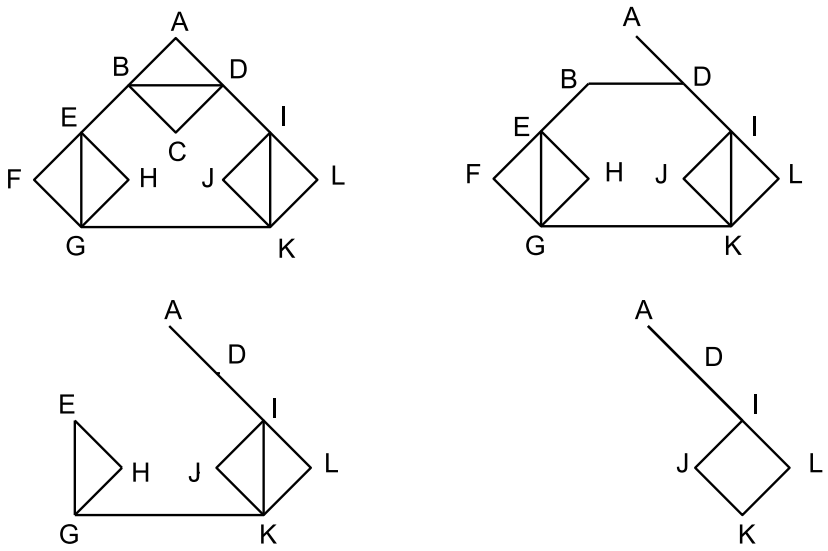


Figure 3.33: The steps in Example 3.18.

- *add e to the current trail*
- *set the current vertex as the other point of edge e*
- *set $A = A \setminus \{e\}$*
- *delete any isolated vertices*

▪ *return an Eulerian tour in case (a) and an Eulerian (x, y)-trail in case (b)*

Example 3.18 *Consider the graph in Figure 3.33 (top left). We will take the vertex A as the starting vertex. The first edges considered in the trail C will be AB, BC, CD. At this point, the remained graph is depicted in Figure 3.33 (top right) and the current vertex is D. The edge DA becomes a bridge, so we will take in C the edges DB, BE, EF, FG, the remained graph being the one in Figure 3.33 (bottom left). Then GK is a bridge, therefore we will choose GE, EH, HG, GK, KI. Next, ID is a bridge, so we choose IJ, JK, KL, LI, ID, DA.*

As applications of Eulerian graphs we mention garbage collection, street sweeping, snow-plowing, line-painting down the center of the street, post delivering.

3.8.3 The postman problem

In 1962, the Chinese mathematician Meigu Guan introduced the problem of finding a shortest closed walk to traverse every edge of a graph at least once. He envisioned a postman who wants to deliver the mail through a network of streets and return to the post office as quickly as possible. J. Edmonds dubbed this problem *The Chinese Postman Problem.*

Definition 3.49 *A **postman tour** in a graph G is a closed walk that uses each edge of G at least once.*

*In a weighted graph, an **optimal postman tour** is a postman tour whose total edge-weight is minimum.*

Of course, if each vertex of the graph has even degree, an Eulerian tour is an optimal postman tour. Otherwise, some edges must be retraced (deadheaded). So, the goal is to find a postman tour whose deadheaded edges have minimum weight. This corresponds, in fact, to an Eulerian tour of the graph G^*, formed from G by adding as many additional copies of an edge as the number of times it was deadheaded during the postman tour.

In 1973, Edmonds and Johnson solved the Chinese Postman Problem using the following algorithm:

Algorithm 15 (Construction of an optimal postman tour)
Input: *a connected weighted graph G*
Output: *an optimal postman tour W*

- *find the set S of odd-degree vertices of G*
- *for each pair of odd-degree vertices $u, v \in S$, find*
  $d(u, v)$= the distance between u and v
- *form a complete graph K on the vertices of S and in K assign the weight $d(u, v)$ to each edge uv*
- *find a perfect matching M in K whose total weight is minimum*
- *for each edge e in M*
  - *let p be the corresponding shortest path in G between the endpoints of edge e*
  - *for each edge f on path p, add to graph G a duplicate copy of edge f, including its weight*
- *Let G^* be the Eulerian graph formed by adding to graph G the edge duplications from the previous step*
- *construct an Eulerian tour W in G^*. This tour will correspond to the optimal tour of the original graph G*

Remark 3.20 *The optimality of the postman tour follows from having chosen a minimum weight perfect matching.*

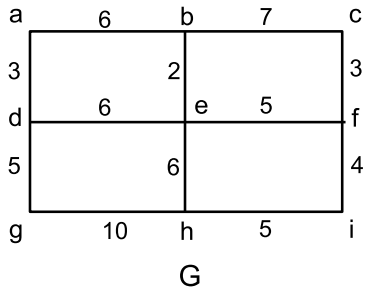


Figure 3.34: The weighted graph G in Example 3.19.

Example 3.19 *Consider the weighted graph G in Figure 3.34. The vertices of odd degree are b, d, f, h. They will form the complete graph K in Figure 3.35 (left). A perfect matching in K with minimum weight is $M = \{bd, fh\}$. Each edge in M represents a path in G: for the edge bd of weight 8 we have the path in G $\langle b, e, d\rangle$ (of length 8), while for the edge fh of length 9 we have the path $\langle f, i, h\rangle$. The graph G^* is represented in Figure 3.35 (right).*

Finally, to construct an Eulerian tour we apply Fleury's algorithm and we obtain

$$W = \langle a, b, c, f, e, b, e, d, e, h, i, f, i, h, g, d, a\rangle.$$

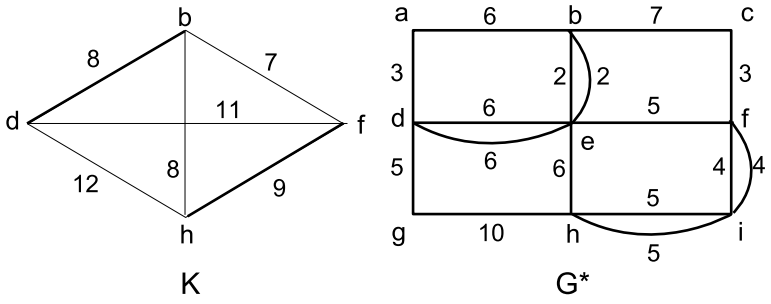


Figure 3.35: The complete graph K and the Eulerian graph G^*.

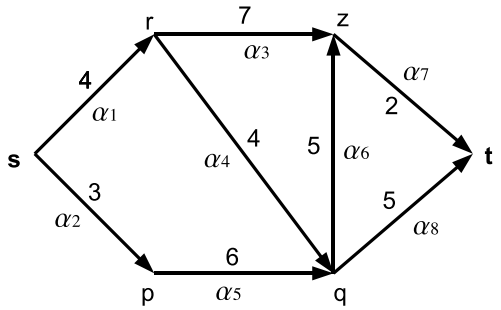


Figure 3.36: The digraph associated to the project in Example 3.20.

3.9 Networks

3.9.1 Critical paths

Let $D = (V, A)$ be a digraph and $w : A \to \mathbb{N}$ be a weight function representing costs, distances or time. This digraphs can be used in planning a project, when the project can be broken down into small activities which are related. For instance, the project of building a house can be broken down into many small activities as laying foundations, bricklaying, roofing, installing electrical network, etc. These activities are of course related, in the sense that on cannot begin some of them until some of the other activities have been completed. The digraph associated to this type of project will be constructed as follows:

- the arcs represent activities,
- the vertices represent events (an event is the completion of some activities),
- the weight of an arc is the time required for that activity .

Example 3.20 *We want to schedule the activities $\alpha_1, \ldots, \alpha_8$ such that the total time required for the project is as small as possible, knowing the times and the prerequisites for each α_i:*

Activity	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_7
Time needed (days)	4	3	7	4	6	5	2	5
Prerequisites	–	–	α_1	α_1	α_2	α_4	α_3	α_4
						α_5	α_6	α_5

Thus, for instance, α_6 cannot start until α_4 and α_5 are completed. We are interested in finding the minimum number of days needed for the whole project. The digraph associated to the project is depicted in Figure 3.36. There, the vertex s is the event "the project starts", t represents the event

"the project is accomplished", while an intermediate vertex, say q, represents the event "activities α_4 and α_5 are completed".

For $v \in V$ we denote by $E(v)$ the earliest time for the event corresponding to v. Thus, we have

$$E(s) = 0,\ E(p) = 3,\ E(r) = 4.$$

Then, at q both α_4 and α_5 must be completed, so

$$E(q) = \max\{E(r) + w(r,q),\ E(p) + w(p,q)\} = \max\{4+4,\ 3+6\} = 9.$$

In general, for the calculation of $E(v)$ we have the formulas

$$\begin{cases} E(s) = 0, \\ E(v) = \max\limits_{u \in A^-(v)} \{E(u) + w(u,v)\}, \end{cases}$$

where $A^-(v)$ is the set of tails of the arcs with heads in v. For the remaining vertices one has

$$E(z) = \max\{E(q) + w(q,z),\ E(r) + w(r,z)\} = 14,$$
$$E(t) = \max\{E(z) + w(z,t),\ E(q) + w(q,t)\} = 16.$$

Therefore, the earliest time for the completion of t is $E(t) = 16$, meaning that the minimum number of days needed for the project is 16. Actually, this number represents the length of the longest path from s to t, and it can be also found by applying to D a BFS algorithm for digraphs.

The method described above is a part of a technique called *critical path analysis*. The rest of the technique is as follows: for each $v \in V$, we calculate the numbers $L(v)$, representing the latest time by which all activities (v,x) must be started, if the whole project is to be completed on time. The formulas for $L(v)$ are:

$$\begin{cases} L(t) = E(t), \\ L(v) = \min\limits_{x \in A^+(v)} \{L(x) - w(v,x)\}, \end{cases}$$

where $A^+(v)$ is the set of heads of the arcs with tails in v. Thus, about an activity (y,z) we know that:

- it cannot start before time $E(y)$, at the earliest;
- it must finish by time $L(z)$, at the latest;
- it takes time $w(y,z)$.

Definition 3.50 *For an activity (y,z), the **float time** $F(y,z)$ is defined as*

$$F(y,z) = L(z) - E(y) - w(y,z).$$

Thus, (y,z) can start at any time after $E(y)$ and before $E(y) + F(y,z)$, without delaying the project.

Definition 3.51 *An activity (y,z) for which the float time $F(y,z)$ is zero is said to be **critical**.*

A critical activity must be started at the earliest possible time $E(y)$, if the project is to finish on time. In the digraph associated to the project there will be at least one directed path from s to t consisting entirely of critical activities, and this is called a **critical path**. This path is in fact the unique path in the BFS spanning tree of the digraph, where the default priority is the longest weight.

For our example, the numbers L are:

$$\begin{aligned}
&L(t) = 16,\\
&L(q) = \min\{L(t) - w(q,t), L(z) - w(q,z)\} = 9,\\
&L(p) = L(q) - w(p,q) = 3,\\
&L(r) = \min\{L(z) - w(r,z), L(q) - w(r,q)\} = 5,\\
&L(s) = \min\{L(r) - w(s,r), L(p) - w(s,p)\} = 0.
\end{aligned}$$

The float times will be:

$$\begin{aligned}
F(s,r) &= L(r) - E(s) - w(r,s) = 5 - 0 - 4 = 1,\\
F(s,p) &= L(p) - E(s) - w(p,s) = 3 - 0 - 3 = 0,\\
F(p,q) &= L(q) - E(p) - w(p,q) = 9 - 3 - 6 = 0,\\
F(r,q) &= L(q) - E(r) - w(r,q) = 9 - 4 - 4 = 1,\\
F(r,z) &= L(z) - E(r) - w(r,z) = 14 - 4 - 7 = 3,\\
F(z,t) &= L(t) - E(z) - w(t,z) = 16 - 14 - 2 = 0,\\
F(q,z) &= L(z) - E(q) - w(q,z) = 14 - 9 - 5 = 0,\\
F(q,t) &= L(t) - E(q) - w(q,t) = 16 - 9 - 5 = 2.
\end{aligned}$$

Therefore, the activities $(s,p) = \alpha_2$, $(p,q) = \alpha_5$, $(q,z) = \alpha_6$, $(z,t) = \alpha_7$ are critical activities. About the activities that are not critical, one can say the following:

$(s,r) = \alpha_1$ can start after $E(s) = 0$ and before $E(s) + 1 = 1$,
$(r,z) = \alpha_3$ can start after $E(r) = 4$ and before $E(r) + 3 = 7$,
$(r,q) = \alpha_4$ can start after $E(r) = 4$ and before $E(r) + 1 = 5$,
$(q,t) = \alpha_8$ can start after $E(r) = 9$ and before $E(r) + 2 = 11$.
One can easily see that a critical path is

$$\langle s, p, q, z, t\rangle.$$

3.9.2 Flows and cuts

In a directed graph, one can regard the set of arcs as a network of pipelines along which some commodity can flow, the weight of an arc representing in this case its capacity. In such a graph, there are two vertices, s (the source) and t (the sink), having a special role:

- all arcs containing s are directed away from s;
- all arcs containing t are directed towards t.

So, we deal with a connected digraph $D = (V, A)$, a **capacity function** $c : A \to \mathbb{R}_+$, a **source** s and a **sink** t. Such a digraph will be called a **transportation network**, or simply **network**.

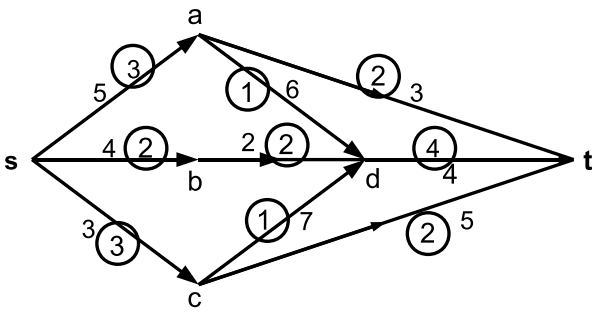


Figure 3.37: The network in Example 3.21.

Suppose a commodity is flowing along the arcs of the network and let $f(x,y)$ be the amount which flows along the arc (x,y). The amount of flow which arrives at a vertex v should be equal to the amount of flow leaving v, except the vertices s and t. If we define

$$\begin{aligned} inflow\,(v) &= \sum_{(x,v)\in A} f(x,v), \\ outflow\,(v) &= \sum_{(v,y)\in A} f(v,y), \end{aligned}$$

this requirement can be written as $inflow\,(v) = outflow\,(v)$, for $v \neq s,t$. We should also require that no arc carries flow exceeding its capacity. In conclusion, we give the following definition:

Definition 3.52 *A **flow** from the source s to the sink t in a network is a function which assigns a nonnegative number $f(x,y)$ to the arc (x,y), such that:*

a) $inflow(v) = outflow(v)$, for $v \neq s,t$ (conservation rule);

b) $f(x,y) \leq c(x,y)$, for all $(x,y) \in A$ (feasibility rule).

Of course, nothing is allowed to accumulate at intermediate vertices, and this means

$$outflow(s) = inflow(t).$$

This common value is called the **value** of the flow f and is denoted $val(f)$.

Example 3.21 *Consider the network given in Figure 3.37. The function f defined in the following table is a flow in this network and has the value $val(f) = 8$.*

(x,y)	*(s,a)*	*(s,b)*	*(s,c)*	*(a,d)*	*(b,d)*	*(c,t)*	*(a,t)*	*(c,t)*	*(d,t)*
f(x,y)	*3*	*2*	*3*	*1*	*2*	*1*	*2*	*2*	*4*

Our aim now is to calculate the maximum value of a flow for the network given in Figure 3.37.

The first step is to find an upper bound for this value in terms of capacities. Since from s one can transport $5+4+3=12$, the flow should be ≤ 12. The idea is to partition the vertex set into two parts, S containing s and T containing t. Thus, the flow from S to T equals the flow from s to t and this common value is $val(f)$. The value $val(f)$ can be written as

$$val(f) = \sum_{x\in S,\, y\in T} f(x,y) - \sum_{u\in T,\, v\in S} f(u,v),$$

where the first sum represents the total flow from S to T and the second sum represents the total flow in reverse direction. In our example, if $S = \{s, a, b, c\}$ and $T = \{d, t\}$, then, indeed

$$val(f) = f(a,d) + f(a,t) + f(b,d) + f(c,d) + f(c,t) - 0 = 8.$$

Of course, the same value is obtained, for the partition $\{S, T\}$ with $S = \{s, b\}$, $T = \{a, c, d, t\}$:

$$val(f) = f(s,a) + f(s,c) + f(b,d) - 0 = 8.$$

Also, let us mention that, for the last partition

$$val(f) \leq c(s,a) + c(s,c) + c(b,d) = 10. \tag{3.11}$$

At this point we need to give the following definition.

Definition 3.53 *In a network $D = (V, A)$, with source s, sink t and capacity $c : A \to \mathbb{R}_+$, the pair (S, T) is called a **cut** if $\{S, T\}$ is a partition of V such that $s \in S$ and $t \in T$. The **capacity** of the cut is defined as*

$$cap\,(S,T) = \sum_{x \in S,\, y \in T} c(x,y).$$

Then, the following result is immediate:

Theorem 3.45 *Let s be the source and t be the sink in a network. If $f : A \to \mathbb{R}_+$ is any flow from s to t and (S, T) is a cut, then*

$$val(f) \leq cap\,(S,T).$$

3.9.3 Max flow, min cut

An immediate consequence is the following result: if f_0 is a flow with maximum possible value and (S_0, T_0) is a cut with minimum capacity, then

$$\begin{array}{c} val(f_0) \leq cap\,(S_0,T_0), \quad \text{meaning that} \\ \text{max-flow} \leq \text{min-cut} \end{array}$$

Actually, the two values are equal, and this will be proved in Theorem 3.46.

The idea used for constructing a max-flow is to increase the value of a given flow, if the flow does not have the maximum possible value. We will illustrate this idea for the network in Figure 3.37 and the flow given in Example 3.21.

There are two types of improvements:

Type 1 improvement: Consider the path $\langle s, a, t\rangle$. Neither (s, a), nor (a, t) carry flow to its full capacity, therefore we can increase the flow on both arcs, until the capacity of one of them is reached. If we define

$$\begin{aligned} f_1(s,a) &= 4, \\ f_1(a,t) &= 3, \end{aligned}$$

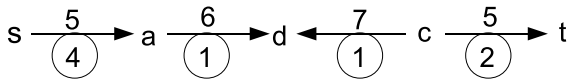
then (a, t) is saturated. Further, since the flows on both arcs have been increased by the same amount, since the conservation rule still holds at a, one has

$$f_1(x,y) = f(x,y)$$

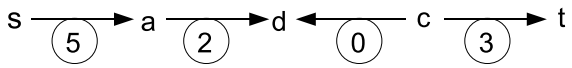
on the remaining arcs. Therefore, we have obtained a new flow f_1 with

$$val(f_1) = val(f) + 1 = 9.$$

Type 2 improvement: Consider the path $\langle s, a, d, c, t\rangle$ in the underlying graph.



This is not a path in the network since (d, c) is not an arc. The arc (c, d) is contrary to the direction of the path, therefore we can reduce the flow on (c, d) by 1 and increase the flows on the other arcs of the path by 1, without violating the conservation rule. Thus, we can define a new flow f_2 as



and $f_2 = f_1$ for the rest of the arcs. The arc (s, a) is thus saturated and the flow on (c, d) cannot be negative, so we cannot make any greater change on this path. The value of the new flow is

$$val(f_2) = val(f_1) + 1 = 10,$$

and it is the maximum value, as we have seen in (3.11). Therefore, f_2 is a maximum flow.

The paths $\langle s, a, t\rangle$ and $\langle s, a, d, c, t\rangle$ used to augment the flows f and f_1 are *flow-augmenting paths.*

Definition 3.54 *Given a flow f in a network, a path $p = \langle s = x_1, x_2, \ldots, x_{k-1}, x_k = t\rangle$ in the underlying graph is called an*
*f–**augmenting path** if*

$$f(x_i, x_{i+1}) < c(x_i, x_{i+1}) \quad and \quad (x_i, x_{i+1}) \in A \tag{3.12}$$

$$or \qquad f(x_{i+1}, x_i) > 0 \quad and \quad (x_{i+1}, x_i) \in A, \tag{3.13}$$

for $1 \leq i \leq k-1$.

In fact, (3.12) means that the forward arcs are not used at their full capacity, while (3.13) means that the backward arcs are carrying some "contra-flow". In fact, the f–augmenting paths are the paths which can be "improved". Given such a path, we can increase flow on the forward arcs and decrease the flow on the backward arcs by the same amount, without violating the conservation rule.

The greatest change (without overloading the forward arcs or making the flow on the backward arcs negative) is the minimum in the range $1 \leq i \leq k-1$ of the quantities

$$\alpha(i) = \begin{cases} c(x_i, x_{i+1}) - f(x_i, x_{i+1}), & \text{if } (x_i, x_{i+1}) \in A, \\ f(x_{i+1}, x_i), & \text{if } (x_{i+1}, x_i) \in A. \end{cases}$$

This minimum will be denoted by α and will be called **residual capacity** of the path p. If we add α to the flow on forward arcs and subtract α from the backward arcs we obtain the flow f^* with

$$val(f^*) = val(f) + \alpha > val(f),$$

therefore f was augmented.

In conclusion, the existence of an augmenting path from s to t enables us to find a new flow f^* with $val(f^*) > val(f)$. This idea will be used in order to prove the next theorem. In the proof we need the following definition.

Definition 3.55 *An **incomplete** f–augmenting path is a path satisfying the conditions* (3.12) – (3.13) *for an f–augmenting path, except that the final vertex is not t.*

Theorem 3.46 (Max-flow, min-cut theorem) *The maximum value of a flow from s to t in a network is equal to the minimum capacity of a cut separating s and t:*

$$\boxed{max\text{-}flow = min\text{-}cut.}$$

Proof. Let f be a maximum flow. We define the sets of vertices

$$S = \{x \in V, \text{ there is an incomplete } f\text{–augmenting path from } s \text{ to } x\},$$
$$T = V \setminus S.$$

Then $t \in T$, otherwise there would exist an f–augmenting path from s to t and f could be augmented, contrary to the hypothesis that f is a maximum flow. Therefore (S, T) is a cut.

We have to prove that

$$cap\,(S, T) = val(f).$$

▪ Let (x, y) be an arc with $x \in S,\ y \in T$. From the definition of S, there exists an incomplete f–augmenting path from s to x. If $f(x, y) < c(x, y)$, then we could extend this path to y, contradiction with $y \in T$. Hence,

$$f(x, y) = c(x, y).$$

▪ Let (u, v) be an arc with $u \in T,\ v \in S$. Then, there exists an incomplete f–augmenting path from s to v. If $f(u, v) > 0$, we could extend the incomplete path to u, contradiction with $u \in T$. Hence,

$$f(u, v) = 0.$$

In conclusion,

$$val(f) = \sum_{x \in S,\, y \in T} f(x, y) - \sum_{u \in T,\, v \in S} f(u, v) = \sum_{x \in S,\, y \in T} c(x, y) = cap\,(S, T).$$

If (S', T') is another cut, then

$$cap\,(S', T') \geq val(f) = cap\,(S, T),$$

whence the cut (S, T) has minimum capacity. Therefore the theorem is proved. ■

3.9.4 Algorithms for finding an integer max flow

Along this section, we consider the network $D = (V, A)$ with source s, sink t and capacity $c : A \to \mathbb{R}_+$.

The first algorithm, constructed by Ford and Fulkerson, is based on the idea used in the proof of Theorem 3.46: given a flow f, we look for an f–augmenting path and we construct a new flow f^*, as described at page 131.

For the second algorithm, constructed by Edmonds and Karp, we need the following definition.

Definition 3.56 *For a flow f and a tree T in a network $D = (V, A)$, an arc (x, y) is called a* ***usable frontier arc*** *for f if:*

$$x \in T,\ y \notin T \quad and \quad f(x,y) < c(x,y)$$
$$or \qquad x \notin T,\ y \in T \quad and \quad f(x,y) > 0.$$

We will give now an algorithm for constructing a tree T with usable frontier arcs.

Algorithm 16 (Edmonds, Karp)
Input: *a network $D = (V, A)$ with source s, sink t, capacity $c : A \to \mathbb{N}$ and a flow $f : A \to \mathbb{N}$*
Output: *a tree T*

- *initialize $T = \{s\}$*
- *put label 0 on vertex s*
- *i=1*
- *while T doesn't contain vertex t and there exist usable frontier arcs for T*
  - *update the set of usable frontier arcs for T*
  - *let $a = (x, y)$ be the usable frontier arc with the labeled endpoint having the smallest label*
  - *add to tree T the arc a and the unlabeled endpoint of a*
  - *put label i on the unlabeled endpoint of a*
  - *i=i+1*
- *return tree T*

The connection between the tree T and the maximum flow is given in the following theorem.

Theorem 3.47 *Let $D = (V, A)$ be a network with source s, sink t, capacity $c : A \to \mathbb{N}$ and a flow $f : A \to \mathbb{N}$. If the tree T which results by applying Algorithm 16 contains the sink t, then the unique path in T from s to t is an f–augmenting path. If T does not contain t, then f is a maximum flow.*

The proof of this theorem can be found in [13].

Binary relations

- The **cartesian product** of two sets A and B is the set

$$A \times B = \{(x, y),\ x \in A, y \in B\}.$$

- A **binary relation** is a triple (A, B, R), when A, B are arbitrary sets and $R \subseteq A \times B$.
- If $(x, y) \in R$, then x is said to be **related** to y (by R). This is often denoted $x\,R\,y$.
- The relation (A, A, R), $R \subseteq A \times A$ is said to be a binary relation on the set A.
- A binary relation on A is called:
  - **reflexive** if $x\,R\,x$ for all $x \in A$
  - **transitive** if $x\,R\,y$ and $y\,R\,z$ implies $x\,R\,z$
  - **symmetric** if $x\,R\,y$ implies $y\,R\,x$
  - **antisymmetric** if $x\,R\,y$ and $y\,R\,x$ implies $x = y$

A.1 Equivalence relations

Definition .1 *A relation $R \subseteq A \times A$ is an **equivalence relation** on A if R is reflexive, transitive and symmetric.*

Example .1 *Let $R \subseteq \mathbb{Z} \times \mathbb{Z}$, given by*

$$x\,R\,y \quad \Longleftrightarrow \quad x - y \in 3\mathbb{Z},$$

where $3\mathbb{Z} = \{3k,\ k \in \mathbb{Z}\}$. Then R is an equivalence relation on $\mathbb{Z}$.

Definition .2 *Let R be an equivalence relation on a set A and let $x \in A$. The **equivalence class of** x, denoted $\widehat{x}$ or $R\langle x\rangle$, is the set*

$$\widehat{x} = \{y \in A,\ x\,R\,y\}.$$

*The set of all classes of equivalence, denoted A/R, is called the **quotient set**:*

$$A/R = \{\widehat{x},\ x \in A\}.$$

Example .2 *For the equivalence relation defined in Example .1,*

$$\begin{aligned}\widehat{0} &= \{3k,\ k\in\mathbb{Z}\},\\ \widehat{1} &= \{3k+1,\ k\in\mathbb{Z}\},\\ \widehat{2} &= \{3k+2,\ k\in\mathbb{Z}\}.\end{aligned}$$

Observe that the equivalence classes are pairwise disjoint and any integer is in one of these classes.

Definition .3 *A collection of subsets $\{S_1, S_2, \ldots, S_n\}$ of a set A is a **partition** of A if the two following conditions are satisfied*

- $S_i \cap S_j = \emptyset$, *for all* $1 \le i < j \le n$,
- $\bigcup_{i=1}^{n} S_i = A$.

Proposition .48 *Let R be an equivalence relation on a set A and let $x, y \in A$. Then the following statements are equivalent:*

1. $x\,R\,y$,
2. $\widehat{x} = \widehat{y}$,
3. $\widehat{x} \cap \widehat{y} \neq \emptyset$.

Corollary .49 *Let R be an equivalence relation on a set A. Then the equivalence classes form a partition of A:*

$$\{\widehat{x},\ x\in A\} \quad \textit{is a partition.}$$

Conversely, given a partition $\{S_1, S_2 \ldots, S_n\}$ of a set A, there exists an equivalence relation on A. It is defined as follows:

$$x\,R\,y \iff \exists\ i\in\mathbb{N}_n \textit{ such that } x, y \in S_i.$$

Example .3 *The equivalence classes $\widehat{0}, \widehat{1}, \widehat{2}$ of Example .2 form a partition of $\mathbb{Z}$. Indeed,*

$$\widehat{0}\cup\widehat{1}\cup\widehat{2} = \mathbb{Z} \textit{ and } \widehat{0}\cap\widehat{1} = \emptyset,\ \widehat{0}\cap\widehat{2} = \emptyset,\ \widehat{1}\cap\widehat{2} = \emptyset.$$

The quotient set in this case is

$$\mathbb{Z}/R = \{\widehat{0}, \widehat{1}, \widehat{2}\} = \mathbb{Z}_3.$$

A.2 Ordered relations

Definition .4 *Let A be a set and $R \subseteq A \times A$.*

*R is an **partial order relation** on A if R is reflexive, transitive and antisymmetric. In this case, the pair (A, R) is called **partially ordered set**, **poset**, or just **ordered set** if the intended meaning is clear.*

*R is a **total order relation** if it is a partial order relation and for every $x, y \in A$ one has $x\,R\,y$ or $y\,R\,x$ (that is every two elements are related). In this case the pair (A, R) is called **totally ordered set**.*

Example .4 *$(\mathbb{R}, \le)$ is a totally ordered set, but $(\mathcal{P}(\mathbb{R}), \subseteq)$ is only a poset. Here $\mathcal{P}(\mathbb{R}) = \{S,\ S \subseteq \mathbb{R}\}$.*

Special elements

In posets there may be some elements that play a special role.

Definition .5 *Let $(A, \leq)$ be a poset.*

The element $x \in A$ is called ***the least element*** *of A if*

$$x \leq y, \quad \text{for all } y \in A.$$

The element $z \in A$ is called ***the greatest element*** *of A if*

$$y \leq z, \quad \text{for all } y \in A.$$

The element $m \in A$ is called a ***minimal element*** *of A if*

$$x \leq m \text{ for some } x \in A \quad \Longrightarrow \quad x = m.$$

The element $M \in A$ is called a ***maximal element*** *of A if*

$$M \leq x \text{ for some } x \in A \quad \Longrightarrow \quad x = M.$$

What is important to note about maximal elements is that they are in general not the greatest elements, i.e., they do not have to be greater than all other elements. Indeed, consider $(A, \subseteq)$, with

$$A = \{\{n\},\ n \in \mathbb{N}\}.$$

It consists only of maximal elements, but has no greatest element. Moreover, all elements of A are minimal. This example also shows that maximal elements are usually not unique and that it is possible for an element to be both maximal and minimal at the same time.

If a subset has a greatest element, then this is the unique maximal element. Conversely, even if a set has only one maximal element, it is not necessarily the greatest one.

Yet, in a totally ordered set, the terms maximal element and greatest element coincide, this is why both terms are used interchangeably.

Bibliography

[1] M. Aigner, *Discrete Mathematics,* American Mathematical Society, Rhode Island, 2007.

[2] N. L. Biggs, *Discrete Mathematics,* Oxford University Press, 2005.

[3] J.A. Bondy and U.S. Murty, *Graph Theory with Applications*, Elsevier North-Holland, 1982.

[4] I.N. Bronshtein, K.A. Semendyayev, *Handbook of Mathematics*, Spinger, 1997.

[5] K. Devlin *Partida neterminată*, Ed. Humanitas, Bucureşti 2015.

[6] R. Durret, *The Essentials of Probability,* Duxbury Press, 1994.

[7] J. Gross, J. Yellen, *Graph Theory and its Applications*, CRC Press, 1999.

[8] H. Lisei, *Probability Theory,* Casa cărţii de Ştiinţă, Cluj-Napoca, 2004.

[9] H. Lisei, S. Micula, A. Soós, *Probability Theory through Problems and Applications*, Cluj University Press, 2006.

[10] J. Michael, J. Gross, J. Grossman, S. Douglas (editors), *Handbook of Discrete and Combinatorial Mathematics*, CRC Press, 2000.

[11] I. Mihoc, *Calculul probabilităţilor şi statistică matematică*, litogr. Univ. Babeş-Bolyai, 1994.

[12] D. Stirzaker, *Elementary Probability,* Cambridge University Press, 1995.

[13] N. Vornicescu, *Grafe: teorie şi algoritmi*, Ed. Mediamira, 2005.

[14] D. Wackerly, W. Menderhall and R.L. Scheaffer, *Mathematical Statistics with Applications*, Duxbury Press, 2001.

[15] *Wikipedia: The Free Encyclopedia*, http://www.wikipedia.org